FFAG 2010 WORKSHOP
KURR-Institute, 25-31 October 2010

NUMERICAL INTEGRATION

A TUTORIAL

A COURSE FOR (POTENTIAL)

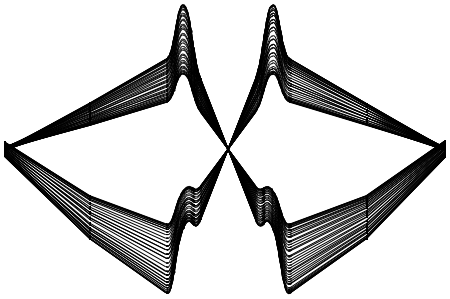
FFAG DESIGNERS

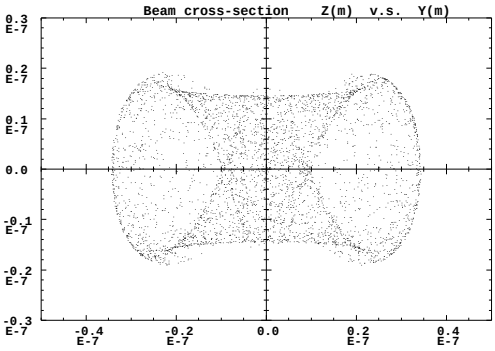
François Méot

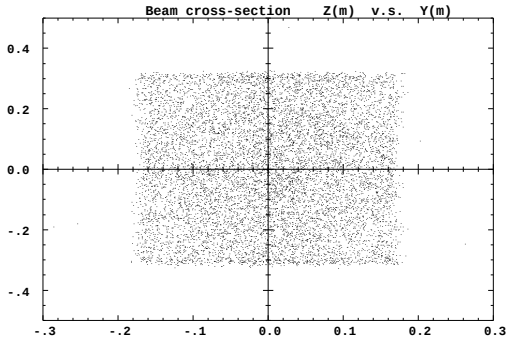
Brookhaven National Laboratory

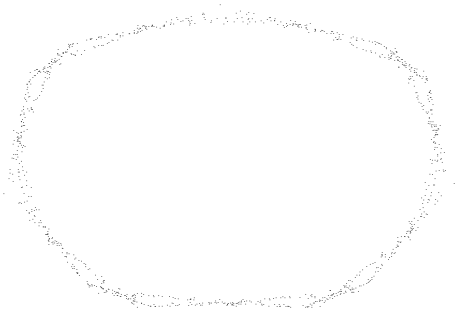
Upton, Long Island, NY

October 26, 2010









[0]**Cover figures** :
upper left : colliding proton beams in LHC interaction regions,
upper right : sub-micronic non-monochromatic beam cross-section at the image plane of a second order achromatic micro-beam line,
lower left : uniform rectangular beam cross section at the downstream end of a non-linear beam expander,
lower right : a tracking of defect limited dynamic aperture in LHC.

Table of contents

1 **INTRODUCTION (1)** **5**

2 **INTRODUCTION (2)** **8**

3 **NUMERICAL INTEGRATION** **10**

4 **NUMERICAL INTEGRATION USING ZGOUBI** **13**
4.1 How it does it : . . . 13
4.1.1 Integration in purely magnetic fields . . . 15
4.1.2 Integration in electric fields . . . 30
4.1.3 Integration in combined electric and magnetic fields . . . 38
4.1.4 Calculation of the time of flight . . . 43
4.2 How $\vec{B}$ and Derivatives Are Provided . . . 45
4.2.1 Extrapolation from 1-D axial field map . . . 45
4.2.2 Extrapolation from Median Plane Fields . . . 46
4.2.3 Extrapolation from arbitrary 2-D Field Maps . . . 51
4.2.4 Interpolation in 3-D Field Maps . . . 51
4.2.5 2-D Analytical Field Models and Extrapolation . . . 51
4.2.6 3-D Analytical Models of Fields . . . 52
4.3 Calculation of $\vec{B}$ from Field Maps . . . 59
4.3.1 1-D Axial Map, with Cylindrical Symmetry . . . 59
4.3.2 2-D Median Plane Map, with Median Plane Antisymmetry . . . 59
4.3.3 Arbitrary 2-D Map, no Symmetry . . . 63
4.3.4 Calculation of $\vec{B}$ from 3-D Field Map . . . 64
4.4 Calculation of $\vec{E}$ and its derivatives . . . 68
4.4.1 Extrapolation from 1-D axial field map . . . 68
4.4.2 1-D (Axial) Analytical E-Field Models and Extrapolation . . . 68
4.4.3 2-D Analytical E-Field Models and Extrapolation . . . 69
4.4.4 3-D Analytical models of fields . . . 69
4.5 Calculation Of $\vec{E}$ From Field Maps . . . 69

5 **SPIN TRACKING** **70**

6 **SYNCHROTRON RADIATION** **77**
6.1 Energy loss and related dynamical effects [10] . . . 77
6.2 Spectral-angular radiated densities [11] . . . 81
6.2.1 Calculation of the radiated electric field . . . 82
6.2.2 Calculation of the Fourier transform of the electric field . . . 83

7 **DESCRIPTION OF THE AVAILABLE PROCEDURES** **87**
7.1 Introduction . . . 87
7.2 Definition of an Object . . . 87
7.3 Declaration of options . . . 102
7.4 Optical Elements and related numerical procedures . . . 139
7.5 Output Procedures . . . 223

7.6 Complementary Features . . . 236
7.6.1 Backward Ray-tracing . . . 236
7.6.2 Checking Fields and Trajectories inside Optical Elements . . . 237
7.6.3 Labeling keywords . . . 238
7.6.4 Multiturn tracking in circular machines . . . 239
7.6.5 Positioning, (mis-)alignement, of optical elements and field maps 240
7.6.6 Coded integration step . . . 244
7.6.7 Ray-tracing of an arbitrarily large number of particles . . . 245
7.6.8 Stopped particles: the *LEX* flag . . . 246
7.6.9 Negative rigidity . . . 247

REFERENCES **249**

INDEX **253**

1 INTRODUCTION (1)

• We are concerned with the motion of charged particles in the electric and magnetic fields of accelerator components that ensure guiding, focusing, accelerating.

Motion of charged particles in accelerator components is determined by the Lorentz forces, electric and magnetic :

$$\vec{F} = d\vec{p}/dt = q\vec{E} + q(\vec{v} \times \vec{B})$$

• Optical elements may be

- either electric : focusing lens, multi-pole or cylindrical, accelerating cavity
- or magnetic : dipole magnet for instance, possibly with complicated fields as in FFAG structures
- or a combination of both : Wien filter for instance.

• From the point of view of particle motion, $\vec{F} = q\vec{E} + q(\vec{v} \times \vec{B})$, accelerator design relies most of the time on matrix and kick-drift transport methods :

fields E and B in F=qE+q(vxB) are approximated in a way :

$B_y(z = 0) = B_0(1n\,h\,x + n'h^2x^2 +)$ and similarly for $B_x,\ B_s$

that allows manipulating the differential equation under a simplified form

$$z'' + K_z z = f(x, y, x', y', \delta p/p) \qquad \text{(z stands for x or y)}$$

of which the linear part is treated with matrix methods, the non-linear part can be treated with kick methods, or using Hamiltonian methods.

This is how the scheme below, the optical functions in a spiral FFAG cell, have been obtained. However, by contrast with pulsed synchrotron techniques that usually rely on paraxial optics, one would not rely on these results to work out magnet parameters, for instance.

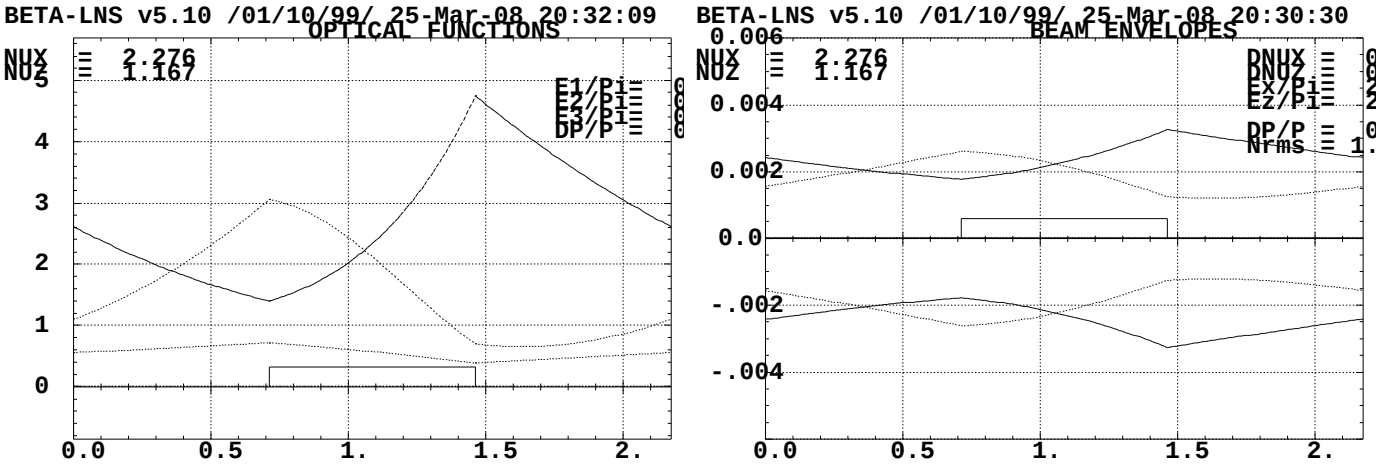


Figure 1: Optical functions in a spiral FFAG cell, obtained using matrix transport methods.

• Most unfortunately life for an accelerator designer is not always as easy, this is the case in the FFAG world :

fields are complicated and the approximations above : field modeling, matrix and/or kick-drift, are not convenient, too inaccurate.

This is where solving numerically the Lorentz forces differential equation is appropriate : this allows pratically unlimited sophistication of magnetic field modeling wherever necessary, as well as using field maps, and even time varying fields, for instance upon traversal of accelerating cavities or to account for Eddy current effects in pulsed magnets.

Note that, as long as first order motion, in “simple” magnetic fields, is concerned, the improvement brought by numerical ODE integration methods may not be so dramatic :

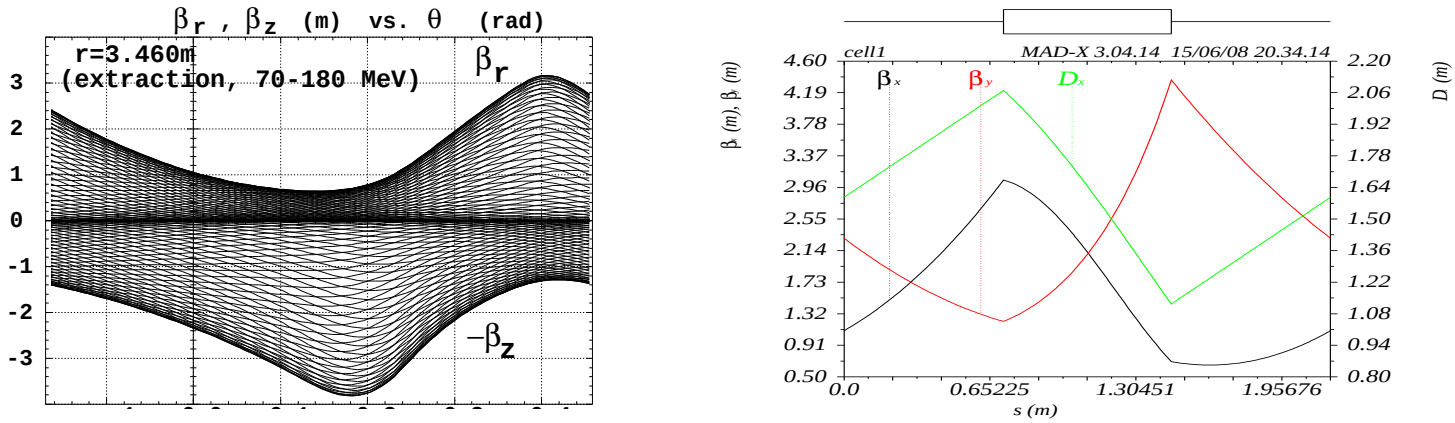


Figure 2: Optical functions in a spiral FFAG cell, obtained using numerical resolution of Lorentz equation (left) or matrix transport methods (right).

2 INTRODUCTION (2)

Leaving matrix and other methods to the benefit of ray-tracing : that's what "numerical integration of Lorentz equation" leads to, means that we will have to rely on multi-particle and multi-turn techniques so to get the optical functions as β_x, y, D_x, y, and other global quantities as envelopes $\sigma_{x,y,l}$, tunes $Q_{x,y,l}$, chromaticities, anharmonicities etc.

As an example, the transport matrix through a structure is deduced from ray coordinates :

$$Y^+ = \left(\frac{Y}{T_0}\right) T_0 + \left(\frac{Y}{T_0^2}\right) T_0^2 + \left(\frac{Y}{T_0^3}\right) T_0^3,$$
$$Y^- = -\left(\frac{Y}{T_0}\right) T_0 + \left(\frac{Y}{T_0^2}\right) T_0^2 - \left(\frac{Y}{T_0^3}\right) T_0^3$$

will yield, neglecting third order terms,

$$R_{12} = \left(\frac{Y}{T_0}\right) = \frac{Y^+ - Y^-}{2T_0}$$

Another example, the transport of beam envelopes using

$$\sigma(s_2) = T(2 \leftarrow 1)\sigma(s_1)T(2 \leftarrow 1)$$

with

$$\sigma(s) = \begin{pmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{pmatrix}$$

the beam matrix and

$$T = (s_2 \leftarrow s_1)$$

the transport matrix from s_1 to s_2, will be replaced by the calculation of the second order momenta :

$$\begin{pmatrix} \overline{x^2} & \overline{xx'} \\ \overline{xx'} & \overline{x'^2} \end{pmatrix}$$

with x(s), x'(s) the ray coordinates at path distance s.

The calculation of tunes from

$$\cos(\mu) = \text{Trace}(T)/2 \quad \text{or from} \quad \mu = \int_{turn} ds/\beta(s)$$

may be replaced by Fourrier analysis of x(n), y(n), dp/p(n).

Note that, this means that the job of turning to ray-tracing methods

- not only means developing the numerical integration tools,

- it also means developping additional tools that allow recovering, from ray coordinates, optical functions and other quantities in use in machine design.

3 NUMERICAL INTEGRATION

The Lorentz force equation

$$F = q(\vec{E}\vec{v} \times \vec{B})$$

is an ordinary differential equation : "ordinary" because it contains only one independent variable, usually time or distance, and functions of that variable, x(t), y(t), z(t) once representing it projected on the three axis x(t), y(t), z(t) of the local frame : the frame attached to the optical element on which we are bending.

Solving the Lorentz equation resorts to ODE solving methods. The basis of these methods is that of Taylor Series expansion at the point where extrapolation is required,

$$f(x_0 + h) = f(x_0) + hf'(x_0) + \frac{h^2}{2!}f''(x_0) + ...$$

$$f(x_0 + h, y_0 + k) = f(x_0, y_0) + hf'(x_0) + kf'(y_0) + \frac{h^2}{2!}\frac{\partial^2 f}{\partial x^2}(x_0, y_0)$$
$$+\frac{k^2}{2!}\frac{\partial^2 f}{\partial y^2}(x_0, y_0) + h\,k\frac{\partial^2 f}{\partial x \partial y}(x_0, y_0) + ...$$
$$...$$

ODE solving mumerical methods are based on the combination of function values at different points in the vicinity of the starting point, aiming at approximating the derivatives in the required equation.

EXAMPLES

The Euler method

The simplest method. The ODE

$$dy/dt = f(t, y)$$

is approximated by the finite difference equation

$$y_{n+1} - y_n = \Delta t \; f(t_n, y_n)$$

Given y_0 at t_0, one then gets $y_1 = y_0 + \Delta t \; f(t_0, y_0)$, $y_2 = y_1 + \Delta t \; f(t_1, y_1)$, etc.

The Runge-Kutta method

An improved Euler method, a so-called "predictor-corrector" method in which a first step *predicts* a solution ${}^{p}y_{n+1}$ which is subsequently *corrected* using the approximation $f(t_{n+1}, {}^{p}y_{n+1})$ to $f(t_{n+1}, y_{n+1})$.

The scheme of a fourth-order Runge-Kutta is the following :

$$k_1 = \Delta t f(t_n, y_n)$$

$$k_2 = \Delta t f(t_n + \Delta t/2, y_n + k_1),$$

$$k_3 = \Delta t f(t_n + \Delta t/2, y_n + k_2),$$

$$k_4 = \Delta t f(t_n + \Delta t, y_n + k_3),$$

$$y_{n+1} = y_n + (k_1 + 2k_2 + 2k_3 + k_4)/6.$$

Comparison with a Taylor Series expansion of y_{n+1} shows that the error is in Δt^5.

RK4 is a good compromise between accuracy and CPU time consumption. RK8 would be presumably more precise, at the expense of more CPU time.

The size of the step Δt determines the accuracy of the numerical integration. However, the smaller the step size, the larger the computing time.

• There are lots of other methods, more or less easy to program, providing more or less accuracy .

• We will focus on the Zgoubi method, (i) as a third illustration and (ii) since we are going to use it for exercises.

4 NUMERICAL INTEGRATION USING ZGOUBI

What zgoubi does :

Essentially, push a particle, step by step, in $\vec{B}$ or $\vec{E}$ fields.

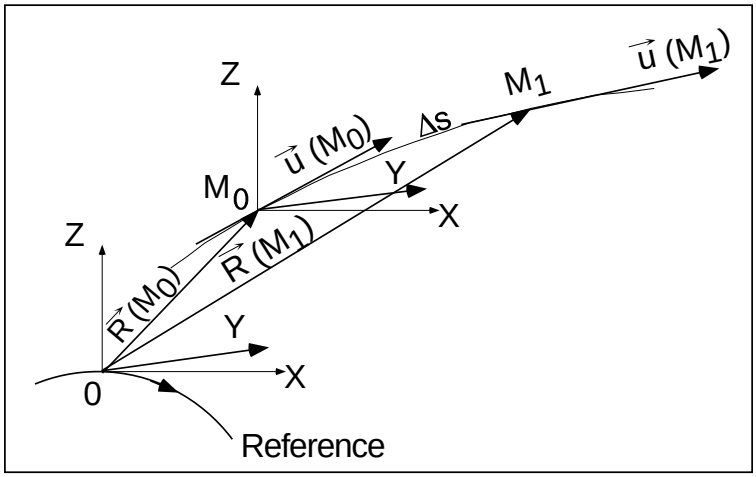


Figure 3: Position and velocity of a particle in the reference frame.

$$\text{i.e., it numerically solves the Lorentz Equation, } \frac{\mathrm{d}(\mathrm{m}\tilde{\mathrm{v}})}{\mathrm{dt}} = \mathrm{q}\,(\tilde{\mathrm{e}}+\tilde{\mathrm{v}}\times\tilde{\mathrm{b}}) \quad (4.0.1)$$

4.1 How it does it :

By using truncated Taylor expansions for position and normalized velocitiy $\vec{u} = \vec{v}/v$:

$$\begin{aligned} \vec{R}(M_1) &\approx \vec{R}(M_0) + \vec{u}(M_0)\,\Delta s + \vec{u}'(M_0)\,\frac{\Delta s^2}{2!} + \ldots + \vec{u}''''(M_0)\,\frac{\Delta s^6}{6!} \\ \vec{u}(M_1) &\approx \vec{u}(M_0) + \vec{u}'(M_0)\,\Delta s + \vec{u}''(M_0)\,\frac{\Delta s^2}{2!} + \ldots + \vec{u}''''(M_0)\,\frac{\Delta s^5}{5!} \end{aligned} \quad (4.1.1)$$

wherein $\boxed{()' = d()/ds}$.

• In case $\vec{E}$ is non-zero, the rigidity at M_1 needs be re-computed as well :

$$(B\rho)(M_1) \approx (B\rho)(M_0) + (B\rho)'(M_0)\Delta s + \ldots + (B\rho)''''(M_0)\frac{\Delta s^4}{4!} \quad (4.1.2)$$

• In case time-of-flight is desired (not always the case !), it is computed in a similar manner :

$$T(M_1) \approx T(M_0) + \frac{dT}{ds}(M_0)\,\Delta s + \frac{d^2T}{ds^2}(M_0)\,\frac{\Delta s^2}{2} + \frac{d^3T}{ds^3}(M_0)\,\frac{\Delta s^3}{3!} + \frac{d^4T}{ds^4}(M_0)\,\frac{\Delta s^4}{4!} \quad (4.1.3)$$

• The truncated Taylor series

$$\begin{aligned}
\vec{R}(M_1) &= \vec{R}(M_0) + \vec{u}(M_0)\,\Delta s + ... \\
\vec{u}(M_1) &= \vec{u}(M_0) + \vec{u}'(M_0)\,\Delta s + ... \\
(B\rho)(M_1) &= (B\rho)(M_0) + (B\rho)'(M_0)\Delta s + ... \\
T(M_1) &= T(M_0) + \frac{dT}{ds}(M_0)\,\Delta s + ...
\end{aligned} \tag{4.1.4}$$

require computation of the derivatives

$$\vec{u}^{(n)} = \frac{d^n\vec{u}}{ds^n}$$

$$(B\rho)^{(n)} = \frac{d^n(B\rho)}{ds^n}$$

$$\frac{d^n(T)}{ds^n}$$

• Let's introduce simplified notations. We will from now on use :

$$\vec{u} = \frac{\vec{v}}{v}, \quad ds = v\,dt, \quad \vec{u}' = \frac{d\vec{u}}{ds}, \quad m\vec{v} = mv\vec{u} = q\,B\rho\,\vec{u} \tag{4.1.5}$$

so that the Lorentz equation can be re-written under the more convenient form (at least more convenient to handle in the code) :

$$\boxed{(B\rho)'\vec{u} + B\rho\,\vec{u}' = \frac{\vec{e}}{v} + \vec{u} \times \vec{b}}$$

4.1.1 Integration in purely magnetic fields

We now assume that $\vec{e} = 0$, and we note $\vec{B} = \dfrac{\vec{b}}{B\rho}$, so that we get the Lorentz equation under the simplified notation

$$\boxed{\vec{u}' = \vec{u} \times \vec{B}}$$

The successive derivatives $\vec{u}^{(n)} = \dfrac{d^n \vec{u}}{ds^n}$ of $\vec{u}$ needed in the Taylor expansions (eqs. 4.1.1) are calculated by differentiating $\vec{u}' = \vec{u} \times \vec{B}$

$$\begin{aligned}
\vec{u}'' &= \vec{u}' \times \vec{B} + \vec{u} \times \vec{B}' \\
\vec{u}''' &= \vec{u}'' \times \vec{B} + 2\vec{u}' \times \vec{B}' + \vec{u} \times \vec{B}'' \\
\vec{u}'''' &= \vec{u}''' \times \vec{B} + 3\vec{u}'' \times \vec{B}' + 3\vec{u}' \times \vec{B}'' + \vec{u} \times \vec{B}''' \\
\vec{u}''''' &= \vec{u}'''' \times \vec{B} + 4\vec{u}''' \times \vec{B}' + 6\vec{u}'' \times \vec{B}'' + 4\vec{u}' \times \vec{B}''' + \vec{u} \times \vec{B}''''
\end{aligned} \tag{4.1.6}$$

where $\vec{B}^{(n)} = \dfrac{d^n \vec{B}}{ds^n}$.

- Computation of these series takes place in the routine “devtra”

Edit devtra, take a look into it...

Looking into subroutine devtra.f for Eq. 4.1.7 series

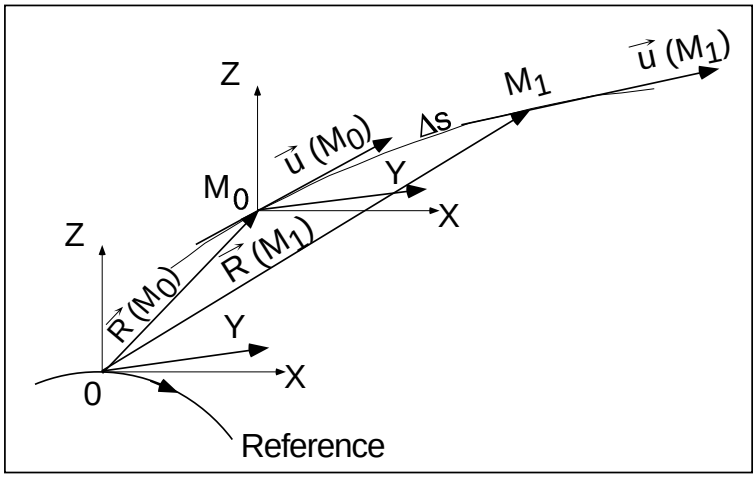
```
      SUBROUTINE DEVTRA
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C     ----------------------------------------
C     CALCULE u ET SES DERIVEES dnu/dsn, n=1,5,
C     i.e., calcul des coeffs des series de Taylor
C     (eqs. 2.2.4-5)
C     ----------------------------------------
      COMMON/CDF/ IES,LF,LST,NDAT,NRES,NPLT,NFAI,NMAP,NSPN,NLOG
      COMMON/CHAVE/ B(5,3),V(5,3),E(5,3)
      COMMON/D3BXYZ/ D3BX(27), D3BY(27), D3BZ(27)
      COMMON/D4BXYZ/ D4BX(81),D4BY(81),D4BZ(81)
      COMMON/DDEXYZ/ DE(9),DDE(27)
      COMMON/D3EXYZ/ D3EX(27), D3EY(27), D3EZ(27)
      COMMON/D4EXYZ/ D4EX(81), D4EY(81), D4EZ(81)
      COMMON/DDBXYZ/ DB(9),DDB(27)
      COMMON/MARK/ KART,KALC,KERK,KUASEX
      LOGICAL ZSYM
      COMMON/OPTION/ KFLD,MG,LC,ML,ZSYM
      COMMON/ORDRES/ KORD,IRD,IDS,IDB,IDE,IDZ
C      COMMON/RIGID/ BORO,DPREF,DPP,BR
      COMMON/RIGID/ BORO,DPREF,DP,BR
      COMMON/TRAJ/ R1,T1,Z1,P1,A1,SAR,TAR,KEX,IT,AMT,QT
      COMMON/VITES/ U(6,3),DBR(6),DDT(6)
      DIMENSION EU(5)
      PARAMETER(IMAX=3, IJMAX=9, IJLMAX=27, IM2 = IMAX+IJMAX)
      PARAMETER(IM3 = IM2+IJLMAX)
      SAVE CL, CL9

C-------------------------------------------------------------
C     CALCUL D'ADRESSE : ADRESSE(I,J) = I + (J-1)*IMAX
C     EXEMPLES:
C      DB(2,1) = DB(2)
C      DB(3,3) = DB(9)
C     CALCUL D'ADRESSE : ADRESSE(I,J,K) = I+(J-1)*IMAX+(K-1)*IMAX*JMAX
C     EXEMPLES:
C      DDB(3,1,1) = DDB(3)
C      DDB(3,3,1) = DDB(3*3**0+2*3**1+0*3**2)=DDB(9)
C-------------------------------------------------------------

CALCUL u=DX/DS
      CP=COS(P1)
      U(1,1)=CP*COS(T1)
      U(1,2)=CP*SIN(T1)
      U(1,3)=SIN(P1)

      GOTO(1,2,3) KFLD

C----- MAGNETIC FIELD
 1    CONTINUE

CALCUL u'=uxB
      CALL PVECT(1,1,1)
      DO 10 K=1,3
 10     U(2,K)=V(1,K)

      IF(IDS.LE.2) GOTO 99

C      B'=dB/dS
      KIM = -IMAX
      DO 12 K=1,3
        TP=0.D0
        KIM = KIM + IMAX
        IF(IDB.GE.1) THEN
          DO 11 I=1,3
            IK = I + KIM
            TP=TP+U(1,I)*DB(IK)
 11       CONTINUE
        ENDIF
        B(2,K)=TP
 12   CONTINUE

CALCUL u''=u'xB + uxB'
      CALL PVECT(1,1,2)
      CALL PVECT(2,2,1)
      DO 13 K=1,3
 13     U(3,K)=V(1,K)+V(2,K)

      IF(IDS.LE.3) GOTO 99

CALCUL B''=d2B/ds2
      KIJM = -IJMAX-IMAX
      KIM = -IMAX
      DO 15 K=1,3
        TP=0.D0
        IF(IDB.GE.1) THEN
          KIJM = KIJM + IJMAX
          KIM = KIM + IMAX
          DO 14 I=1,3
            IK = I + KIM
            TP=TP+U(2,I)*DB(IK)
C
C           symetrie
C
            IF(IDB.GE.2) THEN
              IKIJ = I + KIJM
              DO 141 J=I,3
                IF(I .EQ. J) THEN
                  XMUL=1.0D0
                ELSE
                  XMUL=2.0D0
                ENDIF
                IJK = IKIJ + J*IMAX
                TP=TP+XMUL*U(1,I)*U(1,J)*DDB(
 141          CONTINUE
            ENDIF
 14       CONTINUE
        ENDIF
        B(3,K)=TP
 15   CONTINUE

CALCUL u'''=u''xB + 2*u'B' + uxB''
      CALL PVECT(1,1,3)
      CALL PVECT(2,2,2)
      CALL PVECT(3,3,1)
      DO 16 K=1,3
 16     U(4,K)=V(1,K)+2.D0*V(2,K)+V(3,K)

.........
```

Integration in purely magnetic fields (cont'd)

• It is clear from

$$\vec{u}'' = \vec{u}' \times \vec{B} + \vec{u} \times \vec{B}'$$
$$\vec{u}''' = \vec{u}'' \times \vec{B} + 2\vec{u}' \times \vec{B}' + \vec{u} \times \vec{B}'' \qquad (4.1.7)$$
$$etc.$$

that one needs to compute the magnetic field and - unlike Runge-Kutta method - its derivatives wrt. path variable s ...



• This is also done in devtra.f.
devtra.f computes the field derivatives wrt. path length,

$$\vec{B}^{(n)} = \frac{d^n \vec{B}}{ds^n}$$

from the derivatives of $\vec{B}$ in the reference frame of the magnet, using the receipe

$$d\vec{B} = \frac{\partial \vec{B}}{\partial X} dX + \frac{\partial \vec{B}}{\partial Y} dY + \frac{\partial \vec{B}}{\partial Z} dZ = \sum_{i=1,3} \frac{\partial \vec{B}}{\partial X_i} dX_i,$$

which, by successive differentiations, yields :

$$\vec{B}' = \sum_i \frac{\partial \vec{B}}{\partial X_i} u_i$$

$$\vec{B}'' = \sum_{ij} \frac{\partial^2 \vec{B}}{\partial X_i \partial X_j} u_i u_j + \sum_i \frac{\partial \vec{B}}{\partial X_i} u_i' \qquad (4.1.8)$$

$$\vec{B}''' = \sum_{ijk} \frac{\partial^3 \vec{B}}{\partial X_i \partial X_j \partial X_k} u_i u_j u_k + 3 \sum_{ij} \frac{\partial^2 \vec{B}}{\partial X_i \partial X_j} u_i' u_j + \sum_i \frac{\partial \vec{B}}{\partial X_i} u_i''$$

etc.

See routine devtra :

```
      SUBROUTINE DEVTRA
...................
CALCUL u'=uxB
      CALL PVECT(1,1,1)
      DO 10 K=1,3
 10     U(2,K)=V(1,K)

      IF(IDS.LE.2) GOTO 99

C      B'=dB/dS
      KIM = -IMAX
      DO 12 K=1,3
        TP=0.D0
        KIM = KIM + IMAX
        IF(IDB.GE.1) THEN
          DO 11 I=1,3
            IK = I + KIM
            TP=TP+U(1,I)*DB(IK)
 11       CONTINUE
        ENDIF
        B(2,K)=TP
 12   CONTINUE
......................
CALCUL u''=u'xB + uxB'
      CALL PVECT(1,1,2)
      CALL PVECT(2,2,1)
      DO 13 K=1,3
 13     U(3,K)=V(1,K)+V(2,K)

      IF(IDS.LE.3) GOTO 99

CALCUL B''=d2B/ds2
      KIJM = -IJMAX-IMAX
      KIM = -IMAX
      DO 15 K=1,3
        TP=0.D0
        IF(IDB.GE.1) THEN
          KIJM = KIJM + IJMAX
          KIM = KIM + IMAX
          DO 14 I=1,3
            IK = I + KIM
            TP=TP+U(2,I)*DB(IK)
C
C           symetrie
C
            IF(IDB.GE.2) THEN
              IKIJ = I + KIJM
              DO 141 J=I,3
                IF(I .EQ. J) THEN
                  XMUL=1.0D0
                ELSE
                  XMUL=2.0D0
                ENDIF
                IJK = IKIJ + J*IMAX
                TP=TP+XMUL*U(1,I)*U(1,J)*DDB(IJK)
 141          CONTINUE
            ENDIF
 14       CONTINUE
        ENDIF
        B(3,K)=TP
 15   CONTINUE
.................................
```

Let's see where the $\partial^{l+m+n}B/\partial^l X_i \partial^m X_j \partial^n X_k$ i.e., the DB, DDB, B3Bxyz.. are computed

- An example : the optical element 'MULTIPOL' which we are going to use soon. Edit multip.f.

```
      SUBROUTINE MULTIP(ID,KFL,KUASEX,X,Y0,Z0,BM,DLEM,DLSM,DFM,DE,DS,RT
     >,XE,XS,CE,CS
     >,B,DB,DDB,D3BX,D3BY,D3BZ,D4BX,D4BY,D4BZ,BT)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(MPOL=10)
      DIMENSION BM(MPOL),DLEM(MPOL),DLSM(MPOL),DFM(MPOL)
     >,DE(MPOL,6),DS(MPOL,6),RT(MPOL)
      PARAMETER(MCOEF=6)
      DIMENSION CE(MCOEF),CS(MCOEF)
      DIMENSION B(5,3),DB(3,3),DDB(3,3,3)
      DIMENSION D3BX(3,3,3), D3BY(3,3,3), D3BZ(3,3,3)
      DIMENSION D4BX(3,3,3,3) ,D4BY(3,3,3,3) ,D4BZ(3,3,3,3)
      DIMENSION BT(5, *)

      COMMON/AIM/ BO,RO,FG,GF,XI,XF,EN,EB1,EB2,EG1,EG2
      COMMON/CDF/ IES,LF,LST,NDAT,NRES,NPLT,NFAI,NMAP,NSPN,NLOG
      INCLUDE "MAXTRA.H"
      COMMON/CHAMBR/ LIMIT,IFORM,YLIM2,ZLIM2,SORT(MXT),FMAG,BMAX
     > ,YCH,ZCH
      COMMON/CONST2/ ZERO, UN
      COMMON/INTEG/ PAS,DXI,XLIM,XCE,YCE,ALE,XCS,YCS,ALS,KP
      COMMON/RIGID/ BORO,DPREF,DP,BR

      LOGICAL CHFE, CHFS, CHU
      PARAMETER (Q7=3.038194444D-4)

      LOGICAL CASPI, LTEMP
      SAVE CASPI

      XLS=XLIM-XS
      IOP=1

      GOTO  (1,2,3,4,5,6,7,8,9,10,11), KUASEX
      CALL ENDJOB('*** Error, SBR MULTIP -> KUASEX value',-99)

 11   CONTINUE
      IF( BM(1) .EQ. ZERO) GOTO 82

C----- DIPOLE
 1    CONTINUE

      IP=1
      GN = BM(IP)/BR
      DLE = DLEM(IP)
      DLS = DLSM(IP)
      Y=Y0
      Z=Z0
      IF(RT(IP) .NE. ZERO) CALL ROTX(RT(IP),Y,Z)
      IF(DLE+DLS .EQ. ZERO) THEN
C------- DIPOLE CRENEAU
        IF ( X.GE.XE .AND. X.LE.XS ) THEN
          B(1,3)  = GN
        ENDIF
      ELSEIF(DLE+DLS .NE. ZERO) THEN
.......................
        IF(CHU) THEN
C         *** POINT DANS ZONE DE Champ Uniforme ( G(X)=0 )
            B(1,3)  = GN
        ELSE
C       *** POINT DANS ZONE DE Champ DE FUITE D'ENTRE ET/OU DE SORTIE

          IF(.NOT. CHFE) THEN
C           *** HORS D'EFFET DU Champ DE FUITE D'ENTREE
C             OU GRADIENT CRENEAU
.....................
        ELSE
C           *** EFFET DU Champ DE FUITE D'ENTREE

          SE=SE/DLE
          CALL DRVG(4,CE,SE,GE,DGE,D2GE,D3GE,D4GE,D5GE,D6GE)
C           CALL DRVG(6,CE,SE,GE,DGE,D2GE,D3GE,D4GE,D5GE,D6GE)

        ENDIF

        IF(.NOT. CHFS) THEN
C         *** HORS D'EFFET DU Champ DE FUITE DE SORTIE
C           OU GRADIENT CRENEAU
.....................
          SS=SS/DLS
          CALL DRVG(4,CS,SS,GS,DGS,D2GS,D3GS,D4GS,D5GS,D6GS)
C           CALL DRVG(6,CS,SS,GS,DGS,D2GS,D3GS,D4GS,D5GS,D6GS)

        ENDIF

        G= GE+GS-UN

        IF(G.LT.ZERO) CALL ENDJOB(
     >    'SBR CHAMC : gradient is wrong,  G (=GE+GS-1) < ',0)

.....................................;
C--------- Components Bx, By, Bz of field
        B(1,1)= Z*(DG + (- D3G/8.D0 + D5G/192.D0*YPZ)*YPZ)
        B(1,2)=  YZ*(-D2G/4.D0 + (D4G/48.D0  - D6G/1536.D0*YPZ
        B(1,3)=   G - D2G/8.D0*Y2P3Z2 + D4G/192.D0*Y4YZZ4
     >   - D6G/9216.D0*(Y6+9.D0*Y4*Z2+15.D0*Y2*Z4+7.D0*Z6)
C       ... dBx/dX
        DB(1,1) = Z*(D2G+( -D4G/8.D0+D6G/192.D0*YPZ )*YPZ)
C       ... dBx/dY = dBy/dX
        DB(2,1) = YZ*(-.25D0*D3G + D5G/48.D0*YPZ)
C       ... dBy/dY
        DB(2,2) = Z*(-.25D0*D2G + D4G/48.D0*(3.D0*Y2+Z2)
     >    - D6G/1536.D0*(5.D0*Y4+6.D0*Y2Z2+Z4))
C       .. dBx/dZ = dBz/dX
        DB(3,1) = DG - D3G/8.D0*Y2P3Z2 + D5G/192.D0*Y4YZZ4
C       .. dBy/dZ = dBZ/dY
        DB(3,2) = Y*(-.25D0*D2G + D4G/48.D0*Y2P3Z2-D6G/1536.D0

C       ... d2Bx/dX2
        DDB(1,1,1) = Z*(D3G - D5G/8.D0*YPZ)
C       ... d2Bx/dXdY = d2By/dX2
        DDB(2,1,1)= YZ*(-.25D0*D4G + D6G/48.D0*YPZ)
C       ... d2Bx/dY2
        DDB(2,2,1) = Z*(-.25D0*D3G + D5G/48.D0*(3.D0*Y2+Z2))
C       ... d2By/dY2
        DDB(2,2,2) = YZ*(D4G/8.D0 - D6G/384.D0*(5.D0*Y2+3.D0*Z
C        DDB(2,2,2) = YZ*(D4G/8.D0 - D6G/64.D0*(5.D0*Y2+3.D0*Z
C       .. d2Bx/dXdZ = d2Bz/dX2
        DDB(3,1,1) = D2G-D4G/8.D0*Y2P3Z2+D6G/192.D0*Y4YZZ4
C       .. d2By/dXdZ = d2Bz/dXdY = d2Bx/dYdZ
        DDB(3,2,1) = Y*(-.25D0*D3G + D5G/48.D0*Y2P3Z2)
C       .. d2By/dYdZ = d2Bz/dY2
        DDB(3,2,2) = -.25D0*D2G+D4G/16.D0*YPZ-D6G/1536.D0*
     >    (5.D0*Y4+18.D0*Y2Z2+5.D0*Z4)
          IF(ID .GE. 3) THEN
C---------- THIRD AND FOURTH ORDER NOT INSTALLED
          ENDIF
C         ... ENDIF ID>=3
        ENDIF
C         endif CHU
      ENDIF
C     **  endif test DLE+DLS
 999  CONTINUE
      IF(RT(IP) .NE. ZERO)
     >  CALL XROTB(RT(IP),B,DB,DDB,D3BX,D3BY,D3BZ,D4BX,D4BY,D4BZ
      IF(KUASEX .EQ. IP) RETURN
C-----  END DIP
      CALL ADPOL(ID,IOP,KFL,B,DB,DDB,D3BX,D3BY,D3BZ,D4BX,D4BY,D4
 82   CONTINUE
      IF( BM(2) .EQ. ZERO) GOTO 83
C-------------------------------------------------------------
C----- QUADRUPOLE
 2    CONTINUE
```

There is a lot of magnetic elements available in zgoubi (Guide, pp 7-9, 139-141) :

MAGNETIC ELEMENTS

Decapole	DECAPOLE, MULTIPOL
Dipole	AIMANT, BEND, DIPOLE, DIPOLE-M, MULTIPOL, QUADISEX
Dodecapole	DODECAPO, MULTIPOL
FFAG magnets	DIPOLES, FFAG, FFAG-SPI, MULTIPOL
Multipole	MULTIPOL, QUADISEX, SEXQUAD
Octupole	OCTUPOLE, MULTIPOL, QUADISEX, SEXQUAD
Quadrupole	QUADRUPO, MULTIPOL, SEXQUAD
Sextupole	SEXTUPOL, MULTIPOL, QUADISEX, SEXQUAD
Skewed multipoles	MULTIPOL
Solenoid	SOLENOID
Undulator	UNDULATOR

Without forgetting magnetic field maps

1-D, cylindrical symmetry	BREVOL
2-D, mid-plane symmetry	CARTEMES, POISSON, TOSCA
2-D, no symmetry	MAP2D
2-D, polar mesh, mid-plane symmetry	POLARMES
3-D, no symmetry	TOSCA

EXERCISE #1 - Now, Let's dare an exercise, let's run a file and do some tracking

• Create a directory, name it exercise1. Go in.

• Write down the following data list, save it in "zgoubi.dat", it will be our input to zgoubi :

```
Part of EMMA cell
'OBJET'                          ! A 'keyword'. It generates initial particle coordinates
+5.171103865921708e+01           ! Rigidity (kG.cm) : 15 MeV electrons
2
2 1
1.03189 13.67 0.695 0.0 0.0 +1.19341e-00 'o'      ! y, y', z, z', dl, p/p_0 ->18.0MeV
0.714517 13.59 0.234 0.0 0.0 1.19341e-00 'o'
1 1
'FAISCEAU'                       ! Another 'keyword'. Prints local particle coordinates
'FAISTORE'                       ! Another 'keyword'. Causes storage of local particle coordinates
zgoubi.fai
1
'MARKER'    BEG   dum .plt       ! Just a marker, does nothing
'DRIFT'     DD1
10
 'CHANGREF'                      ! Allows positionning of the magnet
  0 -0.1111304250932937 1.47531616464
  'MULTIPOL'  QF                 ! A multipole, whose sole B_dipole and B_quad are non-zero
  2
  6 2.208947037893755 -5.072134730870179e-01 1.6409348483 0 0 0 0 0 0 0 0
   2. 1.1  1.00 1.00 1.00 1.00 1.00 1. 1. 1. 1.
  4  .1455   2.2670  -.6395  1.1558  0. 0.  0.
   2. 1.1  1.00 1.00 1.00 1.00 1.00 1. 1. 1. 1.
  4  .1455   2.2670  -.6395  1.1558  0. 0.  0.
  0 0 0 0 0 0 0 0 0 0
  #160|160|160                   ! step sizes, in entrance FF / body / exit FF regions
  1 0 0 0
 'CHANGREF'                      ! Allows positionning of the magnet
  0 0.1111304250932937 1.47531616464
'DRIFT'     DQ
3.9
'FAISCEAU'
'END'
```

• We do not want to lose all that, we will use it again in near future... so save "zgoubi.dat" to "exercise1.dat".

• run zgoubi

• looking into "zgoubi.res" will tell you what happened

EXERCISE #1, cont'd - Let's do some plotting, from zgoubi.plt

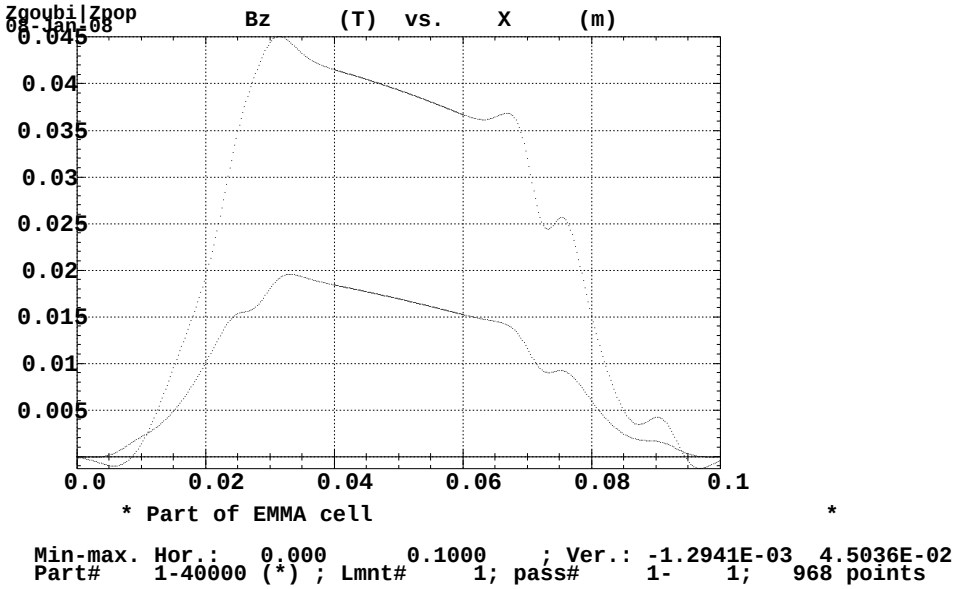
• Note the presence of a '2' (IC=2, following the users' guide nomenclture) right underneath 'MULTIPOL'

This tells **zgoubi** to save particle data, step after step while stepwise ray-tracing, into zgoubi.plt.

• Using zpop you can plot the content of "zgoubi.plt"

• Let's plot the field along trajectories :

- run zpop
- then 7/1/1 to open zgoubi.plt
- then [7/]2/(8,32) to chose he variables to be plotted
- then [7/]7 to draw
- then [7/]8 to save



B_Z (i.e. vertical) component of the magnetic field along trajectories, step by step, across 'MULTIPOL'.

• **This is a fundamental thing to do, prior to going any further :** make sure the field experienced by particles looks (reasonably) physical. Or, even better, exactly what you were expecting from the analytical calculations you did the night before

• **There is a second fundamental thing to do, prior to going any further :**

make sure the numerical integration does converge, i.e. :

- integration step is small enough. Not too small yet because you do not want (i) to spend your life getting a particle transported, (ii) to reach computer truncature limit !!
- integration extends far enough in field fall-off regions that field has become negligible

A few words concerning [binary] storage in "[b_]zgoubi.plt"

Printing into zgoubi.plt is done by impplt.f.

• Edit impplt.f. Look into it :

```
      SUBROUTINE IMPPLT(LN,DS,Y,T,Z,P,X,SAR,TAR,KEX,IT,AMT,QT)
.................
      LOGICAL BINARY,BINARI
      SAVE BINARY
      CALL INTEG4(NSTEP)
      K = LSTK()
C----- Case  IL=2*10**n with n>1 (IL=20, 200, 2000...)
      IF(1+K*((NSTEP-1)/K) .NE. NSTEP) RETURN
      ENTRY IMPPLA(LN,DS,Y,T,Z,P,X,SAR,TAR,KEX,IT,AMT,QT)
C------- Bx,y,z (kG)
      BX=B(1,1)  *BR
      BY=B(1,2)  *BR
      BZ=B(1,3)  *BR
      EX=E(1,1)  *BR
      EY=E(1,2)  *BR
      EZ=E(1,3)  *BR

C  Assign XXX
C      XXX=BR/BORO
      XXX= -1.D0 + BR/BORO
      XXXO=-1.D0 + FO(1,IT)
C-----------------------------------------------
      DY = 0.D0
CCCC test spiral injection      IF(KART.EQ.2) DY = RM
      GOTO 10

      ENTRY IMPPLB(LN,DS,Y,T,Z,P,X,SAR,TAR,KEX,IT,AMT,QT)

      BX=0.D0
      BY=0.D0
      BZ=0.D0
      EX=0.D0
      EY=0.D0
      EZ=0.D0
      XXX =-1.D0 + F(1,IT)
      XXXO=-1.D0 + FO(1,IT)
      DY = 0.D0

 10   CONTINUE
      IF(BINARY) THEN
        WRITE(LN)
     1   LET(IT),KEX,      XXXO,(FO(J,IT),J=2,MXJ),
     2   XXX,Y-DY,T*1.D3,
     3   Z,P*1.D3,SAR,     TAR,     DY,
C     3   Z,P*1.D3,SAR,     TAR,     DS,
     4   KART, IT,IREP(IT),SORT(IT),X, BX,BY,BZ, RET(IT), DPR(IT),
     5   EX,EY,EZ,  BORO,  IPASS, KLEY,(LABEL(NOEL,I),I=1,2),NOEL
      ELSE
        WRITE(LN,100)
     1   LET(IT),KEX,      XXXO,(FO(J,IT),J=2,MXJ),
C       Initial coordinates: D,  Y,T,Z,P,X,Time
     2   XXX,Y-DY,T*1.D3,
     3   Z,P*1.D3,SAR,     TAR,     DY,
C     3   Z,P*1.D3,SAR,     TAR,     DS,
C     current coordinates  time    Step
C                          mu_s    size
     4   KART,  IT,IREP(IT),SORT(IT),X, BX,BY,BZ, RET(IT), DPR(IT),
C        Cart.              Path out     - kG -     (S)   dp/p_Synchro
C         or                 CHAMBR                Synchrotron
C        Polar                                      motion
     5   EX,EY,EZ,   BORO,  IPASS, KLEY,(LABEL(NOEL,I),I=1,2),NOEL
C        - eV/cm -  reference
C                rigidity(kG.cm)
        INCLUDE "FRMPLT.H"
      ENDIF
      RETURN
      ENTRY IMPPL2(BINARI)
      BINARY=BINARI
      RETURN
      END
```

• Saving in binary format is possible.

For that, what is needed is that the storage file's name, as specified in the FILPLT.H include file, starts with 'b_' or 'B_' :

```
C Name of storage file
C normally zgoubi.plt ;
C b_zgoubi.plt for binary storage
      CHARACTER*20 FILPLT
      PARAMETER (FILPLT='zgoubi.plt')
C      PARAMETER (FILPLT='b_zgoubi.plt')
```

The interest is that I/Os are much faster.

• Let's do an EXERCISE on that

- run zgoubi.dat using step sizes $\#90000|90000|90000$.

Note that we have a problem there...

The key is in 'MXSTEP.H' - look inside. This can be changed, we won't for the moment. Needs re-compiling. Instead :

- change to step sizes $\#30000|30000|30000$, hence total $< 10^4$, this will cure the problem.

Run zgoubi, CPU is ≈12s.

- change to b_zgoubi.plt in FILPLT, re-compile zgoubi.

Run zgoubi again, CPU is down to ≈3s.

Note :

(i) same with zpop, CPU needed for plotting differs by $\approx \times 4$

(ii) same with field maps, if you are going to track through fields maps, better use binary files. We'll come back to that later on.

(ii) same with coordinates storage using $[b_]$zgoubi.fai.

• Now, for the rest of the tutorial, get back to the 'zgoubi.plt' notation.

Also, if we are no longer interested in the step-wise infos, it is better to replace IC=2 by IC=0, that switches off the saving into zgoubi.plt - which might get very voluminous, and that saves CPU time.

EXERCISE #1, cont'd

• Let's take a look to the content of "zgoubi.fai", and check that what we find therein is in agreement with what 'FAISCEAU' produces in zgoubi.res.

• You'll notice there is more in zgoubi.fai than we get from 'FAISCEAU'.
Higher precision output format also.

• impfai does the writing into [b_]zgoubi.fai. Edit it, let's take a look inside :

```
      DO 1 I=1,IMAX
           P = BORO*CL9 *F(1,I) * AMQ(2,I)
           ENERG = SQRT(P*P + AMQ(1,I)*AMQ(1,I))
           ENEKI = ENERG - AMQ(1,I)
           WRITE(NFAI,110)
     1     LET(I),IEX(I),-1.D0+FO(1,I),(FO(J,I),J=2,MXJ),
     2     -1.D0+F(1,I),F(2,I),F(3,I),
     3     (F(J,I),J=4,MXJ),ENEKI,
     4     I,IREP(I), SORT(I),(AMQ(J,I),J=1,5),RET(I),DPR(I),
     5                          BORO, IPASS, KLEY,LBL1,LBL2,NOEL
        INCLUDE "FRMFAI.H"
 1     CONTINUE
```

The interest of printing into zgoubi.fai is for plotting, mainly. For instance for plotting local phase space when doing multiturn tracking.

Let's do it...

EXERCISE #2 - Now, we track a large number a of passes through EMMA cell

• Starting from Ex. 1 data file,

- we add the second quadrupole with its positionning ('CHANGREF') and drift :

```
  'CHANGREF'
  0 1.707551921352993 -5.7610304504
  'MULTIPOL'  QD
  0
  7 3.294530758285338 7.485217218484019e-01 -1.52191811761 0 0 0 0 0 0 0 0
   2. 1.1  1.00 1.00 1.00 1.00 1.00 1. 1. 1. 1.
  4  .1455   2.2670  -.6395  1.1558  0. 0.  0.
   2. 1.1  1.00 1.00 1.00 1.00 1.00 1. 1. 1. 1.
 4  .1455   2.2670  -.6395  1.1558  0. 0.  0.
  0 0 0 0 0 0 0 0 0 0
  #60|60|60   step
  1 0 0 0
  'CHANGREF'                ! this positions  'MULTIPOL'
  0 -1.707551921352993 -5.7610304504
'DRIFT'     DD2
9.9
'FAISCEAU'
'REBELOTE'                  ! means 'do it again !', 19000 times, starting
19000 0.2 99                ! with final coordiantes (that's the meaning of '99')
'END'
```

- we make sure 'IC=0' in both quads,

- we add 'REBELOTE' so to get the multiturn, say

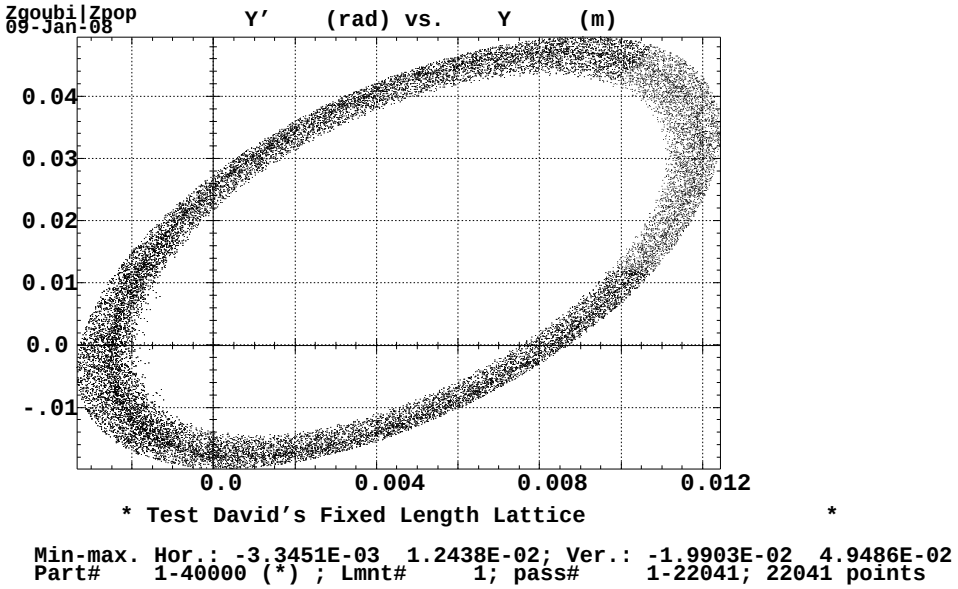
- we now use b_zgoubi.fai rather than zgoubi.fai, under keyword 'FAISTORE', to save with binary format (this is faster, to write - by zgoubi - and to read - e.g. using zpop - and yields more compact file, $\approx 1/4$).

- we get rid of 'PARTICUL', useless

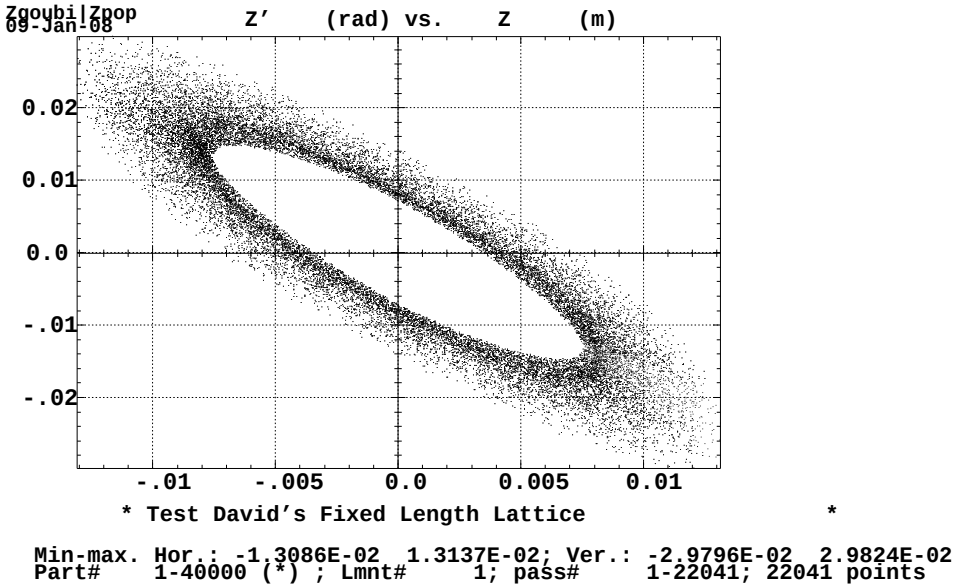
- This all yields :

```
19001 passes in EMMA cell
'OBJET'
+5.171103865921708e+01     ! 15MeV
2
1 1
1.03189 13.67 0.695 0.0 0.0 +1.19341e-00 "o" 18.0MeV
1
'FAISTORE'                 ! Another 'keyword'. Causes storage of local particle
b_zgoubi.fai
1
'MARKER'    BEG   dum .plt ! Just a marker, does nothing
'DRIFT'     DD1
10
 'CHANGREF'                ! Allows positionning of the magnet
  0 -0.1111304250932937 1.47531616464
  'MULTIPOL'  QF           ! A multipole, whose sole B_dipole and B_quad are non
  0
  6 2.208947037893755 -5.072134730870179e-01 1.6409348483 0 0 0 0 0 0 0 0
   2. 1.1  1.00 1.00 1.00 1.00 1.00 1. 1. 1. 1.
  4  .1455   2.2670  -.6395  1.1558  0. 0.  0.
   2. 1.1  1.00 1.00 1.00 1.00 1.00 1. 1. 1. 1.
  4  .1455   2.2670  -.6395  1.1558  0. 0.  0.
  0 0 0 0 0 0 0 0 0 0
  #60|60|60                ! step sizes, in entrance FF / body / exit FF regions
  1 0 0 0
 'CHANGREF'                ! allows positionning of the magnet
  0 0.1111304250932937 1.47531616464
'DRIFT'     DQ
3.9                        ! this positions  'MULTIPOL'
  'CHANGREF'
  0 1.707551921352993 -5.7610304504
  'MULTIPOL'  QD
  0
  7 3.294530758285338 7.485217218484019e-01 -1.52191811761 0 0 0 0 0 0 0 0
   2. 1.1  1.00 1.00 1.00 1.00 1.00 1. 1. 1. 1.
  4  .1455   2.2670  -.6395  1.1558  0. 0.  0.
   2. 1.1  1.00 1.00 1.00 1.00 1.00 1. 1. 1. 1.
 4  .1455   2.2670  -.6395  1.1558  0. 0.  0.
  0 0 0 0 0 0 0 0 0 0
  #60|60|60   step
  1 0 0 0
  'CHANGREF'               ! this positions  'MULTIPOL'
  0 -1.707551921352993 -5.7610304504
'DRIFT'     DD2
9.9
'FAISCEAU'
'REBELOTE'                 ! means 'do it again !', 19000 times, starting
19000 0.2 99               ! with final coordiantes (that's the meaning of '99')
'END'
```

- We now plot x,x’ and z,z’ phase spaces using 7/1/2



x, x’ phase space close to center of long drift.



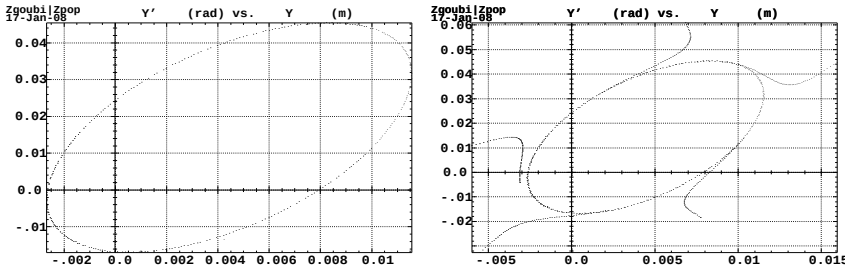
z, z’ phase space close to center of long drift.

EXERCISE #2, cont'd - Now, let's see were painful bugs can come in

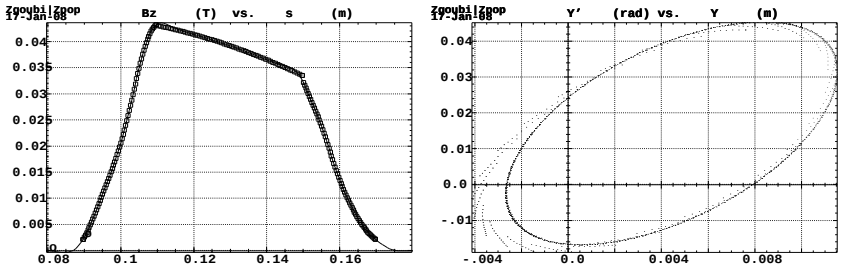
Starting from Ex. 2 data file (copy it into zgoubi.dat), we will try the following 3 experiments :

- increase step size (bugged_step6and30.dat)
- truncate fringe field (bugged_trunkField.dat)
- make the field non-reasonable : unrealistic field fall-off steepness (bugged_steep.dat)

• First, (i) make sure IC=0 in both multipoles, (ii) in 'OBJET', remove vertical motion, (iii) in 'REBELOTE', decrease pass number to 300 about, that will be enough (and fast).
• Second, from this new zgoubi.dat file, get a first **zgoubi** run, defect free, for reference (upper left figure below) and further comparisons.
• Then, introduce the appropriate modifications (bugs) in zgoubi.dat as stated above, and run zgoubi, for each one.
• Make sure you get the results as plotted below.

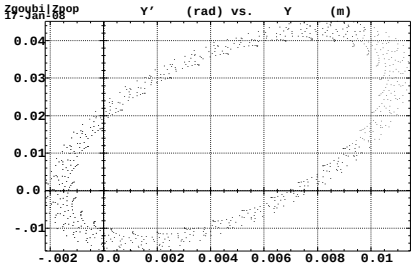


Left : bug free, reference x,x' phase space. Right : number of steps $\#60|60|60$ first changed to $\#30|30|30$ $\rightarrow$ still good, and then to $\#6|6|6$ $\rightarrow$ big change in the motion !



Left : field versus path length, normal case (dots) and case of truncated field fall-off ($XE = XS = 1$ cm instead of 2 cm). Right : as a consequence, evidence for loss of symplecticity - spiraling (bug free, normal ellipse is also shown for reference).

Fringe field fall-off has been decreased (gradient steeper) from 1.1 cm down to 0.5 cm. Symplecticity is affected (motion spirals).



Summary :

Now we are convinced (I hope !) that zgoubi's algorithm can track particles, starting from

(i) initial coordinates

(ii) knowledge of field and derivatives

Let's proceed

4.1.2 Integration in electric fields [4]

• Admitting that $\vec{b} = 0$, $(B\rho)'\vec{u} + B\rho\,\vec{u}' = \frac{\vec{e}}{v} + \vec{u} \times \vec{b}$ reduces to

$$\boxed{(B\rho)'\vec{u} + B\rho\vec{u}' = \frac{\vec{e}}{v}}$$

• which, by successive differentiations, gives the recursive relations

$$\begin{aligned}
&(B\rho)'\vec{u} + B\rho\vec{u}' = \frac{\vec{e}}{v} \\
&(B\rho)''\vec{u} + 2(B\rho)'\vec{u}' + B\rho\vec{u}'' = \left(\frac{1}{v}\right)'\vec{e} + \frac{\vec{e}'}{v} \\
&(B\rho)'''\vec{u} + 3(B\rho)''\vec{u}' + 3(B\rho)'\vec{u}'' + B\rho\vec{u}''' = \left(\frac{1}{v}\right)''\vec{e} + 2\left(\frac{1}{v}\right)'\vec{e}' + \left(\frac{1}{v}\right)\vec{e}'' \qquad (4.1.9)\\
&(B\rho)''''\vec{u} + 4(B\rho)'''\vec{u}' + 6(B\rho)''\vec{u}'' + 4(B\rho)'\vec{u}''' + B\rho\vec{u}'''' = \\
&\qquad \left(\frac{1}{v}\right)'''\vec{e} + 3\left(\frac{1}{v}\right)''\vec{e}' + 3\left(\frac{1}{v}\right)'\vec{e}'' + \frac{1}{v}\vec{e}'''
\end{aligned}$$

• that provide the derivatives $\frac{d^n\vec{u}}{ds^n}$ needed in the Taylor expansions (eq. 4.1.1)

$$\begin{aligned}
\vec{u}' &= \left(\frac{1}{v}\right)\vec{E} - \frac{(B\rho)'}{B\rho}\vec{u} \\
\vec{u}'' &= \left(\frac{1}{v}\right)'\vec{E} + \left(\frac{1}{v}\right)\vec{E}'\,|_{B\rho} - 2\frac{(B\rho)'}{B\rho}\vec{u}' - \frac{(B\rho)''}{B\rho}\vec{u} \\
\vec{u}''' &= \left(\frac{1}{v}\right)''\vec{E} + 2\left(\frac{1}{v}\right)'\vec{E}'\,|_{B\rho} + \frac{1}{v}\vec{E}''\,|_{B\rho} - 3\frac{(B\rho)'}{B\rho}\vec{u}'' - 3\frac{(B\rho)''}{B\rho}\vec{u}' - \frac{(B\rho)'''}{B\rho}\vec{u} \\
\vec{u}'''' &= \left(\frac{1}{v}\right)'''\vec{E} + 3\left(\frac{1}{v}\right)''\vec{E}'\,|_{B\rho} + 3\left(\frac{1}{v}\right)'\vec{E}''\,|_{B\rho} + \left(\frac{1}{v}\right)\vec{E}'''\,|_{B\rho} \\
&\quad - 4\frac{(B\rho)'}{B\rho}\vec{u}''' - 6\frac{(B\rho)''}{B\rho}\vec{u}'' - 4\frac{(B\rho)'''}{B\rho}\vec{u}' - \frac{(B\rho)''''}{B\rho}\vec{u} \\
\vec{u}''''' &= \left(\frac{1}{v}\right)''''\vec{E} + 4\left(\frac{1}{v}\right)'''\vec{E}'\,|_{B\rho} + 6\left(\frac{1}{v}\right)''\vec{E}''\,|_{B\rho} + 4\left(\frac{1}{v}\right)'\vec{E}'''\,|_{B\rho} + \left(\frac{1}{v}\right)\vec{E}''''\,|_{B\rho} \\
&\quad - 5\frac{(B\rho)'}{B\rho}\vec{u}'''' - 10\frac{(B\rho)''}{B\rho}\vec{u}''' - 10\frac{(B\rho)'''}{B\rho}\vec{u}'' - 5\frac{(B\rho)''''}{B\rho}\vec{u}' - \frac{(B\rho)'''''}{B\rho}\vec{u}
\end{aligned} \qquad (4.1.10)$$

where $\vec{E} = \frac{\vec{e}}{B\rho}$, and $(\,)^{(n)}\,|_{B\rho}$ denotes differentiation at constant $B\rho$: $\vec{E}^{(n)}\,|_{B\rho} = \frac{1}{B\rho}\frac{d^n\vec{e}}{ds^n}$.

• These derivatives $\vec{E}^{(n)}\,|_{B\rho}= \dfrac{1}{B\rho}\dfrac{d^n\vec{e}}{ds^n}$ are obtained from the total derivative

$$d\vec{E} = \frac{\partial \vec{E}}{\partial X}\,dX + \frac{\partial \vec{E}}{\partial Y}\,dY + \frac{\partial \vec{E}}{\partial Z}\,dZ, \quad i.e.\ \ \vec{E}' = \sum_i \frac{\partial \vec{E}}{\partial X_i}\,u_i \tag{4.1.11}$$

and then,

$$\begin{aligned}
\vec{E}'' &= \sum_{ij} \frac{\partial^2 \vec{E}}{\partial X_i \partial X_j}\,u_i u_j + \sum_i \frac{\partial \vec{E}}{\partial X_i}\,u_i' \\
\vec{E}''' &= \sum_{ijk} \frac{\partial^3 \vec{E}}{\partial X_i \partial X_j \partial X_k}\,u_i u_j u_k + 3\sum_{ij} \frac{\partial^2 \vec{E}}{\partial X_i \partial X_j}\,u_i' u_j + \sum_i \frac{\partial \vec{E}}{\partial X_i}\,u_i'' \ , \quad \text{etc. as in eq. 4.1.8}
\end{aligned} \tag{4.1.12}$$

• These equations involve derivatives $(B\rho)^{(n)} = \dfrac{d^n(B\rho)}{ds^n}$, which are obtained in the following way. Considering that

$$\frac{dp^2}{dt} = \frac{d\vec{p}^2}{dt} \quad \textit{i.e.,} \quad \frac{dp}{dt}\,p = \frac{d\vec{p}}{dt}\vec{p} \tag{4.1.13}$$

with $\dfrac{d\vec{p}}{dt} = q\,(\vec{e} + \vec{v}\times\vec{b})$ (eq. 4.0.1), we obtain

$$\frac{dp}{dt}\,p = q\,(\vec{e} + v\times\vec{b})\cdot\vec{p} = q\vec{e}\cdot\vec{p} \tag{4.1.14}$$

since $(\vec{v}\times\vec{b})\cdot\vec{p} = 0$. Normalizing as previously with $\vec{p} = p\vec{u} = qB\rho\vec{u}$ and $ds = vdt$, and by successive differentiations, eq. (4.1.14) leads to the $(B\rho)^{(n)}$

$$\begin{aligned}
(B\rho)' &= \frac{1}{v}\,(\vec{e}\cdot\vec{u}) \\
(B\rho)'' &= \left(\frac{1}{v}\right)'(\vec{e}\cdot\vec{u}) + \frac{1}{v}\,(\vec{e}\cdot\vec{u})' \\
(B\rho)''' &= \left(\frac{1}{v}\right)''(\vec{e}\cdot\vec{u}) + 2\left(\frac{1}{v}\right)'(\vec{e}\cdot\vec{u})' + \frac{1}{v}\,(\vec{e}\cdot\vec{u})'' \\
(B\rho)'''' &= \left(\frac{1}{v}\right)'''(\vec{e}\cdot\vec{u}) + 3\left(\frac{1}{v}\right)''(\vec{e}\cdot\vec{u})' + 3\left(\frac{1}{v}\right)'(\vec{e}\cdot\vec{u})'' + \frac{1}{v}\,(\vec{e}\cdot\vec{u})'''
\end{aligned} \tag{4.1.15}$$

Note that the derivatives $(\vec{e}\cdot\vec{u})^{(n)} = \dfrac{d^n(\vec{e}\cdot\vec{u})}{ds^n}$ can be related to the derivatives of the kinetic energy W by $dW = \dfrac{d\vec{p}}{dt}\cdot\vec{v}\,dt = q\vec{e}\cdot\vec{v}\,dt$ which leads to

$$\frac{d^{n+1}W}{ds^{n+1}} = q\frac{d^n(\vec{e}\cdot\vec{u})}{ds^n} \tag{4.1.16}$$

• Finally, the derivatives $\left(\frac{1}{v}\right)^{(n)} = \frac{d^n\left(\frac{1}{v}\right)}{ds^n}$ involved in eqs. (4.1.10,4.1.15) are obtained from $p = \frac{v}{c}\frac{W + m_0c^2}{c}$, ($m_0$ is the rest mass) by successive differentiations, that give the recursive relations

$$\begin{aligned}
\left(\frac{1}{v}\right) &= \frac{1}{c^2}\frac{W + m_0c^2}{qB\rho} \\
\left(\frac{1}{v}\right)' &= \frac{1}{c^2}\frac{(\vec{e}\cdot\vec{u})}{B\rho} - \frac{1}{v}\frac{(B\rho)'}{B\rho} \\
\left(\frac{1}{v}\right)'' &= \frac{1}{c^2}\frac{(\vec{e}\cdot\vec{u})'}{B\rho} - 2\left(\frac{1}{v}\right)'\frac{(B\rho)'}{B\rho} - \frac{1}{v}\frac{(B\rho)''}{B\rho} \\
\left(\frac{1}{v}\right)''' &= \frac{1}{c^2}\frac{(\vec{e}\cdot\vec{u})''}{B\rho} - 3\left(\frac{1}{v}\right)''\frac{(B\rho)'}{B\rho} - 3\left(\frac{1}{v}\right)'\frac{(B\rho)''}{B\rho} - \frac{1}{v}\frac{(B\rho)'''}{B\rho}
\end{aligned} \tag{4.1.17}$$

Again, edit devtra.f, look into it now for Eqs. 4.1.9-4.1.22 series

```
      SUBROUTINE DEVTRA
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
.............................

CALCUL u=DX/DS
      CP=COS(P1)
      U(1,1)=CP*COS(T1)
      U(1,2)=CP*SIN(T1)
      U(1,3)=SIN(P1)

C      umod1 = sqrt(U(1,1)*U(1,1) +U(1,2)*U(1,2) +U(1,3)*U(1,3))

      GOTO(1,2,3) KFLD

C----- MAGNETIC FIELD
 1    CONTINUE

.................

C----- ELECTRIC FIELD
C----- E=e/Bro
 2    CONTINUE

NOTE THAT MASS AND CHARGE ARE NEEDED NOW,
RIGIDITY ALONE IS NOT ENOUGH

      AM2 = AMT*AMT
      CQ = CL9*QT
      BRC = BR*CQ
C----- Wt=sqrt(p2+m2), p=beta.Wt/c
      CSV = SQRT(1.D0 + AM2/(BRC*BRC))
Calcul (e.u)/Bro, Bro'
      EU(1)=( ( E(1,1)*U(1,1)+E(1,2)*U(1,2)+E(1,3)*U(1,3) ) *QT)
      VL=CL9/CSV
      DDT(1) = 1.D0/VL
      DBSB=EU(1)/VL
      DBR(1)=DBSB*BR
Calcul u'=E/v + uxB - uBro'/Bro
      DO 21 K=1,3
 21     U(2,K)=E(1,K)/VL  - U(1,K)*DBSB

      IF(IDS.LE.2) GOTO 99

Calcul e'/Bro = (e/Bro)'|Bro=cste = E'|Bro=cste
      KIM = -IMAX
      DO 23 K=1,3
        TP=0.D0
        IF(IDE.GE.1) THEN
          KIM = KIM + IMAX
          DO 22 I=1,3
            IK = I + KIM
            TP=TP+U(1,I)*DE(IK)
 22       CONTINUE
        ENDIF
        E(2,K)=TP
 23   CONTINUE

Calcul (e.u)'/Bro, Bro''
      EU(2)= ( E(2,1)*U(1,1)+E(2,2)*U(1,2)+E(2,3)*U(1,3)
     >+ E(1,1)*U(2,1)+E(1,2)*U(2,2)+E(1,3)*U(2,3) ) *QT
      DCSV=EU(1)/BRC-CSV*DBSB
      D1SV=DCSV/CL9
      DDT(2) =D1SV
      D2BSB=D1SV*EU(1)+EU(2)/VL
      DBR(2)=D2BSB*BR
Calcul u''=(1/v)'E + (e'/Bro)/v - 2Bro'/Brou' - Bro''/B
      DO 26 K=1,3
        U(3,K)=E(1,K)*D1SV + E(2,K)/VL
     >  -2.D0*U(2,K)*DBSB - U(1,K)*D2BSB
 26   CONTINUE

      IF(IDS.LE.3) GOTO 99

Calcul e''/Bro = E''|Bro=cste
      KIJM = -IJMAX-IMAX
      KIM = -IMAX
      DO 25 K=1,3
        TP=0.D0
        IF(IDE.GE.1) THEN
          KIJM = KIJM + IJMAX
          KIM = KIM + IMAX
          DO 24 I=1,3
            IK = I + KIM
            TP=TP+U(2,I)*DE(IK)
            IF(IDE.GE.2) THEN
              IKIJ = I + KIJM
C
C               symmetry
C
              DO 241 J=I,3
                IF(I .EQ. J) THEN
                  XMUL=1.0D0
                ELSE
                  XMUL=2.0D0
                ENDIF
                IJK = IKIJ + J*IMAX
                TP=TP+XMUL*U(1,I)*U(1,J)*DDE(IJK)
 241          CONTINUE
            ENDIF
 24       CONTINUE
        ENDIF
        E(3,K)=TP
 25   CONTINUE
Calcul (e.u)''/Bro, Bro'''
      EU(3)=(( E(3,1)*U(1,1)+E(3,2)*U(1,2)+E(3,3)*U(1,3
     >+ 2.D0*  ( E(2,1)*U(2,1)+E(2,2)*U(2,2)+E(2,3)*U(2
     >+        E(1,1)*U(3,1)+E(1,2)*U(3,2)+E(1,3)*U(3,3
      D2CSV= EU(2)/BRC - 2.D0*DBSB*DCSV - D2BSB*CSV
      D21SV=D2CSV/CL9
      DDT(3) = D21SV
      D3BSB=D21SV*EU(1)+2.D0*D1SV*EU(2)+EU(3)/VL
      DBR(3)=D3BSB*BR
Calcul u'''=(1/v)''E + 2(1/v)'(e'/Bro) + (e''/Bro)/v
C           - 3Bro'/Brou'' - 3Bro''/Brou' - Bro'''/Brou
      DO 27 K=1,3
        U(4,K)=E(1,K)*D21SV + 2.D0*E(2,K)*D1SV + E(3,K)
     >  -3.D0*( DBSB*U(3,K) + D2BSB*U(2,K) ) - D3BSB*U(
 27   CONTINUE

      IF(IDS.LE.4) GOTO 99

Calcul e'''/Bro = E'''|Bro=cste
      KIJM = -IJMAX-IMAX
      KIM = -IMAX
      DO 28 K=1,3
 .......................;;
        E(4,K)=TP
 28   CONTINUE
Calcul (e.u)'''/Bro, Bro''''
      EU(4)=(( E(4,1)*U(1,1)+E(4,2)*U(1,2)+E(4,3)*U(1,3
     >+ 3.D0* (  E(3,1)*U(2,1)+E(3,2)*U(2,2)+E(3,3)*U(2
     >+ 3.D0* (  E(2,1)*U(3,1)+E(2,2)*U(3,2)+E(2,3)*U(3
     >+        E(1,1)*U(4,1)+E(1,2)*U(4,2)+E(1,3)*U(4,3
      D3CSV= EU(3)/BRC-3.D0*(DBSB*D2CSV + D2BSB*DCSV) -
      D31SV=D3CSV/CL9
      DDT(4) =D31SV
      D4BSB=D31SV*EU(1)+3.D0*(D21SV*EU(2)+D1SV*EU(3))+E
      DBR(4)=D4BSB*BR
Calcul u''''=(1/v)'''E+3(1/v)''(e'/Bro)+3(1/v)'(e''/Bro
C           - 4Bro'/Brou''' - 6Bro''/Brou'' - 4Bro'''/B

.........
```

There is a number of electrostatic elements (Guide, pp 7-9, 139-141) :

2-tube (bipotential) lens	EL2TUB
3-tube (unipotential) lens	UNIPOT
Decapole	ELMULT
Dipole	ELMULT
Dodecapole	ELMULT
Multipole	ELMULT
N-electrode mirror/lens, straight slits	ELMIR
N-electrode mirror/lens, circular slits	ELMIRC
Octupole	ELMULT
Quadrupole	ELMULT
R.F. (kick) cavity	CAVITE
Sextupole	ELMULT
Skewed multipoles	ELMULT

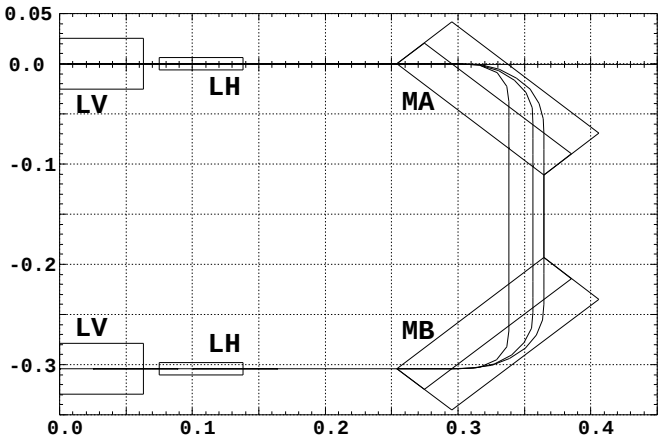
Without forgetting electric field maps

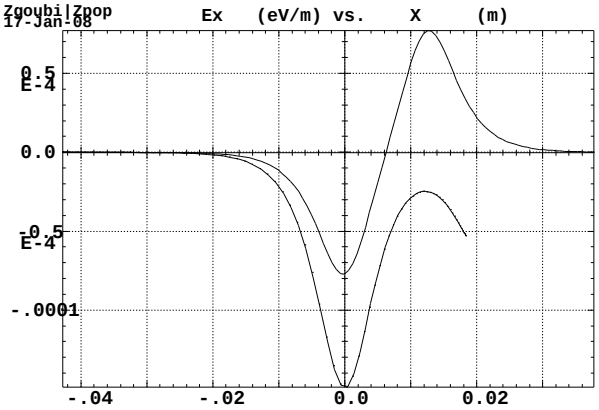
1D, cylindrical symmetry	ELREVOL
2-D, no symmetry	MAP2D_E

Example - 'ELMIR', electrostatic lens or miror (Guide, pp. 92, 171)

We are circular machine folks, so, let's consider that : a ring can be built from that class of optical elements !

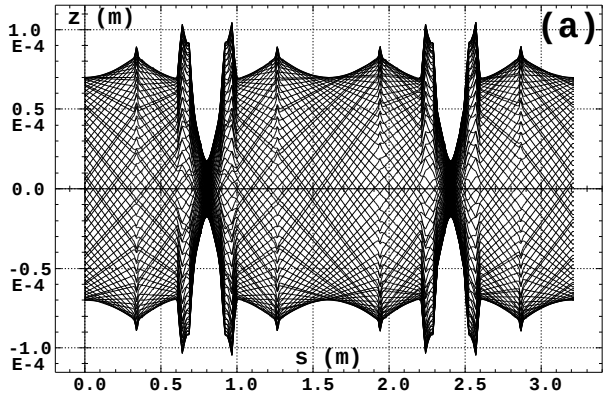
(a TOF mass spectrometer)

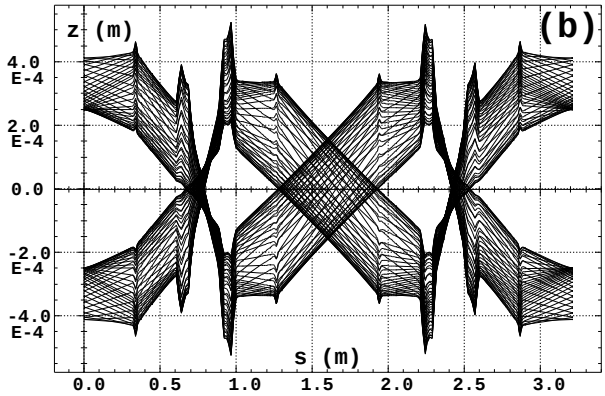




Electrostatic ring. Trajectories over a half-ring (1 cell), obtained doing (i) 7/1/1, (ii) 7/2(8,34), (iii) 7/7

• Check E field... (e.g., E_x in LH, MA lenses, on the right plot)





Vertical envelopes as generated by a particle running 80 turns around the ring. Left : particle launched with $z_0 = 0.07\ 10^{-3}$ m. Right : particle launched with $z_0 = 0.25\ 10^{-3}$ m.

Using 'MATRIX', see data file on next page, a Twiss matrix can be computed, that yields tunes, etc.

```
 1.319333     0.000001     0.000000     0.000000     0.000000     0.000000
 0.000001     0.757959     0.000000     0.000000     0.000000     0.000000
 0.000000     0.000000     0.166466    -0.476734     0.000000     0.000000
 0.000000     0.000000    -0.476734     7.372547     0.000000     0.000000
 0.000000     0.000000     0.000000     0.000000     0.000000     0.000000
 0.000000     0.000000     0.000000     0.000000     0.000000     0.000000

                        Betatron  tunes

          NU_Y = 0.36918035         NU_Z = 0.67530735
```

This matrix can be obtained running elmir_Matrix.dat.

data file for the electrostatic ring

```
   Multitour, 2*MMT   objet pour FigDisp
   'OBJET'
15.23683   Rigidity (kG.cm) -> 0.4 KeV N2+ (mass=28)
  5
0.001  0.001  0.001    0.001     0.  .01
  0.  0.  0.   0.  0.  1.
'FAISTORE'
b_zgoubi.fai   #END
 1
'PARTICUL'
 26.082E3  1.60217733E-19 0. 0. 0.
 'DRIFT'
 3.000  30.7
 'FAISCEAU'
  'ELMIR'         LV1
   00000  200
  3  .02525   0.01250000   .02525 .012    21
 0. 100.    0. LV
 #10|1000|10
1 0. 0. 0.
 'FAISCEAU'
 'DRIFT'
 1.2
  'ELMIR'         LH1
   000200  200
  3  .02525   0.012500   .02525 .012    22
 0. 040.0   0.  LH
 #10|1000|10
1 0. 0. 0.
 'FAISCEAU'
 'DRIFT'
 5.6
'ELMIR'        M1A
   000200  200
   3 .04275   0.0209  .1 .012   11       (meters)
0. 178.45  253.    200.  400.
 #10|1001|10     500.010v102
3 0.   1.56468550E+01  .7853981634
 'FAISCEAU'
  'DRIFT'
 10.
'FAISCEAU'    1QUART
  'DRIFT'
 10.
 'ELMIR'        M1B
   000200  200
   3 .04275   0.0209  .1 .012   11       (meters)
0. 178.45  253.    200.  400.
 #10|1001|10
3 0.   1.56468550E+01  .7853981634
 'DRIFT'
 5.6
  'ELMIR'         LH1
   000200  200
  3  .02525   0.012500   .02525 .012    22
 0. 040.0   0.  LH
 #10|1000|10
 1 0. 0. 0.
 'FAISCEAU'
 'DRIFT'
 1.2
  'ELMIR'         LV1
   000200  200
  3  .02525   0.01250000   .02525 .012    21
 0. 100.    0. LV
 #10|1000|10
 1 0. 0. 0.
 'FAISCEAU'
 'DRIFT'
 3.000  30.7
 'FAISCEAU'

[and then symmetrize]

'FAISCEAU'   #END
'MATRIX'
 1 11
'END'
```

4.1.3 Integration in combined electric and magnetic fields

• When both $\vec{e}$ and $\vec{b}$ are non-zero, the complete eq. (4.1.2) must be considered,

$$\boxed{(B\rho)'\vec{u} + B\rho\vec{u}' = \frac{\vec{e}}{v} + \vec{u}\times\vec{b}}$$

Recursive differentiations give the following relations :

$$\begin{aligned}
(B\rho)''\vec{u} + 2(B\rho)'\vec{u}' + B\rho\vec{u}'' &= \left(\frac{1}{v}\right)'\vec{e} + \left(\frac{1}{v}\right)\vec{e}' + (\vec{u}\times\vec{b})' \\
(B\rho)'''\vec{u} + 3(B\rho)''\vec{u}' + 3(B\rho)'\vec{u}'' + B\rho\vec{u}''' &= \left(\frac{1}{v}\right)''\vec{e} + 2\left(\frac{1}{v}\right)'\vec{e}' + \left(\frac{1}{v}\right)\vec{e}'' + (\vec{u}\times\vec{b})'' \\
(B\rho)''''\vec{u} + 4(B\rho)'''\vec{u}' + 6(B\rho)''\vec{u}'' + 4(B\rho)'\vec{u}''' + B\rho\vec{u}'''' &= \\
\left(\frac{1}{v}\right)'''\vec{e} + 3\left(\frac{1}{v}\right)''\vec{e}' + 3\left(\frac{1}{v}\right)'\vec{e}'' + \frac{1}{v}\vec{e}''' + (\vec{u}\times\vec{b})'''
\end{aligned} \tag{4.1.18}$$

• that provide the derivatives $\dfrac{d^n\vec{u}}{ds^n}$ needed in the Taylor expansions (4.1.1)

$$\begin{aligned}
\vec{u}' &= \left(\frac{1}{v}\right)\vec{E} + (\vec{u}\times\vec{B}) - \frac{(B\rho)'}{B\rho}\vec{u} \\
\vec{u}'' &= \left(\frac{1}{v}\right)'\vec{E} + \left(\frac{1}{v}\right)\vec{E}'\,|_{B\rho} + (\vec{u}\times\vec{B}')'\,|_{B\rho} - 2\frac{(B\rho)'}{B\rho}\vec{u}' - \frac{(B\rho)''}{B\rho}\vec{u} \\
\vec{u}''' &= \left(\frac{1}{v}\right)''\vec{E} + 2\left(\frac{1}{v}\right)'\vec{E}'\,|_{B\rho} + \frac{1}{v}\vec{E}''\,|_{B\rho} + (\vec{u}\times\vec{B})''\,|_{B\rho} - 3\frac{(B\rho)'}{B\rho}\vec{u}'' - 3\frac{(B\rho)''}{B\rho}\vec{u}' - \frac{(B\rho)'''}{B\rho}\vec{u} \\
\vec{u}'''' &= \left(\frac{1}{v}\right)'''\vec{E} + 3\left(\frac{1}{v}\right)''\vec{E}'\,|_{B\rho} + 3(\frac{1}{v})'\vec{E}''\,|_{B\rho} + \left(\frac{1}{v}\right)\vec{E}'''\,|_{B\rho} \\
&\quad + (\vec{u}\times\vec{B})'''\,|_{B\rho} - 4\frac{(B\rho)'}{B\rho}\vec{u}''' - 6\frac{(B\rho)''}{B\rho}\vec{u}'' - 4\frac{(B\rho)'''}{B\rho}\vec{u}' - \frac{(B\rho)''''}{B\rho}\vec{u}
\end{aligned} \tag{4.1.19}$$

where $\vec{E} = \dfrac{\vec{e}}{B\rho}$, $\vec{B} = \dfrac{\vec{b}}{B\rho}$, and $^{(n)}\,|_{B\rho}$ denotes differentiation at constant $B\rho$

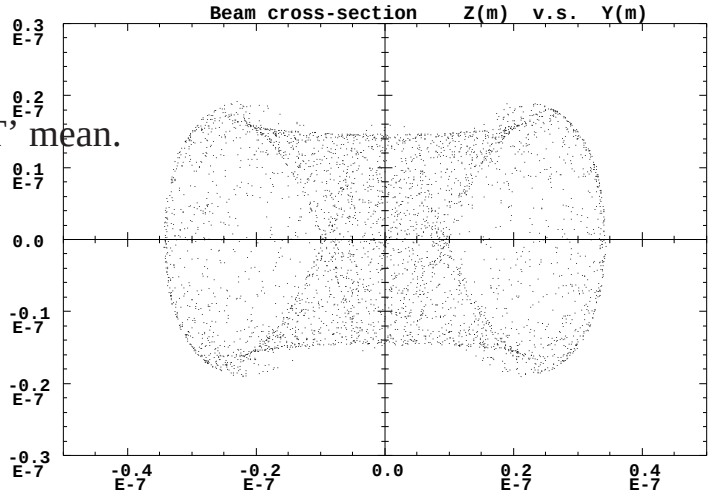
$$\vec{E}^{(n)}\,|_{B\rho} = \frac{1}{B\rho}\frac{d^n\vec{e}}{ds^n} \quad \text{and} \quad (\vec{u}\times\vec{B})^{(n)}\,|_{B\rho} = \frac{1}{B\rho}(\vec{u}\times\vec{b})^{(n)}. \tag{4.1.20}$$

• These derivatives $\vec{E}^{(n)}$ and $\vec{B}^{(n)}$ of the electric and magnetic fields are calculated from the vector fields $\vec{E}(X,Y,Z)$, $\vec{B}(X,Y,Z)$ and their derivatives $\dfrac{\partial^{i+j+k}\vec{E}}{\partial X^i\partial Y^j\partial Z^k}$, $\dfrac{\partial^{i+j+k}\vec{B}}{\partial X^i\partial Y^j\partial Z^k}$ following eqs. (4.1.8) and (4.1.12).

A last look into subroutine devtra.f, now for Eqs. 4.1.18-4.1.20

```
      SUBROUTINE DEVTRA
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C     ----------------------------------------
C     CALCULE u ET SES DERIVEES dnu/dsn, n=1,5,
C     i.e., calcul des coeffs des series de Taylor
C     (eqs. 2.2.4-5)
C     ----------------------------------------
      COMMON/CDF/ IES,LF,LST,NDAT,NRES,NPLT,NFAI,NMAP,NSPN,NLOG
      COMMON/CHAVE/ B(5,3),V(5,3),E(5,3)
      COMMON/D3BXYZ/ D3BX(27), D3BY(27), D3BZ(27)
      COMMON/D4BXYZ/ D4BX(81),D4BY(81),D4BZ(81)
      COMMON/DDEXYZ/ DE(9),DDE(27)
      COMMON/D3EXYZ/ D3EX(27), D3EY(27), D3EZ(27)
      COMMON/D4EXYZ/ D4EX(81), D4EY(81), D4EZ(81)
      COMMON/DDBXYZ/ DB(9),DDB(27)
      COMMON/MARK/ KART,KALC,KERK,KUASEX
      LOGICAL ZSYM
      COMMON/OPTION/ KFLD,MG,LC,ML,ZSYM
      COMMON/ORDRES/ KORD,IRD,IDS,IDB,IDE,IDZ
C      COMMON/RIGID/ BORO,DPREF,DPP,BR
      COMMON/RIGID/ BORO,DPREF,DP,BR
      COMMON/TRAJ/ R1,T1,Z1,P1,A1,SAR,TAR,KEX,IT,AMT,QT
      COMMON/VITES/ U(6,3),DBR(6),DDT(6)

      DIMENSION EU(5)

      PARAMETER(IMAX=3, IJMAX=9, IJLMAX=27, IM2 = IMAX+IJMAX)
      PARAMETER(IM3 = IM2+IJLMAX)

C      SAVE CL, CL9, CQ, AM2, QEV
      SAVE CL, CL9

C-----------------------------------------------------------
C     CALCUL D'ADRESSE : ADRESSE(I,J) = I + (J-1)*IMAX
C     EXEMPLES:
C      DB(2,1) = DB(2)
C      DB(3,3) = DB(9)
C     CALCUL D'ADRESSE : ADRESSE(I,J,K) = I+(J-1)*IMAX+(K-1)*IMAX*JMAX
C     EXEMPLES:
C      DDB(3,1,1) = DDB(3)
C      DDB(3,3,1) = DDB(3*3**0+2*3**1+0*3**2)=DDB(9)
C-----------------------------------------------------------

CALCUL u=DX/DS
      CP=COS(P1)
      U(1,1)=CP*COS(T1)
      U(1,2)=CP*SIN(T1)
      U(1,3)=SIN(P1)

C      umod1 = sqrt(U(1,1)*U(1,1) +U(1,2)*U(1,2) +U(1,3)*U(1,3))

      GOTO(1,2,3) KFLD

C----- MAGNETIC FIELD
 1    CONTINUE
...................
C----- ELECTRIC FIELD
C----- E=e/Bro
 2    CONTINUE
...................
      GOTO 99

C----- MAGNETIC+ELECTRIC FIELD
C----- B=b/Bro, E=e/Bro
 3    CONTINUE

      AM2 = AMT*AMT
      CQ = CL9*QT
      BRC = BR*CQ
C----- Wt=sqrt(p2+m2), p=beta.Wt/c
      CSV = SQRT(1.D0 + AM2/(BRC*BRC))

Calcul (e.u)/Bro, Bro'
      EU(1)=( ( E(1,1)*U(1,1)+E(1,2)*U(1,2)+E(1,3)*U(1,3) ) *QT)
      VL=CL9/CSV
      DDT(1) = 1.D0/VL
      DBSB=EU(1)/VL
      DBR(1)=DBSB*BR
CALCUL u'=uxB
      CALL PVECT(1,1,1)
Calcul u'=E/v + uxB - uBro'/Bro
      DO 31 K=1,3
 31     U(2,K)=E(1,K)/VL + V(1,K) - U(1,K)*DBSB
Calcul e'/Bro = (e/Bro)'|Bro=cste = E'|Bro=cste
      KIM = -IMAX
      DO 33 K=1,3
        TP=0.D0
        IF(IDE.GE.1) THEN
          KIM = KIM + IMAX
          DO 32 I=1,3
            IK = I + KIM
            TP=TP+U(1,I)*DE(IK)
 32       CONTINUE
        ENDIF
        E(2,K)=TP
 33   CONTINUE
Calcul (e.u)'/Bro, Bro''
      EU(2)= ( E(2,1)*U(1,1)+E(2,2)*U(1,2)+E(2,3)*U
     >+ E(1,1)*U(2,1)+E(1,2)*U(2,2)+E(1,3)*U(2,3) )
      DCSV=EU(1)/BRC-CSV*DBSB
      D1SV=DCSV/CL9
      DDT(2) = D1SV
      D2BSB=D1SV*EU(1)+EU(2)/VL
      DBR(2)=D2BSB*BR
C      b'/Bro = B'|Bro=cste
      KIM = -IMAX
      DO 312 K=1,3
        TP=0.D0
        IF(IDB.GE.1) THEN
          KIM = KIM + IMAX
          DO 311 I=1,3
            IK = I + KIM
            TP=TP+U(1,I)*DB(IK)
 311      CONTINUE
        ENDIF
        B(2,K)=TP
 312  CONTINUE
CALCUL (uxb)'/Bro
      CALL PVECT(1,1,2)
      CALL PVECT(2,2,1)
Calcul u''=(1/v)'E + (e'/Bro)/v + (uxb)'/Bro - 2Bro
      DO 36 K=1,3
        U(3,K)=E(1,K)*D1SV + E(2,K)/VL + V(1,K)+V(2
     >  -2.D0*U(2,K)*DBSB - U(1,K)*D2BSB
 36   CONTINUE

      IF(IDS.LE.3) GOTO 99

Calcul e''/Bro = E''|Bro=cste
      KIJM = -IJMAX-IMAX
      KIM = -IMAX
      DO 35 K=1,3
        TP=0.D0
        IF(IDE.GE.1) THEN
          KIJM = KIJM + IJMAX
          KIM = KIM + IMAX
          DO 34 I=1,3
            IK = I + KIM
            TP=TP+U(2,I)*DE(IK)
            IF(IDE.GE.2) THEN
              IKIJ = I + KIJM
C
C               symetrie
C
              DO 341 J=I,3
                IF(I .EQ. J) THEN
                  XMUL=1.0D0
                ELSE
                  XMUL=2.0D0
                ENDIF
                IJK = IKIJ + J*IMAX
                TP=TP+XMUL*U(1,I)*U(1,J)*DDE(IJK)
 341          CONTINUE
            ENDIF
 34       CONTINUE
        ENDIF
        E(3,K)=TP
 35   CONTINUE
Calcul (e.u)''/Bro, Bro'''
      EU(3)=(( E(3,1)*U(1,1)+E(3,2)*U(1,2)+E(3,3)*U
     >+ 2.D0*  ( E(2,1)*U(2,1)+E(2,2)*U(2,2)+E(2,3)
     >+        E(1,1)*U(3,1)+E(1,2)*U(3,2)+E(1,3)*U
...................;
 99   CONTINUE
      IF(LST .GE. 1) CALL IMPDEV
      RETURN

      ENTRY DEVTRW(CEL)
      CL = CEL
      CL9=CL*1.D-9
```

EXERCISE #3 - Magneto-electrostatic quadrupole

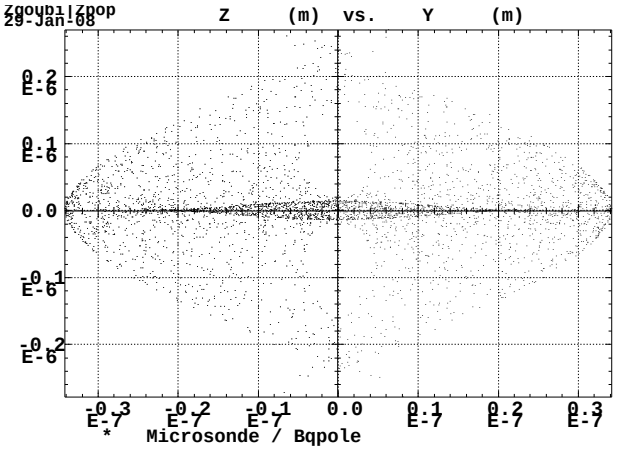


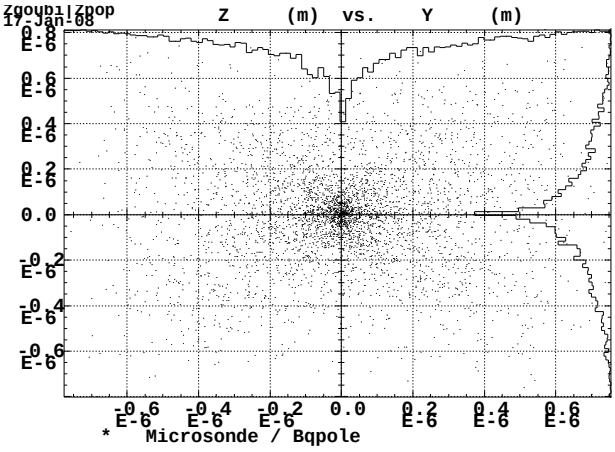
- Using data file 'microProbe_matrix.dat'
  - figure out what the data in 'MCOBJET' mean. Then, run zgoubi
  - check fields ! (home work)
  - save into microProbe_matrix.res
- Using data file 'bqpole_matrix.dat'
  - run zgoubi
  - check fields ! (home work)
  - save into bqpole_matrix.res
- Comparing microProbe_matrix.res and bqpole_matrix.res shows (see next page)
  - same first order matrix
  - chromatic coefficients zero (resp. non-zero) if electrostatic quad is (is not) superimposed.

- using the following 'MCOBJET' (in microProbe_image.dat and bqpole_image.dat):

```
'MCOBJET'                                        1
 20.435                                          H+, 20keV
 1                                               UNIFORM DISTRIBUTION IN WINDOW
 4000                                             NUMGER  OF  PARTICLES
 1    1    1    1    1    1
 0.   0.   0.   0.   0.   1.                     CENTRAL  VALUE
 0.  .2E-3 .0  .2E-3 0.   0.0003                 1/2 WIDTH
 10   10   10   10   10   10                     Cut-offs - unused
 9   9. 9. 9. 9.                                 P(D) - UNUSED
 186387 548728 472874                            SEEDS
```

yields the images at focal plane as shown below - left is with E+B doublet, right is w/o, note the change in scales :





- It can be checked that all chromatic coefficients are zero with E+B doublet. In the case of purely magnetic quads, they are not zero.

```
                TRANSFER  MATRIX  ORDRE  1  (MKSA units)
      -0.142144        2.813325E-04    2.535856E-17    9.388382E-17     0.00000         0.00000
       -1.63367        -7.03191        2.685418E-16    1.306810E-15     0.00000         0.00000
       5.107443E-18   -1.283400E-17   -2.982830E-02    2.180188E-04     0.00000         0.00000
       1.214480E-16    6.835762E-16    -5.94687        -33.4817         0.00000         0.00000
        0.00000         0.00000         0.00000         0.00000         1.00000         0.00000
        0.00000         0.00000         0.00000         0.00000         0.00000         1.00000

       DetY-1 =      -0.0000000293,     DetZ-1 =      -0.0000000293
    First order sympletic conditions (expected values = 0) :
      -2.9312E-08   -2.9336E-08   -2.0341E-16    1.0901E-15    2.3495E-17   -2.7511E-16

                TRANSFER  MATRIX  ORDRE  2  (MKSA units)
1 11   0.00       1 21   0.00       1 31   0.00       1 41   0.00       1 51   0.00       1 61  2.339E-06
1 12   0.00       1 22   0.00       1 32   0.00       1 42   0.00       1 52   0.00       1 62  1.196E-05
1 13   0.00       1 23   0.00       1 33   0.00       1 43   0.00       1 53   0.00       1 63 -4.993E-14
1 14   0.00       1 24   0.00       1 34   0.00       1 44   0.00       1 54   0.00       1 64 -3.268E-13
1 15   0.00       1 25   0.00       1 35   0.00       1 45   0.00       1 55   0.00       1 65   0.00
1 16  2.339E-06   1 26  1.196E-05   1 36 -4.993E-14   1 46 -3.268E-13   1 56   0.00       1 66   0.00

2 11   0.00       2 21   0.00       2 31   0.00       2 41   0.00       2 51   0.00       2 61  4.603E-06
2 12   0.00       2 22   0.00       2 32   0.00       2 42   0.00       2 52   0.00       2 62  2.175E-05
2 13   0.00       2 23   0.00       2 33   0.00       2 43   0.00       2 53   0.00       2 63 -1.706E-13
2 14   0.00       2 24   0.00       2 34   0.00       2 44   0.00       2 54   0.00       2 64 -8.203E-13
2 15   0.00       2 25   0.00       2 35   0.00       2 45   0.00       2 55   0.00       2 65   0.00
2 16  4.603E-06   2 26  2.175E-05   2 36 -1.706E-13   2 46 -8.203E-13   2 56   0.00       2 66   0.00

3 11   0.00       3 21   0.00       3 31   0.00       3 41   0.00       3 51   0.00       3 61  8.105E-15
3 12   0.00       3 22   0.00       3 32   0.00       3 42   0.00       3 52   0.00       3 62  3.180E-14
3 13   0.00       3 23   0.00       3 33   0.00       3 43   0.00       3 53   0.00       3 63  8.070E-06
3 14   0.00       3 24   0.00       3 34   0.00       3 44   0.00       3 54   0.00       3 64  4.633E-05
3 15   0.00       3 25   0.00       3 35   0.00       3 45   0.00       3 55   0.00       3 65   0.00
3 16  8.105E-15   3 26  3.180E-14   3 36  8.070E-06   3 46  4.633E-05   3 56   0.00       3 66   0.00

4 11   0.00       4 21   0.00       4 31   0.00       4 41   0.00       4 51   0.00       4 61  2.826E-14
4 12   0.00       4 22   0.00       4 32   0.00       4 42   0.00       4 52   0.00       4 62  1.339E-13
4 13   0.00       4 23   0.00       4 33   0.00       4 43   0.00       4 53   0.00       4 63  3.105E-05
4 14   0.00       4 24   0.00       4 34   0.00       4 44   0.00       4 54   0.00       4 64  1.778E-04
4 15   0.00       4 25   0.00       4 35   0.00       4 45   0.00       4 55   0.00       4 65   0.00
4 16  2.826E-14   4 26  1.339E-13   4 36  3.105E-05   4 46  1.778E-04   4 56   0.00       4 66   0.00
```

Case of magnetic doublet :

```
                TRANSFER  MATRIX  ORDRE  1  (MKSA units)
      -0.142190        2.604669E-05   -4.597290E-38   -1.052423E-38     0.00000         0.00000
       -1.63378        -7.03254        1.687465E-36    1.035149E-35     0.00000         0.00000
      -2.907972E-38   -6.708125E-39   -2.986377E-02    1.343496E-05     0.00000         0.00000
       1.061254E-36    6.514339E-36    -5.94704        -33.4827         0.00000         0.00000
        0.00000         0.00000         0.00000         0.00000         1.00000         0.00000
        0.00000         0.00000         0.00000         0.00000         0.00000         1.00000

                TRANSFER  MATRIX  ORDRE  2  (MKSA units)
1 11   0.00       1 21   0.00       1 31   0.00       1 41   0.00       1 51   0.00       1 61   1.15
1 12   0.00       1 22   0.00       1 32   0.00       1 42   0.00       1 52   0.00       1 62   6.27
1 13   0.00       1 23   0.00       1 33   0.00       1 43   0.00       1 53   0.00       1 63  4.828E-37
1 14   0.00       1 24   0.00       1 34   0.00       1 44   0.00       1 54   0.00       1 64 -2.154E-36
1 15   0.00       1 25   0.00       1 35   0.00       1 45   0.00       1 55   0.00       1 65   0.00
1 16   1.15       1 26   6.27       1 36  4.828E-37   1 46 -2.154E-36   1 56   0.00       1 66   0.00

2 11   0.00       2 21   0.00       2 31   0.00       2 41   0.00       2 51   0.00       2 61   2.85
2 12   0.00       2 22   0.00       2 32   0.00       2 42   0.00       2 52   0.00       2 62   15.4
2 13   0.00       2 23   0.00       2 33   0.00       2 43   0.00       2 53   0.00       2 63 -3.872E-37
2 14   0.00       2 24   0.00       2 34   0.00       2 44   0.00       2 54   0.00       2 64 -1.663E-35
2 15   0.00       2 25   0.00       2 35   0.00       2 45   0.00       2 55   0.00       2 65   0.00
2 16   2.85       2 26   15.4       2 36 -3.872E-37   2 46 -1.663E-35   2 56   0.00       2 66   0.00

3 11   0.00       3 21   0.00       3 31   0.00       3 41   0.00       3 51   0.00       3 61  8.325E-38
3 12   0.00       3 22   0.00       3 32   0.00       3 42   0.00       3 52   0.00       3 62 -2.158E-37
3 13   0.00       3 23   0.00       3 33   0.00       3 43   0.00       3 53   0.00       3 63   1.02
3 14   0.00       3 24   0.00       3 34   0.00       3 44   0.00       3 54   0.00       3 64   5.89
3 15   0.00       3 25   0.00       3 35   0.00       3 45   0.00       3 55   0.00       3 65   0.00
3 16  8.325E-38   3 26 -2.158E-37   3 36   1.02       3 46   5.89       3 56   0.00       3 66   0.00

4 11   0.00       4 21   0.00       4 31   0.00       4 41   0.00       4 51   0.00       4 61 -7.642E-37
4 12   0.00       4 22   0.00       4 32   0.00       4 42   0.00       4 52   0.00       4 62 -6.730E-36
4 13   0.00       4 23   0.00       4 33   0.00       4 43   0.00       4 53   0.00       4 63   4.72
4 14   0.00       4 24   0.00       4 34   0.00       4 44   0.00       4 54   0.00       4 64   27.0
4 15   0.00       4 25   0.00       4 35   0.00       4 45   0.00       4 55   0.00       4 65   0.00
```

```
4 16 -7.642E-37   4 26 -6.730E-36   4 36   4.72       4 46   27.0       4 56   0.00       4 66   0.00
```

4.1.4 Calculation of the time of flight

The time of flight

$$T(M_1) \approx T(M_0) + \frac{dT}{ds}(M_0)\,\Delta s + \frac{d^2T}{ds^2}(M_0)\,\frac{\Delta s^2}{2} + \frac{d^3T}{ds^3}(M_0)\,\frac{\Delta s^3}{3!} + \frac{d^4T}{ds^4}(M_0)\,\frac{\Delta s^4}{4!} \quad (4.1.21)$$

involves the derivatives $dT/ds = 1/v,\ \ d^2T/ds^2 = d(1/v)/ds, etc.$, these are obtained from :

$$\begin{aligned}
\left(\frac{1}{v}\right) &= \frac{1}{c^2}\,\frac{W + m_0c^2}{qB\rho} \\
\left(\frac{1}{v}\right)' &= \frac{1}{c^2}\,\frac{(\vec{e}\cdot\vec{u})}{B\rho} - \frac{1}{v}\,\frac{(B\rho)'}{B\rho} \\
\left(\frac{1}{v}\right)'' &= \frac{1}{c^2}\,\frac{(\vec{e}\cdot\vec{u})'}{B\rho} - 2\left(\frac{1}{v}\right)'\frac{(B\rho)'}{B\rho} - \frac{1}{v}\,\frac{(B\rho)''}{B\rho} \\
\left(\frac{1}{v}\right)''' &= \frac{1}{c^2}\,\frac{(\vec{e}\cdot\vec{u})''}{B\rho} - 3\left(\frac{1}{v}\right)''\frac{(B\rho)'}{B\rho} - 3\left(\frac{1}{v}\right)'\frac{(B\rho)''}{B\rho} - \frac{1}{v}\,\frac{(B\rho)'''}{B\rho}
\end{aligned} \quad (4.1.22)$$

• Note that, in the absence of electric field, the Taylor expansion eq. (4.1.7) reduces to the simple form

$$T(M_1) = T(M_0) + \Delta s/v \quad (4.1.23)$$

• Note :

For calculation of time of flight, 'PARTICUL' keyword is needed, because it requires knowledge of velocity, which cannot be drawn from just the rigidity. Just like when using electrostatic elements that cause a variation of the velocity.

Otherwise, when dealing with B fields only, Zgoubi only needs the rigidity, no matter what Q and Mass are. Just like Matrix Transport codes (!!), Zgoubi works with strengths, so once it has read data from zgoubi.dat, it first converts fields and geometry into strengths, for instance :

- B and $B\rho$ will yield dipole strength $1/\rho = B\ /\ B\rho$
- $B, radius$ and $B\rho$ will yield quadrupole strength $K = B/radius\ /\ B\rho$
- etc.

EXERCISE #4 - TOF parabola in EMMA

• Create an 'exercise4' directory. Go in.

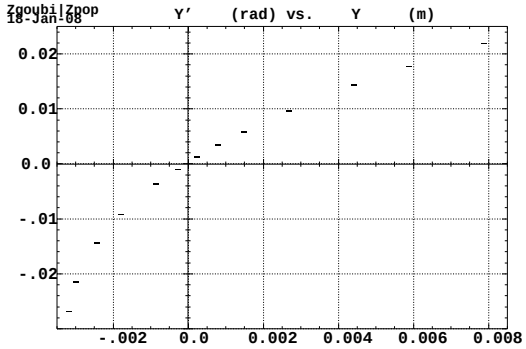
• We use 'exercise4.dat', a copy of Exercise #2 data file in which have been added coordinates of 13 closed orbits ranging from 10 to 20 MeV.

(i) make sure 'IC=0' in the first quads (to save on i/o CPU time, and avoid monstruous zgoubi.plt output)

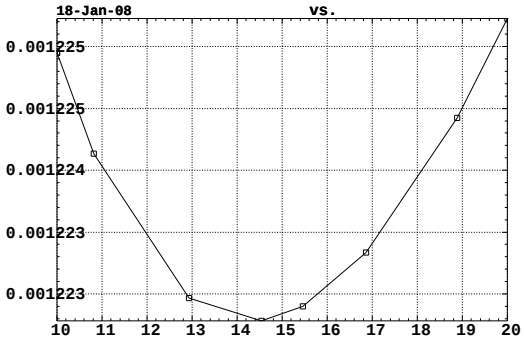
(ii) run zgoubi

(iii) check that we do have closed orbits. There are two simple ways for that, try the other one, too[1]

(iv) check TOF parabola, either from zgoubi.res, at last occurence of 'FAISCEAU', or plotting from 'b_zgoubi.fai' with zpop, see below.



Phase space trajectories, 200 turns in a cell (each dot on the plot is in fact a superimposition of 200 successive (x,x') values), 13 different closed orbits.



TOF (μs) versus Energy (MeV) in EMMA, single cell.

[1](i) with single pass, looking at final 'FAISCEAU' in zgoubi.res, one observes equal initial and final coordinates, (ii) running the data file with multi-turn option ('REBELOTE'), then, plotting with zpop, one gets zero emittance trajectories (dots) in x,x' phase space ;

4.2 How $\vec{B}$ and Derivatives Are Provided

• Remember : these quantities are needed for solving the equation of motion. $\vec{B}(X,Y,Z)$ and derivatives are calculated in various ways, depending whether field maps or analytic representations of optical elements are used.

The basic means are the following.

4.2.1 Extrapolation from 1-D axial field map [5]

• Optical elements of concern : BREVOL (1-D field map)

Possible application : solenoidal field, e.g., solenoid with shielding, from magnet computation/measurement.

- A cylindrically symmetric field can be described by an axial 1-D field map of its longitudinal component $B_X(X, r = 0)$ $(r = (Y^2 + Z^2)^{1/2})$, while the radial component on axis $B_r(X, r = 0)$ is assumed to be zero.

- $B_X(X, r = 0)$ is obtained at any point along the X-axis by a polynomial interpolation from the map mesh (see section 4.3.1).

- Then the field components $B_X(X, r)$, $B_r(X, r)$ at the position of the particle, (X, r) are obtained from Taylor expansions truncated at the fifth order in r (hence, up to the fifth order derivative $\dfrac{\partial^5 B_X}{\partial X^5}(X, 0)$), assuming cylindrical symmetry

$$\begin{aligned} B_X(X,r) &= B_X(X,0) - \frac{r^2}{4}\frac{\partial^2 B_X}{\partial X^2}(X,0) + \frac{r^4}{64}\frac{\partial^4 B_X}{\partial X^4}(X,0) \\ B_r(X,r) &= -\frac{r}{2}\frac{\partial B_X}{\partial X}(X,0) + \frac{r^3}{16}\frac{\partial^3 B_X}{\partial X^3}(X,0) - \frac{r^5}{384}\frac{\partial^5 B_X}{\partial X^5}(X,0) \end{aligned} \tag{4.2.1}$$

By differentiation with respect to X and r, up to the second order, these expressions provide the derivatives of $\vec{B}(X, r)$. Finally a conversion from the (X, r) coordinates to the (X, Y, Z) Cartesian coordinates of **zgoubi** is performed, thus providing the expressions $\dfrac{\partial^{i+j+k}\vec{B}}{\partial X^i \partial Y^j \partial Z^k}$ needed in the eq. (4.1.8).

4.2.2 Extrapolation from Median Plane Fields

• Optical elements of concern : *VENUS* (a magnet at SATURNE and CERN), *SEXQUAD*, *QUADISEX* (special multipoles), *BEND* (dipole), *CARTEME* *TOSCA* (2-D field maps)

In the median plane, $B_Z(X, Y, 0)$, and its derivatives with respect to X or Y, may be calculated from analytical models.

Median plane antisymmetry is assumed, which results in

$$\begin{aligned} B_X(X, Y, 0) &= 0 \\ B_Y(X, Y, 0) &= 0 \\ B_X(X, Y, Z) &= -B_X(X, Y, -Z) \\ B_Y(X, Y, Z) &= -B_Y(X, Y, -Z) \\ B_Z(X, Y, Z) &= B_Z(X, Y, -Z) \end{aligned} \tag{4.2.2}$$

Accomodated with Maxwell's equations, this results in Taylor expansions below, for the three components of $\vec{B}$ (here, B stands for $B_Z(X, Y, 0)$)

$$\begin{aligned} B_X(X, Y, Z) &= Z\frac{\partial B}{\partial X} - \frac{Z^3}{6}\left(\frac{\partial^3 B}{\partial X^3} + \frac{\partial^3 B}{\partial X \partial Y^2}\right) \\ B_Y(X, Y, Z) &= Z\frac{\partial B}{\partial Y} - \frac{Z^3}{6}\left(\frac{\partial^3 B}{\partial X^2 \partial Y} + \frac{\partial^3 B}{\partial Y^3}\right) \\ B_Z(X, Y, Z) &= B - \frac{Z^2}{2}\left(\frac{\partial^2 B}{\partial X^2} + \frac{\partial^2 B}{\partial Y^2}\right) + \frac{Z^4}{24}\left(\frac{\partial^4 B}{\partial X^4} + 2\frac{\partial^4 B}{\partial X^2 \partial Y^2} + \frac{\partial^4 B}{\partial Y^4}\right) \end{aligned} \tag{4.2.3}$$

which are then differentiated one by one with respect to X, Y, or Z, up to second or fourth order (depending on optical element or *IORDRE* option, see section 4.3.2) so as to get the expressions involved in eq. (4.1.8).

Example, BENDF (we will soon use 'BEND'...).

• Let's first look into chamc.f, edit chamc.f, that's where BENDF ('BEND' field) is called

```
        ELSEIF(KUASEX .EQ. 8 ) THEN

C--------- BEND
          IF(BM(6) .NE. 0) THEN
            Y0=Y
            Z0=Z
            CALL ROTX(BM(6),Y0,Z0)
          ENDIF
          CALL BENDF(BM(1),MPOL,XE,XS,CE,CS,DLE,DLS,DI,DS,RTB,X,Y,
     >                                                        BZ0)
        ENDIF

        CALL SYMMED(Z,IDZ,BZ0,
     >                    B,DB,DDB,D3BX,D3BY,D3BZ,D4BX,D4BY,D4BZ)

        IF(KUASEX .EQ. 8 ) THEN
          IF(BM(6) .NE. 0) THEN
            CALL XROTB(BM(6),B,DB,DDB,D3BX,D3BY,D3BZ,D4BX,D4BY,D4BZ)
          ENDIF
        ENDIF
```

BENDF only computes the field and derivatives *in the median plane*, Z-extrapolation is next performed by SYMMED.

• Edit bendf.f :

```
      SUBROUTINE BENDF(BN,MPOL,XE,XS,CE,CS,DLE,DLS,DE,DS,RTB,X,Y,
     >                                                        BZ0)
C see FM CeeRainer 11/2003. QCE in CHAMC
C      SUBROUTINE BENDF(BN,MPOL,DLE,DLS,DE,DS,RTB,X,Y,
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(MCOEF=6)
      DIMENSION CE(MCOEF), CS(MCOEF)
      DIMENSION DLE(MPOL),DLS(MPOL),DE(MPOL,10),DS(MPOL,10),RTB(MPOL),
      DIMENSION BZ0(5,*)

      COMMON/CDF/ IES,LF,LST,NDAT,NRES,NPLT,NFAI,NMAP,NSPN,NLOG
      COMMON/CONST2/ ZERO, UN
      COMMON/INTEG/ PAS,DXI,XLIM,XCE,YCE,ALE,XCS,YCS,ALS,KP
      COMMON/RIGID/ BORO,DPREF,DP,BR

      SAVE IOPIN, IOPOU, DISTE, DISTS

      CTE=RTB(1)
      STE=RTB(2)
      CTS=RTB(4)
      STS=RTB(5)
C-----  dSE/dx=-cosTE=-CTE , dSE/dy = -sinTE =-STE
C-----  dSS/dx=-cosTS=-CTS , dSS/dy = +sinTS =+STS
      SE = ( (XE-X)*CTE - Y*STE )

      IF( SE*SE .LT. 1.D-16) SE=ZERO

      IF(SE*SE .LE. XE*XE) THEN
        IF(DLE(1) .NE. ZERO) THEN
          SE=SE/DLE(1)
          CALL DRVG(4,CE,SE,GE,DGE,D2GE,D3GE,D4GE,D5GE,D6GE)
C     >                                    D7GE,D8GE,D9GE,D10GE)
          DGE  = DGE  * DE(1,1)/BR
          D2GE = D2GE * DE(1,2)/BR
          D3GE = D3GE * DE(1,3)/BR
          D4GE = D4GE * DE(1,4)/BR
        IF(IOPIN.LT.0) THEN
          IF(SE.LT.0.D0) THEN
            SE = -SE*DLE(1)
            GE = 2.D0 * (GE - 0.5D0)*SE/XE
            DGE  = 2.D0 * DGE  * SE/XE !!! + 2.D0 *
            D2GE = 2.D0 * D2GE * SE/XE !!! + 4.D0 *
            D3GE = 2.D0 * D3GE * SE/XE !!! + 6.D0 *
            D4GE = 2.D0 * D4GE * SE/XE !!! + 8.D0 *
          ELSE
            GE = ZERO
            DGE = ZERO
            D2GE= ZERO
            D3GE= ZERO
            D4GE= ZERO
          ENDIF
        ENDIF
      ELSE
C--------- DLE=0. => XE=0.
        GE = 1.D0
        DGE = ZERO
        D2GE= ZERO
        D3GE= ZERO
        D4GE= ZERO
      ENDIF
    ELSE
      IF(SE .GT. ZERO) THEN
        GE= ZERO
      ELSE
        GE = 1.D0
      ENDIF
      DGE = ZERO
      D2GE= ZERO
      D3GE= ZERO
      D4GE= ZERO
    ENDIF

    XLS = XLIM-XS
    SS = -( (XS-X)*CTS + Y*STS )

    IF( SS*SS .LT. 1.D-16) SS=ZERO

    IF(SS*SS .LE. XLS*XLS) THEN
      IF(DLS(1) .NE. ZERO) THEN
        SS = SS/DLS(1)
        CALL DRVG(4,CS,SS,GS,DGS,D2GS,D3GS,D4GS,D5GS
C     >                                    D7GS,D8GS,D9
        DGS  = DGS  * DS(1,1)/BR
        D2GS = D2GS * DS(1,2)/BR
        D3GS = D3GS * DS(1,3)/BR
        D4GS = D4GS * DS(1,4)/BR
.................................
    BZ = ( GE+GS-1.D0 )*BN/BR

    BZX=  DGE*CTE - DGS*CTS
    BZY=  DGE*STE + DGS*STS

    BZXX= D2GE*CTE2    + D2GS*CTS2
    BZXY= D2GE*CTE*STE - D2GS*CTS*STS
    BZYY= D2GE*STE2    + D2GS*STS2

    BZXXX=  D3GE*CTE2*CTE - D3GS*CTS2*CTS
    BZXXY=  D3GE*CTE2*STE + D3GS*CTS2*STS
    BZXYY=  D3GE*CTE*STE2 - D3GS*CTS*STS2
    BZYYY=  D3GE*STE2*STE + D3GS*STS2*STS

    BZ0(1,1)=BZ
    BZ0(2,1)=BZX
    BZ0(1,2)=BZY
    BZ0(3,1)=BZXX
    BZ0(2,2)=BZXY
    BZ0(1,3)=BZYY
    BZ0(4,1)=BZXXX
    BZ0(3,2)=BZXXY
    BZ0(2,3)=BZXYY
    BZ0(1,4)=BZYYY

    RETURN
.........................
```

SYMMED is in symmed.f. Let's edit symmed.f :

What it does is, given Z and the field dericatives, it extrapolates.
This applies to many elements in **zgoubi** , that are introduced in that simple way : just needs a model of the field in the median plane
Advantage : it allows e.g. introducing elements with unknown off-mid plane field Z-dependance.

```
      SUBROUTINE SYMMED(Z,IDZ,BZ0,
     >                          B,DB,DDB,D3BX,D3BY,D3BZ,D4BX,D4BY,D4BZ)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION BZ0(5,5)
      DIMENSION B(5,3),DB(3,3),DDB(3,3,3)
      DIMENSION D3BX(3,3,3), D3BY(3,3,3), D3BZ(3,3,3)
      DIMENSION D4BX(3,3,3,3) ,D4BY(3,3,3,3) ,D4BZ(3,3,3,3)
C     -------------------------------------------------
C     CARTES  2-D  OU  ELEMENTS  DEFINIS  PAR  BZ(X,Y,0):
C     SYMMETRISE PAR RAPPORT AU PLAN MEDIAN
C     -------------------------------------------------
      COMMON/CDF/ IES,LF,LST,NDAT,NRES,NPLT,NFAI,NMAP,NSPN,NLOG
C Reminicences from ancient times
      B0=BZ0(1,1)
      B1=BZ0(2,1)
      B2=BZ0(1,2)
      B11=BZ0(3,1)
      B12=BZ0(2,2)
      B22=BZ0(1,3)
      B111=BZ0(4,1)
      B112=BZ0(3,2)
      B122=BZ0(2,3)
      B222=BZ0(1,4)
      B1111=BZ0(5,1)
      B1112=BZ0(4,2)
      B1122=BZ0(3,3)
      B1222=BZ0(2,4)
      B2222=BZ0(1,5)
      B33=-(B11+B22)
CALCUL DES DERIVEES DE BX, BY, BZ(X,Y,Z) PAR DEVELOPPEMENTS DE TAYLOR
C EN Z. LES DERIVEES PAR RAPPORT A Z SONT EXPRIMEES EN TERMES DE
C COMBINAISONS DES DERIVEES PAR RAPPORT A X ET Y DU PLAN MEDIAN.
C      Z=Z1
      ZB33 = Z*B33
      DDB(1,1,1) = Z * B111
      DDB(2,1,1) = Z * B112
      DDB(2,2,1) = Z * B122
      DDB(2,2,2) = Z * B222

      ZZB12 = -Z * (DDB(1,1,1) + DDB(2,2,1))
      ZZB21 = -Z * (DDB(2,1,1) + DDB(2,2,2))
      DB(1,1) = Z*B11
      DB(2,1) = Z*B12
      DB(2,2) = Z*B22
      DB(3,1) = B1 + 0.5D0 * ZZB12
      DB(3,2) = B2 + 0.5D0 * ZZB21

      DDB(3,1,1)=B11
      DDB(3,2,1)=B12
      DDB(3,2,2)=B22

C----- CALCUL DE BX, BY, BZ(X,Y,Z) PAR EXTRAPOLATION EN Z
      B(1,1) = Z * B1 + Z * ZZB12 /6.D0
      B(1,2) = Z * B2 + Z * ZZB21 /6.D0
      B(1,3) = B0 + Z *  ZB33 * .5D0

      IF(IDZ.GE.3) THEN
C--------- EXTRAPOLATION A L'ORDRE 3 EN Z
        D3BX(3,1,1) = B111
        D3BX(3,2,1) = B112
        D3BX(3,2,2) = B122
        D3BY(3,2,2) = B222

        IF(IDZ.GE.4) THEN

C--------- EXTRAPOLATION A L'ORDRE 4 EN Z
          D4BX(3,1,1,1) = B1111
          D4BX(3,2,1,1) = B1112
          D4BX(3,2,2,1) = B1122
          D4BX(3,2,2,2) = B1222
          D4BY(3,2,2,2) = B2222

          D4BX(3,3,3,1)=-(B1111+B1122)
          D4BX(3,3,3,2)=-(B1112+B1222)
          D4BY(3,3,3,2)=-(B1122+B2222)

          D3BX(2,1,1)= Z*B1112
          D3BX(2,2,1)= Z*B1122
          D3BX(2,2,2)= Z*B1222
          D3BZ(3,3,3)=Z*(B1111+2.D0*B1122+B2222)
          D3BX(3,3,1)=Z*D4BX(3,3,3,1)
          D3BX(3,3,2)=Z*D4BX(3,3,3,2)
          D3BY(3,3,2)=Z*D4BY(3,3,3,2)

          Z2B422=Z*D3BX(3,3,1)*.5D0
          Z2B313=Z*D3BX(3,3,2)*.5D0
          Z2B224=Z*D3BY(3,3,2)*.5D0

          Z3B44=Z*Z*D3BZ(3,3,3)
          Z3B422=Z*Z2B422/6.D0
          Z3B313=Z*Z2B313/6.D0
          Z3B224=Z*Z2B224/6.D0

          B(1,3) = B(1,3) + Z * Z3B44 /24.D0
          DB(1,1) = DB(1,1) + Z3B422
          DB(2,1) = DB(2,1) + Z3B313
          DB(2,2) = DB(2,2) + Z3B224

          DDB(3,1,1)=B11+Z2B422
          DDB(3,2,1)=B12+Z2B313
          DDB(3,2,2)=B22+Z2B224

          D3BX(1,1,1)=Z*B1111
          D3BY(2,2,2)=Z*B2222
        ENDIF
      ENDIF

      RETURN
      END
```

By the way... let's stop a few minutes here and see how an optical element is initialized

We'll do it for 'BEND'

- It starts with a 'call' in zgoubi. Edit zgoubi.f, and also LSTKEY.H.
  Find 'BEND', CALL RBEND

- A read into zgoubi.dat follows. That's done by 'rbend.f'

- Then follows initialization of COMMONs, and various variables to be used by bendf.f.
  This is done in bendi.f. Edit it, take a look.

While we are here, let's see how tracking follows, once an optical element has been initialized

• next to reading from zgoubi.dat (ex.: 'rbend') 'quasex' is called for elements defined in cartesian frame (or 'aimant' for cylindrical frame)

• 'quasex' (or 'aimant') calls 'chxc' ('chxp') that does the initialization of COMONs ('call bendi')

• Then integration is launched : 'transf', 'integr', 'chamc' (or 'chamk' for field maps)... Edit integr.f, go through it

4.2.3 Extrapolation from arbitrary 2-D Field Maps

• Optical elements of concern : *MAP2D* (magnetic), *MAP2D-E* (electric)

2-D field maps that give the three components $B_X(X, Y, Z_0)$, $B_Y(X, Y, Z_0)$ and $B_Z(X, Y, Z_0)$ at each node (X, Y) of a Z_0 Z-elevation map may be used. $\vec{B}$ and its derivatives at any point (X, Y, Z) are calculated by polynomial interpolation followed by Taylor expansions in Z, without any hypothesis of symmetries (see section 4.3.3 and keywords *MAP2D*, *MAP2D-E*).

4.2.4 Interpolation in 3-D Field Maps [6]

In 3-D field maps $\vec{B}$ and its derivatives up to the second order with respect to X, Y, or Z are calculated by means of a second order polynomial interpolation, from 3-D $3 \times 3 \times$ 3-point grid (see section 4.3.4).

4.2.5 2-D Analytical Field Models and Extrapolation

Several optical elements such as *BEND*, *WIENFILT* (that uses the *BEND* procedures), *QUADISEX*, *VENUS*, etc., are defined from the expression of the field and derivatives in the median plane. 3-D extrapolation of these off the median is drawn from Taylor expansions.

4.2.6 3-D Analytical Models of Fields

• Optical elements of concern : *QUADRUPO, SEXTUPOL, MULTIPOL, EBMULT*

The three components of $\vec{B}$ and their derivatives with respect to X, Y or Z are obtained at any step along trajectories from analytical expression drawn from the scalar potential $V(X,Y,Z)$ following

$$B_X = \frac{\partial V}{\partial X}, \quad B_Y = \frac{\partial V}{\partial Y}, \quad B_Z = \frac{\partial V}{\partial Z}, \quad \frac{\partial B_X}{\partial X} = \frac{\partial^2 V}{\partial X^2}, \quad \frac{\partial B_X}{\partial Y} = \frac{\partial^2 V}{\partial X \partial Y}, \quad \text{etc.} \tag{4.2.4}$$

Multipoles

The scalar potential used for the calculation of $\frac{\partial^{i+j+k}\vec{B}_n(X,Y,Z)}{\partial X^i \partial Y^i \partial Z^k}$ ($i+j+k = 0$ to 4) in the case of magnetic and electro-magnetic multipoles with $2n$ poles (namely, *QUADRUPO* ($n=2$) to *DODECAPO* ($n=6$), *MULTIPOL* ($n=1$ to 10), *EBMULT* ($n=1$ to 10)) is [7]

$$V_n(X,Y,Z) = (n!)^2 \left(\sum_{q=0}^{\infty} (-1)^q \frac{G^{(2q)}(X)(Y^2+Z^2)^q}{4^q q!(n+q)!} \right) \left(\sum_{m=0}^{n} \frac{sin\left(m\frac{\pi}{2}\right) Y^{n-m} Z^m}{m!(n-m)!} \right) \tag{4.2.5}$$

where $G(X)$ is the longitudinal gradient, defined at the entrance or exit of the optical element by

$$G(s) = \frac{G_0}{1+\exp(P(s))}, \quad G_0 = \frac{B_0}{R_0^n} \tag{4.2.6}$$

wherein

$$P(s) = C_0 + C_1\left(\frac{s}{\lambda}\right) + C_2\left(\frac{s}{\lambda}\right)^2 + C_3\left(\frac{s}{\lambda}\right)^3 + C_4\left(\frac{s}{\lambda}\right)^4 + C_5\left(\frac{s}{\lambda}\right)^5$$

and s is the distance to the EFB.

Skewed multipoles

A multipole component with arbitrary order n can be tilted independently of the others by an arbitrary angle A_n around the X-axis. If so, the calculation of the field and derivatives in the rotated axis (X, Y_R, Z_R) is done in two steps. First, they are calculated at the rotated position (X, Y_R, Z_R), in the (X, Y, Z) frame, as derived from expression (4.2.5) above. Second, $\vec{B}$ and its derivatives at (X, Y_R, Z_R) in the (X, Y, Z) frame are transformed to the rotated (X, Y_R, Z_R) frame by a rotation of the same angle A_n.

In particular a skewed $2n$-pole component is created by taking $A_n = \pi/2n$.

EXERCISE #5-a - 'MULTIPOLE' is a very important element, it is very often used. Let's see how it works

• Edit chamc.f, find the following sequence :

```
      IF( KALC .EQ. 3) THEN
C------- DIP, QUAD, SEXTU, ..., MULT, SOLENO...

        CALL RAZDRV(KFLD,IDB,IDE)

        IF(KUASEX.LE.MPOL+1) THEN
C--------- DIP, QUAD, SEXTU, etc. up to 20-POLE ; MULTIPOL ; ELMULT ; EBMULT

          IF(KFLD .EQ. MG) THEN
C----------- Magnetic
c        write(*,*) ' sbr chamc di ds ',di(1,1),di(1,2),di(1,3),di(1,4)
            CALL MULTIP(IDB,MG,KUASEX,X,Y,Z,BM,DLE,DLS,DFM,DI,DS,RTB,
     >               XE,XS,CE, CS,
     >               B,DB,DDB,D3BX,D3BY,D3BZ,D4BX,D4BY,D4BZ,BT)
          ELSEIF(KFLD .EQ. LC) THEN
C----------- Electric
            CALL MULTIP(IDE,LC,KUASEX,X,Y,Z,EM,QLE,QLS,QFM,QE,QS,RTQ,
     >               XE,XS,QCE,QCS,
     >               E,DE,DDE,D3EX,D3EY,D3EZ,D4EX,D4EY,D4EZ,ET)
          ELSEIF(KFLD .EQ. ML) THEN
C----------- Electric & Magnetic
            CALL MULTIP(IDE,LC,KUASEX,X,Y,Z,EM,QLE,QLS,QFM,QE,QS,RTQ,
     >               XE,XS,QCE,QCS,
     >               E,DE,DDE,D3EX,D3EY,D3EZ,D4EX,D4EY,D4EZ,ET)
            CALL MULTIP(IDB,MG,KUASEX,X,Y,Z,BM,DLE,DLS,DFM,DI,DS,RTB,
     >               XE,XS, CE, CS,
     >               B,DB,DDB,D3BX,D3BY,D3BZ,D4BX,D4BY,D4BZ,BT)
          ENDIF

        ELSEIF(KUASEX .EQ. 20) THEN
```

Note that MULTIP returns the field B and its derivatives dB, ddB, etc., at all X,Y,Z - unlike BENDF that returned B and its derivatives dB, ddB, etc., at all X,Y for Z=0 and needed SYMMED for Z-extrapolation.

• Edit multip.f, see how multipoles are superimposed

EXERCISE #5-a, CONT'ED

- Go to the guide, see 'MULTIPOLE', see where skew angles are introduced. Check in exercise #2 data file.
- Looking in multip.f, find how independent skewing of multipole components works.
- Get the transfer matrix of a quadrupole, and that of its skewed homologue.
- Same for a sextupole, second order

EXERCISE #5-b - 'FFAG' is a sophisticated n-tuple dipole simulation. Let's see how it works

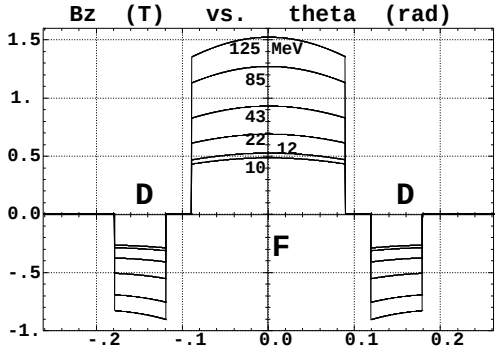


Figure 4: Field on closed orbits, hard edge model.

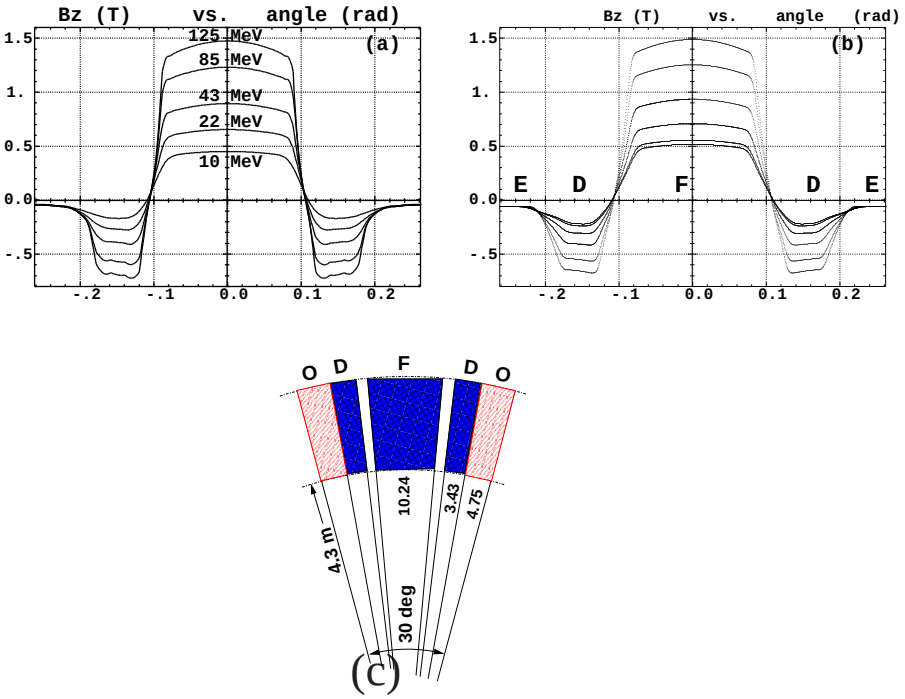


Figure 5: Comparison of magnetic field along closed orbits in the case of, **(a)** : TOSCA 3-D map representative of (former) KEK 150 MeV FFAG (now at Kyushu Univ.) and, **(b,c)** : a "3+2"-dipole geometrical model, "EDFDE", featuring about 700 G field in the "O" dipole regions.

• Let's take a look to the way that works :

• Edit chamc.f, find the following sequence :

```
        ELSEIF(KUASEX .EQ. 27) THEN
C--------- FFAG
C  Equivalence (X,ANGLE), (Y,RADIUS)
          IF    (KAN.EQ.0) THEN
C  Compute FFAG field and derivatives from analytical model
            CALL FFAGFA(IDB,X,Y,
     >                          BZ0)
          ELSEIF(KAN.EQ.1) THEN
C  Compute FFAG field  and derivatives from flying field-mesh
            CALL FFAGF(X,Y,
     >                   DA,DR,FTAB)
            AAA = ZERO
            RRR = ZERO
            CALL INTPLF(Y,AAA,RRR,DA,DR,FTAB,IRD,
     >                                          BZ0)
          ELSE
            STOP '*** SBR CHAMC. No such field case'
          ENDIF
```

FFAGF[A] returns the field B and some derivatives at all X,Y in median plane.

Computation of B and derivatives dB, ddB, etc., at all X,Y,Z further require SYMMED and DBDXYZ. Look into these routines so to figure out what it does.

• Edit ffagi.f, find "entries" ffagf, ffgafa, look in there see what goes on...

EXERCISE #5-b, radial FFAG magnet, CONT'ED

• Go to 'MATRIX' directory, Get the transfer matrix of the DFD triplet FFAG magnet, using various 'KIRD', 'resol'

• Go to 'plotField' directory, plot the field along a closed orbit, using either zpop, or gnuplot.

• Go to 'searchCO' directory, use fortran code 'searcCO_HV' to find closed orbits. Plots these and B field on these.

Note : input zgoubi.dat is in zgoubi_searchCO-In.dat, output files generated by searchCO_HV are zgoubi_searchCO-Out.dat and others.

• Go to 'acceleration' directory, run a cycle. Figure out how the acceleration cycle is accounted for from zgoubi.freqLaw.In.

Note : I store in b_zgoubi.fai rather than in zgoubi.fai, that makes smaller file, and faster run. Hence I use zpop to plot, here.

Plot (Y,Y') what do we see ?

Plot (E,Y), make sure it behaves as expected. Hint :

$$r/r_0 = (p/p_0)^{1/1+k}$$

.

Plot (turn#,Energy).

Plot (Z,Z') phase space, check the dmping law ($\sim 1/r$?).

4.3 Calculation of $\vec{B}$ from Field Maps

4.3.1 1-D Axial Map, with Cylindrical Symmetry

Let B_i be the value of the longitudinal component $B_X(X, r = 0)$ of the field $\vec{B}$, at node i of a uniform mesh that defines a 1-D field map along the symmetry X-axis, while $B_r(X, r = 0)$ is assumed to be zero $(r = (Y^2 + Z^2)^{1/2})$. The field component $B_X(X, r = 0)$ is calculated by a polynomial interpolation of the fifth degree in X, using a 5 points grid centered at the node of the 1-D map which is closest to the actual coordinate X of the particle.
The interpolation polynomial is

$$B(X, 0) = A_0 + A_1X + A_2X^2 + A_3X^3 + A_4X^4 + A_5X^5 \tag{4.3.1}$$

and the coefficients A_i are calculated by expressions that minimize the quadratic sum

$$S = \sum_i (B(X, 0) - B_i)^2 \tag{4.3.2}$$

Namely, the source code contains the explicit analytical expressions of the coefficients A_i solutions of the normal equations $\partial S/\partial A_i = 0$.
The derivatives $\dfrac{\partial^n B}{\partial X^n}(X, 0)$ at the actual position X, as involved in eqs. (4.2.1), are then obtained by differentiation of the polynomial (4.3.1), giving

$$\begin{aligned}
\frac{\partial B}{\partial X}(X, 0) &= A_1 + 2A_2X + 3A_3X^2 + 4A_4X^3 + 5A_5X^4 \\
\frac{\partial^2 B}{\partial X^2}(X, 0) &= 2A_2 + 6A_3X + 12A_4X^2 + 20A_5X^3 \\
&\cdots \\
\frac{\partial^5 B}{\partial X^5}(X, 0) &= 120A_5
\end{aligned} \tag{4.3.3}$$

4.3.2 2-D Median Plane Map, with Median Plane Antisymmetry

Let B_{ij} be the value of $B_Z(X, Y, 0)$ at the nodes of a mesh which defines a 2-D field map in the (X, Y) plane while $B_X(X, Y, 0)$ and $B_Y(X, Y, 0)$ are assumed to be zero. Such a map may have been built or measured in either Cartesian or polar coordinates. Whenever polar coordinates are used, a change to Cartesian coordinates (described below) provides the expression of $\vec{B}$ and its derivatives as involved in eq. (4.1.8).
zgoubi provides three types of polynomial interpolation from the mesh (option *IORDRE*); namely, a second order interpolation, with either a 9- or a 25-point grid, or a fourth order interpolation with a 25-point grid (Fig. 6).
If the 2-D field map is built up from simulation, the grid simply aims at interpolating the field at a given point from its 9 or 25 neighbors. If the map results from measurements, the grid also smoothes field measurement fluctuations.

The mesh may be defined in Cartesian coordinates, (Figs. 3A and 3B) or in polar coordinates (Fig. 6C).
The interpolation grid is centered on the node which is closest to the projection in the (X, Y) plane of the actual point of the trajectory.
The interpolation polynomial is

$$B(X,Y,0) = A_{00} + A_{10}X + A_{01}Y + A_{20}X^2 + A_{11}XY + A_{02}Y^2 \tag{4.3.4}$$

in second order, or

$$\begin{aligned} B(X,Y,0) =& A_{00} + A_{10}X + A_{01}Y + A_{20}X^2 + A_{11}XY + A_{02}Y^2 \\ &+ A_{30}X^3 + A_{21}X^2Y + A_{12}XY^2 + A_{03}Y^3 \\ &+ A_{40}X^4 + A_{31}X^3Y + A_{22}X^2Y^2 + A_{13}XY^3 + A_{04}Y^4 \end{aligned} \tag{4.3.5}$$

in fourth order. The coefficients A_{ij} are calculated by expressions that minimize, with respect to A_{ij}, the quadratic sum

$$S = \sum_{ij} (B(X,Y,0) - B_{ij})^2 \tag{4.3.6}$$

The source code contains the explicit analytical expressions of the coefficients A_{ij} solutions of the normal equations $\partial S/\partial A_{ij} = 0$.
The A_{ij} may then be identified with the derivatives of $B(X, Y, 0)$ at the central node of the grid

$$A_{ij} = \frac{1}{i!j!} \frac{\partial^{i+j} B}{\partial X^i \partial Y^j} (0,0,0) \tag{4.3.7}$$

The derivatives of $B(X, Y, 0)$ with respect to X and Y, at the actual point $(X, Y, 0)$ are obtained by differentiation of the interpolation polynomial, which gives (e.g. from (4.3.4) in the case of second order interpolation)

$$\begin{aligned} \frac{\partial B}{\partial X}(X,Y,0) &= A_{10} + 2A_{20}X + A_{11}Y \\ \frac{\partial B}{\partial Y}(X,Y,0) &= A_{01} + A_{11}X + 2A_{02}Y \\ &\text{etc.} \end{aligned} \tag{4.3.8}$$

This allows stepping to the calculation of $\vec{B}(X, Y, Z)$ and its derivatives as described in subsection 4.2.2 (eq. 4.2.3).

Case of polar maps

It is necessary to change from polar map frame (R, α, Z) to the Cartesian moving frame (X, Y, Z). This is done as follows. In second order calculations the correspondence is (we note $B \equiv B_Z(Z=0)$)

$$\begin{aligned}
\frac{\partial B}{\partial X} &= \frac{1}{R}\frac{\partial B}{\partial \alpha}, \quad \frac{\partial B}{\partial Y} = \frac{\partial B}{\partial R} \\
\frac{\partial^2 B}{\partial X^2} &= \frac{1}{R^2}\frac{\partial^2 B}{\partial \alpha^2} + \frac{1}{R}\frac{\partial B}{\partial R} \\
\frac{\partial^2 B}{\partial X \partial Y} &= \frac{1}{R}\frac{\partial^2 B}{\partial \alpha \partial R} - \frac{1}{R^2}\frac{\partial B}{\partial \alpha} \\
\frac{\partial^2 B}{\partial Y^2} &= \frac{\partial^2 B}{\partial R^2} \\
\frac{\partial^3 B}{\partial X^3} &= \frac{3}{R^2}\frac{\partial^2 B}{\partial \alpha \partial R} - \frac{2}{R^3}\frac{\partial B}{\partial \alpha} \\
\frac{\partial^3 B}{\partial X^2 \partial Y} &= \frac{-2}{R^3}\frac{\partial^2 B}{\partial \alpha^2} - \frac{1}{R^2}\frac{\partial B}{\partial R} + \frac{1}{R}\frac{\partial^2 B}{\partial R^2} \\
\frac{\partial^3 B}{\partial X \partial Y^2} &= \frac{2}{R^3}\frac{\partial B}{\partial \alpha} - \frac{2}{R^2}\frac{\partial^2 B}{\partial \alpha \partial R} \\
\frac{\partial^3 B}{\partial Y^3} &= 0
\end{aligned} \tag{4.3.9}$$

In fourth order calculations the relations are the same up to second order, and then

$$\begin{aligned}
\frac{\partial^3 B}{\partial X^3} &= \frac{1}{R^3}\frac{\partial^3 B}{\partial \alpha^3} + \frac{3}{R^2}\frac{\partial^2 B}{\partial \alpha \partial R} - \frac{2}{R^3}\frac{\partial B}{\partial \alpha} \\
\frac{\partial^3 B}{\partial X^2 \partial Y} &= \frac{1}{R^2}\frac{\partial^3 B}{\partial \alpha^2 \partial R} - \frac{2}{R^3}\frac{\partial^2 B}{\partial \alpha^2} - \frac{1}{R^2}\frac{\partial B}{\partial R} + \frac{1}{R}\frac{\partial^2 B}{\partial R^2} \\
\frac{\partial^3 B}{\partial X \partial Y^2} &= \frac{1}{R}\frac{\partial^3 B}{\partial \alpha \partial R^2} + \frac{2}{R^3}\frac{\partial B}{\partial \alpha} - \frac{2}{R^2}\frac{\partial^2 B}{\partial \alpha \partial R} \\
\frac{\partial^3 B}{\partial Y^3} &= \frac{\partial^3 B}{\partial R^3} \\
\frac{\partial^4 B}{\partial X^4} &= \frac{1}{R^4}\frac{\partial^4 B}{\partial \alpha^4} - \frac{8}{R^4}\frac{\partial^2 B}{\partial \alpha^2} + \frac{6}{R^3}\frac{\partial^3 B}{\partial \alpha^2 \partial R} + \frac{3}{R^2}\frac{\partial^2 B}{\partial R^2} - \frac{3}{R^3}\frac{\partial B}{\partial R} \\
\frac{\partial^4 B}{\partial X^3 \partial Y} &= \frac{1}{R^3}\frac{\partial^4 B}{\partial \alpha^3 \partial R} - \frac{3}{R^4}\frac{\partial^3 B}{\partial \alpha^3} + \frac{3}{R^2}\frac{\partial^3 B}{\partial \alpha \partial R^2} - \frac{8}{R^3}\frac{\partial^2 B}{\partial \alpha \partial R} + \frac{6}{R^4}\frac{\partial B}{\partial \alpha} \\
\frac{\partial^4 B}{\partial X^2 2Y^2} &= \frac{1}{R^4}\frac{\partial^2 B}{\partial \alpha^2} - \frac{4}{R^3}\frac{\partial^3 B}{\partial \alpha^2 \partial R} - \frac{2}{R^2}\frac{\partial^2 B}{\partial R^2} + \frac{2}{R^3}\frac{\partial B}{\partial R} + \frac{1}{R^2}\frac{\partial^4 B}{\partial \alpha^2 \partial R^2} + \frac{1}{R}\frac{\partial^3 B}{\partial R^3} \\
\frac{\partial^4 B}{\partial X \partial Y^3} &= \frac{1}{R}\frac{\partial^4 B}{\partial \alpha \partial R^3} - \frac{3}{R^2}\frac{\partial^3 B}{\partial \alpha \partial R^2} + \frac{6}{R^3}\frac{\partial^2 B}{\partial \alpha \partial R} - \frac{6}{R^4}\frac{\partial^4 B}{\partial \alpha^4} \\
\frac{\partial^4 B}{\partial Y^4} &= \frac{\partial^4 B}{\partial R^4}
\end{aligned} \tag{4.3.10}$$

NOTE: If a particle goes beyond the limits of the field map, the field and its derivatives will be extrapolated by means of the same calculations, from the border grid which is the closest to the actual position of the particle. Its flag *IEX* is given the value -1 (see section 7.6.8).

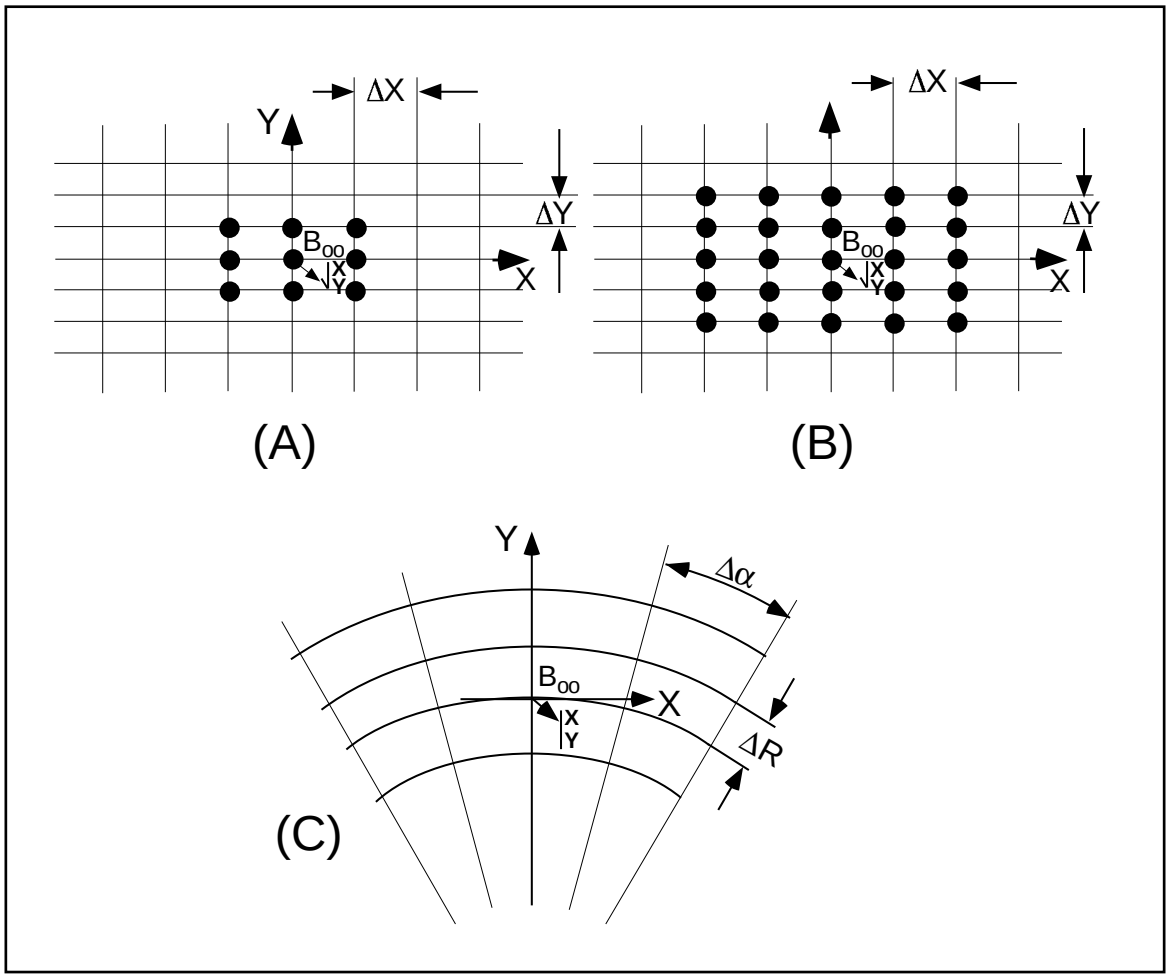


Figure 6: Mesh in the (X, Y) plane in Cartesian coordinates. The grid is centered on the node which is closest to the actual position of the particle.
A: 9-point interpolation grid.
B: 25-point interpolation grid.
C: Mesh in the (X, Y) plane in polar coordinates.

4.3.3 Arbitrary 2-D Map, no Symmetry

The map is supposed to describe the field $\vec{B}(B_X, B_Y, B_Z)$ in the (X, Y) plane at elevation Z_0. It provides the components $B_{X,ij}$, $B_{Y,ij}$, $B_{Z,ij}$ at each node (i, j) of a 2-D mesh.
The value of $\vec{B}$ and its derivatives at the projection (X, Y, Z_0) of the actual position (X, Y, Z) of a particle is obtained by means of a polynomial interpolation from a 3×3 points grid centered at the node (i, j) which is closest to the position (X, Y)

$$B_\ell(X, Y, Z_0) = A_{00} + A_{10}X + A_{01}Y + A_{20}X^2 + A_{11}XY + A_{02}Y^2 \quad (4.3.11)$$

where B_ℓ stands for any of the three components B_X, B_Y or B_Z. Differentiating then gives the derivatives

$$\begin{aligned} \frac{\partial B_\ell}{\partial X}(X, Y, Z_0) &= A_{10} + 2A_{20}X + A_{11}Y \\ \frac{\partial^2 B_\ell}{\partial X \partial Y}(X, Y, Z_0) &= A_{11} \\ &\text{etc.} \end{aligned} \quad (4.3.12)$$

Then follows the procedure of extrapolation from (X, Y, Z_0) to the actual position (X, Y, Z).
No special symmetry is assumed, which allows the treatment of arbitrary field distribution.
Fourth order polynomial interpolation is available upon request (parameter *IORDRE* in keyword data list - see *MAP2D, MAP2D-E*), using the method above based on eq. (4.3.11 developped up to fourth order in X and Y.

4.3.4 Calculation of $\vec{B}$ from 3-D Field Map

The vector field $\vec{B}(X, Y, Z)$ and its derivatives necessary for the calculation of position and velocity of the particle are now defined by means of a 3-D field map, through second degree polynomial interpolation

$$\begin{aligned} B_\ell(X, Y, Z) = A_{000} + A_{100}X + A_{010}Y + A_{001}Z + A_{200}X^2 \\ +A_{020}Y^2 + A_{002}Z^2 + A_{110}XY + A_{101}XZ + A_{011}YZ \end{aligned} \tag{4.3.13}$$

B_ℓ stands for any of the three components, B_X, B_Y or B_Z. By differentiation of B_ℓ one gets

$$\begin{aligned} \frac{\partial B_\ell}{\partial X} &= A_{100} + 2A_{200}X + A_{110}Y + A_{101}Z \\ \frac{\partial^2 B_\ell}{\partial X^2} &= 2A_{200} \end{aligned} \tag{4.3.14}$$

and so on for first and second order derivatives with respect to X, Y or Z.

The interpolation involves a $3 \times 3 \times 3$-point parallelipipedic grid (Fig. 7), the origin of which is positioned at the node of the 3-D field map which is closest to the actual position of the particle.

Let B^ℓ_{ijk} be the value of the — measured or computed — magnetic field at each one of the 27 nodes of the 3-D grid (B^ℓ stands for B_X, B_Y or B_Z), and $B_\ell(X, Y, Z)$ be the value at a position (X, Y, Z) with respect to the central node of the 3-D grid. Thus, any coefficient A_i of the polynomial expansion of B_ℓ is obtained by means of expressions that minimize, with respect to A_i, the sum

$$S = \sum_{ijk} \left(B_\ell(X, Y, Z) - B^\ell_{ijk} \right)^2 \tag{4.3.15}$$

where the indices i, j and k take the values -1, 0 or +1 so as to sweep the 3-D grid. The source code contains the explicit analytical expressions of the coefficients A_{ijk} solutions of the normal equations $\partial S / \partial A_{ijk} = 0$.

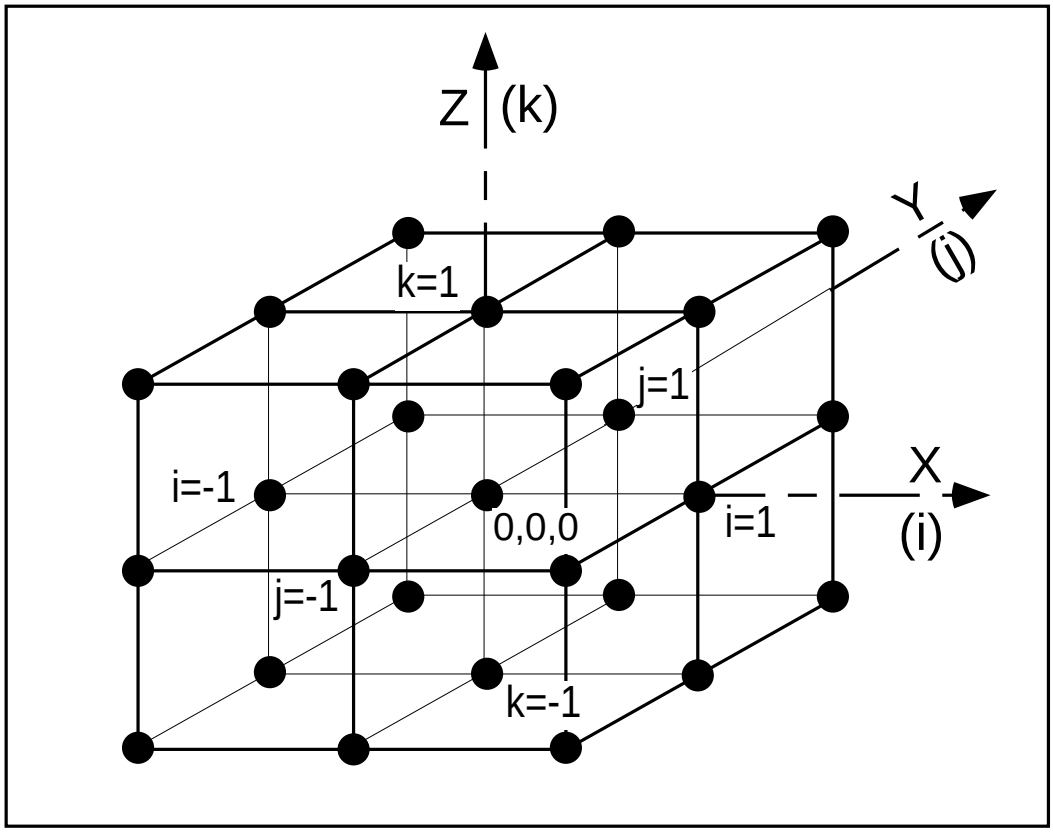


Figure 7: A 3-D 27-point grid is used for interpolation of $\vec{B}$ and its derivatives up to second order. The central node of the grid $(i = j = k = 0)$ is at the closest vicinity of the actual position of the particle.

EXERCISE #6-a - Let's manipulate field maps (1)

We will consider the GSI QD spectrometer. We will use 'TOSCA' keyword.

• Get the data file of concern.

(i) Get the transfer matrix

(ii) Find the focal plane

Do the simulations using either the sole mid-plane field map, or a 3D field map.

• Edit chamk.f, see where field computations take place.

• Track bunches of particles with several different momenta, using OBJET/KOBJ=1.
Check that images are separated at the focal plane (use 'HISTO' or zpop).

EXERCISE #6-b - Let's manipulate field maps (2)

We will consider the former KEK 150 MeV FFAG, now at Kyusu University. We will use 'TOSCA' keyword.

- Make sure you do have the data file of concern.
  (i) Get the closed orbits at 5 different momenta
  (ii) Get the transfer matrices at these various momenta, using OBJET/KOBJ=5.0

- Edit chamk.f, see where field computations take place.

- Track bunches of particles with several different momenta, using OBJET/KOBJ=8.

Check periodicity of the DFD triplet cell.

4.4 Calculation of $\vec{E}$ and its derivatives

zgoubi calculates $\vec{E}(X,Y,Z)$ and its derivatives in various ways, depending whether field maps or analytical representations of optical elements are used. The basic means are the following [4].

4.4.1 Extrapolation from 1-D axial field map

A cylindrically symmetric field can be described by an axial 1-D field map of its longitudinal component $E_X(X, r = 0)$ $(r = (Y^2 + Z^2)^{1/2})$, while the radial component $E_r(X, r = 0)$ is assumed to be zero (e.g. in *ELREVOL*). $E_X(X, r = 0)$ is obtained at any point along the X-axis by a polynomial interpolation from the map mesh (see section 4.3.1). Then the field components $E_X(X,r)$, $E_r(X,r)$ at the position of the particle, (X,r) are obtained from Taylor expansions to the fifth order in r (hence, up to the fifth order derivative $\frac{\partial^5 E_X}{\partial X^5}(X,0)$), assuming cylindrical symmetry

$$\begin{aligned} E_X(X,r) &= E_X(X,0) - \frac{r^2}{4}\frac{\partial^2 E_X}{\partial X^2}(X,0) + \frac{r^4}{64}\frac{\partial^4 E_X}{\partial X^4}(X,0) \\ E_r(X,r) &= -\frac{r}{2}\frac{\partial E_X}{\partial X}(X,0) + \frac{r^3}{16}\frac{\partial^3 E_X}{\partial X^3}(X,0) - \frac{r^5}{384}\frac{\partial^5 E_X}{\partial X^5}(X,0) \end{aligned} \tag{4.4.1}$$

By differentiation with respect to X and r, up to the second order, these expressions provide the derivatives of $\vec{E}(X,r)$. Finally a conversion from the (X,r) coordinates to the (X,Y,Z) Cartesian coordinates of **zgoubi** is performed, thus providing the $\partial^{i+j+k}\vec{E}/\partial X^i \partial Y^j \partial Z^k$ in eqs. (4.1.12).

4.4.2 1-D (Axial) Analytical E-Field Models and Extrapolation

This procedure assumes cylindrical symmetry with respect to the X-axis. The longitudinal field component $E_X(X, r = 0)$ $(r = (Y^2 + Z^2)^{1/2})$, along this axis are derived from differentiation of an adequate model of the electrostatic potential $V(X)$ (e.g. in *EL2TUB, UNIPOT*). The longitudinal and radial field components $E_X(X,r)$, $E_r(X,r)$ and their derivatives off-axis $\frac{\partial^{i+j} E_X}{\partial X^i \partial r^j}$ and $\frac{\partial^{i+j} E_r}{\partial X^i \partial r^j}$ are obtained by Taylor expansions to the fifth order in r assuming cylindrical symmetry (see eq. (4.4.1)), and then transformed to the (X,Y,Z) Cartesian

frame of **zgoubi** in order to provide the derivatives $\partial^{i+j+k}\vec{E}/\partial X^i \partial Y^j \partial Z^k$ needed in eq. (4.1.12).

4.4.3 2-D Analytical E-Field Models and Extrapolation

Several optical elements such as *WIENFILT* (that uses the *BEND* procedures) have their field defined from the expression of the field and derivatives in the median plane. 3-D extrapolation of these off the median is drawn from Taylor expansions.

4.4.4 3-D Analytical models of fields

Various electrostatic or magneto-electrostatic elements, e.g., *ELMULT*, *EBMULT*), have their $\vec{E}$ field components, namely E_X, E_Y, E_Z, as well as the derivatives with respect to X, Y or Z, derived from analytical expressions drawn from models of the potential $V(X, Y, Z)$.

Multipoles And Skewed Multipoles

A right electric multipole is considered to have the same effect as the equivalent skewed magnetic multipole. Therefore, calculation of the right electric or electro-magnetic multipoles (*ELMULT, EBMULT*) uses the same eq. (4.2.5) together with the rotation process as described in section 4.2.6. The same method is used, for arbitrary rotation of arbitrary multipole component around the X-axis.

4.5 Calculation Of $\vec{E}$ From Field Maps

1-D axial map, with cylindrical symmetry

The only type of field map treated in the actual version is the 1-D axial map, with cylindrical symmetry. The same procedure as for the case of magnetic fields is involved (see section 4.3.1).

5 SPIN TRACKING [8]

The depolarization of a particle beam travelling in a magnetic field $\vec{b}$ takes its origin in the spin precession undergone by each particle. This motion of the spin $\vec{S}$ is governed by the Thomas-BMT first order differential equation [9]

$$\frac{d\vec{S}}{dt} = \frac{q}{m}\,\vec{S} \times \vec{\Omega} \tag{5.1}$$

where

$$\vec{\Omega} = (1+\gamma G)\vec{b} + G(1-\gamma)\vec{b}_{/\!/} \tag{5.2}$$

q, m, γ and G are respectively the charge, mass, Lorentz relativistic factor, and anomalous magnetic moment of the particle. $\vec{b}_{/\!/}$ is the component of $\vec{b}$ which is parallel to the velocity $\vec{v}$ of the particle.
These equations are normalized by introducing the same notation as previously. Let $b = \parallel \vec{b} \parallel$ and $v = \parallel \vec{v} \parallel$; $ds = vdt$ is the differential path, $\frac{\gamma m v}{q} = B\rho$ is the rigidity of the particle; $\vec{S}' = \frac{d\vec{S}}{ds} = \frac{1}{v}\frac{d\vec{S}}{dt}$ is the derivative of the spin with respect to the path.
Introducing also $\vec{B} = \frac{\vec{b}}{B\rho}$, $\vec{B}_{/\!/} = \frac{\vec{b}_{/\!/}}{B\rho}$ and

$$\vec{\omega} = \frac{\vec{\Omega}}{B\rho} = (1+\gamma G)\vec{B} + G(1-\gamma)\vec{B}_{/\!/} \tag{5.3}$$

eq. (5.1) can be re-written in a normalized way

$$\vec{S}' = \vec{S} \times \vec{\omega} \tag{5.4}$$

This equation is then solved in the same way as the reduced Lorentz equation (4.1.2). From the values of the magnetic factor $\vec{\omega}(M_0)$ and the spin $\vec{S}(M_0)$ of the particle at position M_0 of its trajectory, the spin $\vec{S}(M_1)$ at position M_1, following a displacement Δs (fig. 2), is obtained from truncated Taylor expansion

$$\vec{S}(M_1) \approx \vec{S}(M_0) + \frac{d\vec{S}}{ds}(M_0)\,\Delta s + \frac{d^2\vec{S}}{ds^2}(M_0)\frac{\Delta s^2}{2} + \frac{d^3\vec{S}}{ds^3}(M_0)\frac{\Delta s^3}{3!} + \frac{d^4\vec{S}}{ds^4}(M_0)\frac{\Delta s^4}{4!} \tag{5.5}$$

The derivatives $\vec{S}^{(n)} = \frac{d^n\vec{S}}{ds^n}$ of $\vec{S}$ at M_0 are obtained by differentiating eq. (5.4)

$$\begin{aligned}
\vec{S}' &= \vec{S} \times \vec{\omega} \\
\vec{S}'' &= \vec{S}' \times \vec{\omega} + \vec{S} \times \vec{\omega}' \\
\vec{S}''' &= \vec{S}'' \times \vec{\omega} + 2\vec{S}' \times \vec{\omega}' + \vec{S} \times \vec{\omega}'' \\
\vec{S}'''' &= \vec{S}''' \times \vec{\omega} + 3\vec{S}'' \times \vec{\omega}' + 3\vec{S}' \times \vec{\omega}'' + \vec{S} \times \vec{\omega}'''
\end{aligned} \tag{5.6}$$

where the derivatives $\vec{\omega}^{(n)}$ are obtained from eq. (5.3).
The last point consists in getting $\vec{B}_{/\!/}$ and its derivatives. This can be done in the following way. Let $\vec{u} = \dfrac{\vec{v}}{v}$ be the normalized velocity of the particle, then,

$$\begin{aligned}
\vec{B}_{/\!/} &= (\vec{B}\cdot\vec{u})\,\vec{u} \\
\vec{B}'_{/\!/} &= (\vec{B}'\cdot\vec{u} + \vec{B}\cdot\vec{u}')\,\vec{u} + (\vec{B}\cdot\vec{u})\,\vec{u}' \\
\vec{B}''_{/\!/} &= (\vec{B}''\cdot\vec{u} + 2\vec{B}'\cdot\vec{u}' + \vec{B}\cdot\vec{u}'')\,\vec{u} + 2(\vec{B}'\cdot\vec{u} + \vec{B}\cdot\vec{u}')\,\vec{u}' + (\vec{B}\cdot\vec{u})\,\vec{u}'' \\
&\text{etc.}
\end{aligned} \tag{5.7}$$

The quantities $\vec{u}$, $\vec{B}$ and their n-th derivatives as involved in these equations are picked up from eqs. (4.1.7, 4.1.8).

Where spin tracking stands : Looking into subroutine spntrk.f for Eq. 5.6 series

- Please edit spntrk.f

```
      SUBROUTINE SPNTRK(IT,DS)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CDF/ IES,LF,LST,NDAT,NRES,NPLT,NFAI,NMAP,NSPN,NLOG
      COMMON/CHAVE/ B(5,3),V(5,3),E(5,3)
      COMMON/CONST/ CL9,CL ,PI,RAD,DEG,QE ,AMPROT, CM2M
      COMMON/PTICUL/ AM,Q,G,TO
      COMMON/RIGID/ BORO,DPREF,DP,BR
      INCLUDE "MAXTRA.H"
      COMMON/SPIN/ KSPN,KSO,SI(4,MXT),SF(4,MXT)
      COMMON/VITES/ U(6,3),DBR(6),DDT(6)
       DIMENSION BU(4,4), BP(4,3), O(4,3), S(5,3)

      PM = BR*CL9/AM *Q
      GG = G * SQRT( 1.D0+PM*PM )
      GP = 1.D0+ GG
      GM = G-GG

CALCUL B.U, B.U', B.U'',B.U''', B'.U, B'.U',B'U'', B''.U, B''.U', B'''.U
      DO 1 IDB=1,4
        DO 1 IDU=1,5-IDB
          TEMP = 0.D0
          DO 2 I=1,3
            TEMP = TEMP + B(IDB,I)*U(IDU,I)
 2        CONTINUE
          BU(IDB,IDU) = TEMP
 1    CONTINUE

C       n              n    n           n
CALCUL d B-parallele/ds  = d ((B.U)U)/ds
      DO 5 I=1,3
        BP(1,I) =  BU(1,1) * U(1,I)
        DBU = BU(2,1)+BU(1,2)
        BP(2,I) = DBU * U(1,I) + BU(1,1) * U(2,I)
        D2BU = BU(3,1) + 2.D0*BU(2,2) + BU(1,3)
        BP(3,I) = D2BU*U(1,I) + 2.D0*DBU*U(2,I) + BU(1,1)*U(3,I)
        D3BU = BU(4,1) + 3.D0*( BU(3,2) + BU(2,3) ) + BU(1,4)
        BP(4,I) = D3BU*U(1,I) + 3.D0*( D2BU*U(2,I)
     >   + DBU*U(3,I) ) + BU(1,1)*U(4,I)
 5    CONTINUE

CALCUL Omega
      DO 6 ID=1,4
        O(ID,1) = GP*B(ID,1) + GM*BP(ID,1)
        O(ID,2) = GP*B(ID,2) + GM*BP(ID,2)
        O(ID,3) = GP*B(ID,3) + GM*BP(ID,3)
 6    CONTINUE
      S(1,1) = SF(1,IT)
      S(1,2) = SF(2,IT)
      S(1,3) = SF(3,IT)

CALCUL S'=SxO
      S(2,1) = S(1,2)*O(1,3) - S(1,3)*O(1,2)
      S(2,2) = S(1,3)*O(1,1) - S(1,1)*O(1,3)
      S(2,3) = S(1,1)*O(1,2) - S(1,2)*O(1,1)
CALCUL S''=S'xO + SxO'
      S(3,1) = S(2,2)*O(1,3) - S(2,3)*O(1,2)
     >       + S(1,2)*O(2,3) - S(1,3)*O(2,2)
      S(3,2) = S(2,3)*O(1,1) - S(2,1)*O(1,3)
     >       + S(1,3)*O(2,1) - S(1,1)*O(2,3)
      S(3,3) = S(2,1)*O(1,2) - S(2,2)*O(1,1)
     >       + S(1,1)*O(2,2) - S(1,2)*O(2,1)
CALCUL S'''=S''xO + 2*S'xO' + SxO''
      S(4,1) = S(3,2)*O(1,3) - S(3,3)*O(1,2)
     >   + 2.D0*(S(2,2)*O(2,3) - S(2,3)*O(2,2))
     >       + S(1,2)*O(3,3) - S(1,3)*O(3,2)
      S(4,2) = S(3,3)*O(1,1) - S(3,1)*O(1,3)
     >   + 2.D0*(S(2,3)*O(2,1) - S(2,1)*O(2,3))
     >       + S(1,3)*O(3,1) - S(1,1)*O(3,3)
      S(4,3) = S(3,1)*O(1,2) - S(3,2)*O(1,1)
     >   + 2.D0*(S(2,1)*O(2,2) - S(2,2)*O(2,1))
     >       + S(1,1)*O(3,2) - S(1,2)*O(3,1)

CALCUL S''''=S'''xO + 3*S''xO' + 3*S'xO'' + SxO'''
      S(5,1) = S(4,2)*O(1,3) - S(4,3)*O(1,2)
     >   + 3.D0*(S(3,2)*O(2,3) - S(3,3)*O(2,2))
     >   + 3.D0*(S(2,2)*O(3,3) - S(2,3)*O(3,2))
     >       + S(1,2)*O(4,3) - S(1,3)*O(4,2)
      S(5,2) = S(4,3)*O(1,1) - S(4,1)*O(1,3)
     >   + 3.D0*(S(3,3)*O(2,1) - S(3,1)*O(2,3))
     >   + 3.D0*(S(2,3)*O(3,1) - S(2,1)*O(3,3))
     >       + S(1,3)*O(4,1) - S(1,1)*O(4,3)
      S(5,3) = S(4,1)*O(1,2) - S(4,2)*O(1,1)
     >   + 3.D0*(S(3,1)*O(2,2) - S(3,2)*O(2,1))
     >   + 3.D0*(S(2,1)*O(3,2) - S(2,2)*O(3,1))
     >       + S(1,1)*O(4,2) - S(1,2)*O(4,1)

C                                        2
CALCUL S(s+ds) = S(s) +S'(s)ds +S''(s)ds /2 +S'''(s)ds
      SF(1,IT)=S(1,1)+(S(2,1)+(S(3,1)/2.D0+(S(4,1)/6.D
     >*DS)*DS)*DS
      SF(2,IT)=S(1,2)+(S(2,2)+(S(3,2)/2.D0+(S(4,2)/6.D
     >*DS)*DS)*DS
      SF(3,IT)=S(1,3)+(S(2,3)+(S(3,3)/2.D0+(S(4,3)/6.D
     >*DS)*DS)*DS
      AN=SQRT(SF(1,IT)*SF(1,IT)+SF(2,IT)*SF(2,IT)+SF(3
      SF(4,IT) = AN

C----- NORMALISATION
      SF(1,IT)=SF(1,IT)/AN
      SF(2,IT)=SF(2,IT)/AN
      SF(3,IT)=SF(3,IT)/AN
      SF(4,IT) = 1D0

      RETURN
      END
```

EXERCISE #6 - Let's track spin in a ring

- **This is going to be a long exercise**, we will have to go through many steps, building and checking the problem data file.

- Practical use : some of you may be involved in Neutrino Factory design (that's the interest of EMMA...).

There, tracking muon's spin may be desired.

- We will consider a 3 GeV proton ring here, and crossing of the $\gamma G = 7 - Q_z$ depolarizing resonance.

We excite that resonance thanks to a (small) field defect in the first quadrupole of the structure.

This is in fact the example #4 in the guide, page 244. Take a look !...

• **We need first to find the closed orbit**, this will give the TOF, necessary for further setting up 'CAVITE' data.

- run 1_closedOrbit-FIT.dat, find the closed orbit for the reference particle (that which has dp/p=0).

Note that 'PARTICUL' is included, otherwise we would not get the TOF.

- explain why the closed orbit is not zero.

- Do not forget to check field on closed orbit in various magnets

Responses :

What we get from zgoubi.res :

```
****************************************************************************************************************************
    83  FAISCEAU
0                                          TRACE DU FAISCEAU

                                             1 TRAJECTOIRES

                                  OBJET                                                  FAISCEAU

          D       Y(CM)    T(MR)    Z(CM)    P(MR)    S(CM)        D      Y(CM)   T(MR)   Z(CM)   P(MR)    S(CM)

o   1   1.0000    0.024    0.011    0.000    0.000    0.000      1.0000   0.024   0.011   0.000   0.000  10555.574     1
               Time of flight (mus) :   1.1568444     mass (MeV/c2) :   938.272
****************************************************************************************************************************
```

In conclusion to this first step : From what precedes, under the last 'FAISCEAU', we get TOF = 1.1568444 μs, closed orbit length $C = 10555.574$ cm, and we also get, under 'PARTICUL', $v/c = 0.3043590202$... which convenientely confirms that TOF = C/v !

We also get the closed orbit coordinates : $(Y, T) = (2.38971390E-02, 1.05938758E-02)$ (cm, mrad)

EXERCISE #6, cont'd - Now let's find the tunes. This is a way to check that the optical data are correct. In addition tunes must be accurately known for such gymnastic as neighbouring or crossing a depolarizing resonance.

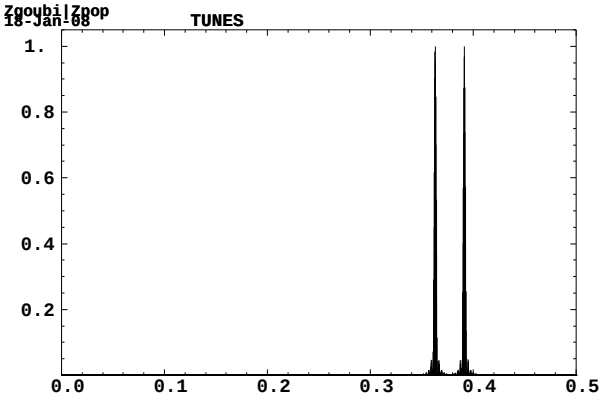
• run 2_tunes.dat. This file only differs from 1_closedOrbit.dat by (i) readance of FIT, addition of 'FAISTORE' for storage of phase space coordinates, (ii) addition of 'REBELOTE' with a few hundred turns to allow fractional tune computation from Fourier analysis, (iii) some more Y amplitude at start and addition of paraxial Z amplitude
• run zpop to get the tunes
• another way to get the tunes is to use MATRIX. Do it. Compare with the previous results.

Responses :

Tunes, from Fourier analysis.

zpop, Menu 8, [8/]13/6 yields
NuY=/[1-] 0.363065 / 0.636935
NuZ=/[1-] 0.391378 / 0.608622

This can be plotted :



Here is what the MATRIX method yields :

```
          TWISS  parameters,  periodicity  of   1  is  assumed :

 Beam  matrix  (beta/-alpha/-alpha/gamma) and  periodic  dispersion  (MKSA units)

    15.176507     2.797647     0.000000     0.000000     0.000000     6.375439
     2.797647     0.581611     0.000000     0.000000     0.000000     0.679629
     0.000000     0.000000     2.049797    -0.440724     0.000000     0.000000
     0.000000     0.000000    -0.440724     0.582613     0.000000     0.000000
     0.000000     0.000000     0.000000     0.000000     0.000000     0.000000
     0.000000     0.000000     0.000000     0.000000     0.000000     0.000000

Betatron  tunes :         NU_Y = 0.63696381         NU_Z = 0.60866886
```

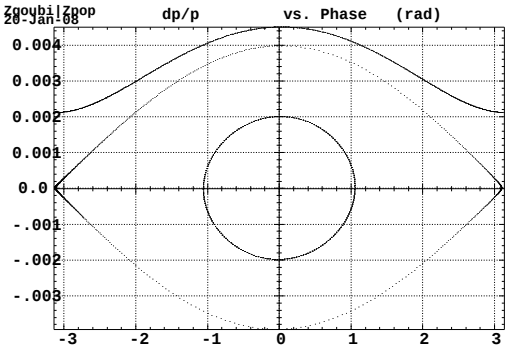
EXERCISE #6, cont'd - Install stationary bucket RF dynamics.

This is done in '3_bucketStatio.dat'. '3_bucketStatio.dat' is obtained from 2_tunes.dat by (i) changing the 'OBJET', (ii) changing 'b_zgoubi.fai #S' to 'b_zgoubi.fai #E', (iii) adding 'CAVITE' at the end, with orbit length from previous exercise.

- run 3_bucketStatio.dat. Plot the longitudinal phase-space.
- Note that, tracking

Y,T,Z,P,L,D = 2.38971390E-02, 1.05938758E-02 .0 0. 0. 1. (quasi synchronous particle, zero horizontal emittance) yields quasi-zero horizontal and longitudinal emittances. This is because Y,T = 2.38971390E-02, 1.05938758E-02 are the closed orbit coordinates.
Do the tracking, plot both phase-spaces, make sure they are quasi-zero.

Responses :

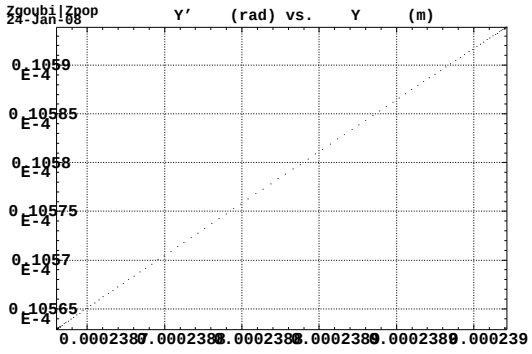


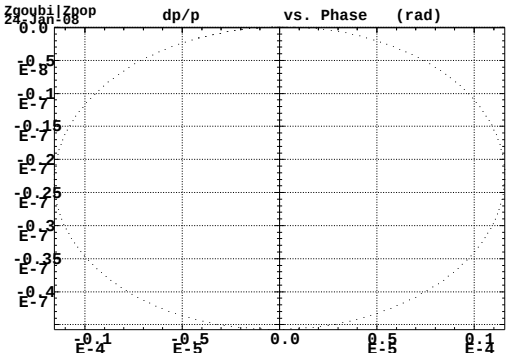
3 particles, (i) a stable one, (ii) a particle (almost) on the separatrix, and (iii) a non-stable one. Their initial coordinates :

Y,T,Z,P,L,D = 2.38971390E-02, 1.05938758E-02 .0 0. 0. 1.0045

Y,T,Z,P,L,D = 2.38971390E-02, 1.05938758E-02 .0 0. 0. 1.0039758

Y,T,Z,P,L,D = 2.38971390E-02, 1.05938758E-02 .0 0. 0. 1.002





Transverse (left) and longitudinal (right) phase spaces, near synchronous conditions.
Note that, orbit length 105.555738 instead of 105.55574 would further reduces longitudinal and transverse emittances, this becomes extreme accuracy !

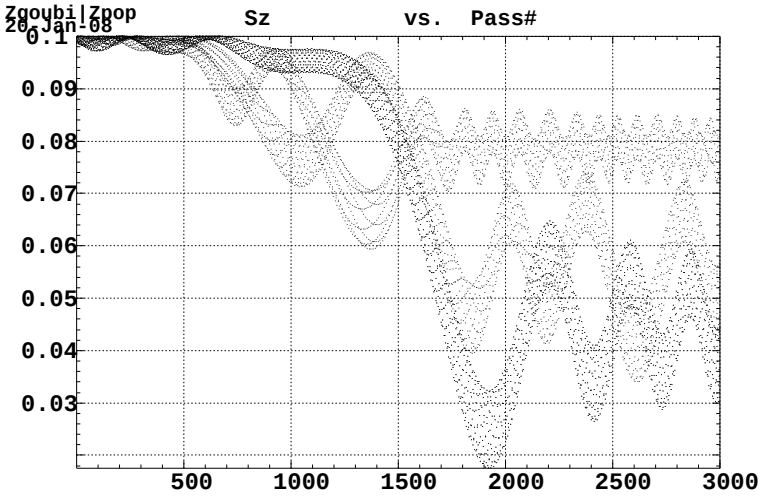
EXERCISE #6, end - Let's cross $\gamma G = 7 - Q_z$.

This is done in '4_resonanceXing.dat'. '4_resonanceXing.dat' is obtained from 3_bucketStatio.dat by (i) adding vertical emittance in 'OBJET', (ii) adding the 'SCALING' instruction, which will allow ramping the magnets and ramping the RF frequency, (iii) switching on spin tracking, using 'SPNTRK', (iv) adding 'SPNPRNL' instruction (e.g., at the end of the .dat file) for storage of spin coordinates, (v) 3000 turns.

Note the presence of a defect in the second quadrupole. This is for exciting the depolarizing resonance.

• run 4_resonanceXing.dat, plot vertical spin component S_z, turn by turn.

Responses :



Spin resonance crossing, 3 particles.

6 SYNCHROTRON RADIATION

zgoubi allows the simulation of two types of synchrotron radiation (SR) related effects namely, on the one hand energy loss by stochastic emission of photon and the ensuing perturbation on particle dynamics and, on the other hand calculation of the radiated spectral-angular energy densities as observed in the lab.

6.1 Energy loss and related dynamical effects [10]

Given a particle wandering in the magnetic field of an arbitrary optical element or field map, **zgoubi** computes the energy loss undergone, and its effect on the particle motion. The energy loss is calculated in a classical manner, by calling upon two random processes that accompany the emission of a photon namely,

- the probability of emission,
- the energy of the photon.

The effects on the dynamic of the emitting particle is either limited to the alteration of the energy, or extended to angular kick effect, following user requested working options ; particle position is supposed not to change upon emission of a photon. These calculations and ensuing dynamics corrections are performed after each integration step. In a practical manner, this means every centimer or tens of centimers in smoothly varying magnetic fields.

Main aspects of the method are developped in the following.

Probability of emission of a photon

Given that the number of photons emitted within a step Δs can be very low (units or fractions of unit)[2] a Poisson probability law

$$p(k) = \frac{\lambda^{-k}}{k!} exp(-k) \tag{6.1.1}$$

is considered. k is the number of photons emitted over a $\Delta\theta$ (circular) arc of trajectory such that, the mean number of photons per radian expresses as[3]

$$\lambda = \frac{20 e r_0}{8\bar{h}\sqrt{3}} \beta^2 B\rho \Delta s \tag{6.1.2}$$

where $r_0 = e^2/4\pi\epsilon_0 m_0 c^2$ is the classical radius of the particle of rest-mass m_0, e is the elementary charge, $\bar{h} = h/2\pi, h$ is the Planck constant, $\beta = v/c$, $B\rho$ is the particle stiffness. λ is evaluated at each integration step from the current values β, $B\rho$ and Δs, then a value of k is drawn by a rejection method [37, routine POIDEV].

[2] For instance, a 1 GeV electron will emit about 20.6 photons per radian; an integration step size $\Delta s = 0.1$ m upon $\rho = 10$ m bending radius results in 0.2 photons per step.

[3] This leads for instance, in the case of electrons, to the classical formula $\lambda/\Delta\theta \approx 129.5\text{E(GeV)}/2\pi \approx \gamma/94.9$.

Energy of the photons

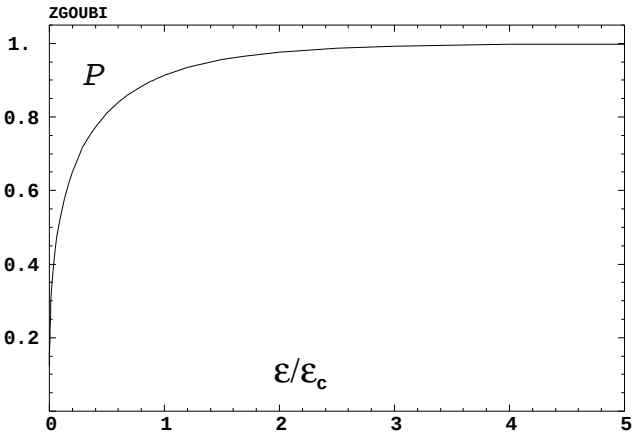
These k photons are assigned energies $\epsilon = h\nu$ at random, in the following way. The cumulative distribution of the energy probability law $p(\epsilon/\epsilon_c)d\epsilon/\epsilon_c$ writes

$$\mathcal{P}(\epsilon/\epsilon_c) = \frac{3}{5\pi}\int_0^{\epsilon/\epsilon_c}\int_{\epsilon/\epsilon_c}^{\infty} K_{5/3}(x)dx \tag{6.1.3}$$

where $K_5/3$ is a modified Bessel function and, $\epsilon_c = \bar{h}\omega_c$ with $\omega_c = 2\pi 3\gamma^3 c/2\rho$ being the critical frequency of the radiation in constant field with bending radius ρ ; ω_c is evaluated at each integration step from the current values γ and ρ, in other words, this energy loss calculation assumes constant magnetic field [4] over the trajectory arc Δs. In the low frequency region $(\epsilon/\epsilon_c \ll 1)$ it can be approximated by

$$\mathcal{P}(\epsilon/\epsilon_c) = \frac{12\sqrt{3}}{5\,2^{1/3}\Gamma(\frac{1}{3})}(\frac{\epsilon}{\epsilon_c})^{1/3} \tag{6.1.4}$$

About 40 values of $\mathcal{P}(\epsilon/\epsilon_c)$ computed from eq. 6.1.3 [38], honnestly spread over a range $\epsilon/\epsilon_c \leq 10$ are tabulated in **zgoubi** source file (see figure). In order to get ϵ/ϵ_c, first a random value $0 < \mathcal{P} < 1$ is generated uniformly, then ϵ/ϵ_c is drawn either by simple inverse linear interpolation of the tabulated values if $\mathcal{P} > 0.26$ (corresponding to $\epsilon/\epsilon_c > 10^{-2}$), or, if $\mathcal{P} < 0.26$ from eq. 6.1.4 that directly gives $\epsilon/\epsilon_c = \left(\frac{5\,2^{1/3}\Gamma(\frac{1}{3})}{12\sqrt{3}\mathcal{P}}\right)^3$ with precision no less than 1% at $\mathcal{P} \to 0.26$.



Cumulative distribution $\mathcal{P}(\epsilon/\epsilon_c)$.

[4] From a practical viewpoint, note that the value of the magnetic field first computed for a one-step push of the particle (eqs. 4.1.1,4.1.7) is next used to obtain ρ and perform SR loss corrections afterwards.

Upon request of SR loss tracking, several optical elements that contain dipole magnetic field component (e.g., *MULTIPOL*) provide a printouot of various quantities related to SR emission, as drawn from classical theoretical expressions, such as for instance,

- energy loss per particle $\Delta E(eV) = \frac{2}{3} r_0 c \gamma^3 B(T) \Delta\theta$, ($B$ is the dipole field, exclusive of any other multipole component or non-linearity in the magnet; $\Delta\theta$ is the total deviation as calculated from B, the magnet length, and the reference rigidity *BORO*(as defined with, e.g., *OBJET*)
- energy $\epsilon_c(eV) = \frac{3\gamma^3 c}{2\rho}\frac{\bar{h}}{e}$, with $\rho =$*BORO*$/B$
- energy of radiated photons $< \epsilon >= \frac{8}{15\sqrt{3}}\epsilon_c$,
- r.m.s. energy of radiated photons $\epsilon_{rms} = 0.5591\epsilon_c$,
- number of radiated photons per particle $N = \Delta E / < \epsilon >$.

This is done in order to facilitate verifications, since on the other hand statistics regarding those values are drawn from the tracking and printed upon use of the dedicated keyword *SYNPRNL*.

Finally, upon user's request as well, SR loss can be limited to particular classes of optical elements, for instance dipole fields alone, or dipole + quadrupole magnets, etc. These tricks are made available in order to permit deeper insight, or easier comparison with other codes, for instance.

Example - Emittance increase in the e^+e^- linear collider beam delivery system

See CEA report.

6.2 Spectral-angular radiated densities [11]

The ray-tracing procedures provide the ingredients necessary for the determination of the electric field radiated by the particle subject to acceleration, as shown in Fig. 8 (section 6.2.1). This allows calculation[5] of spectral-angular densities radiated by particles in magnetic fields (section 6.2.2).

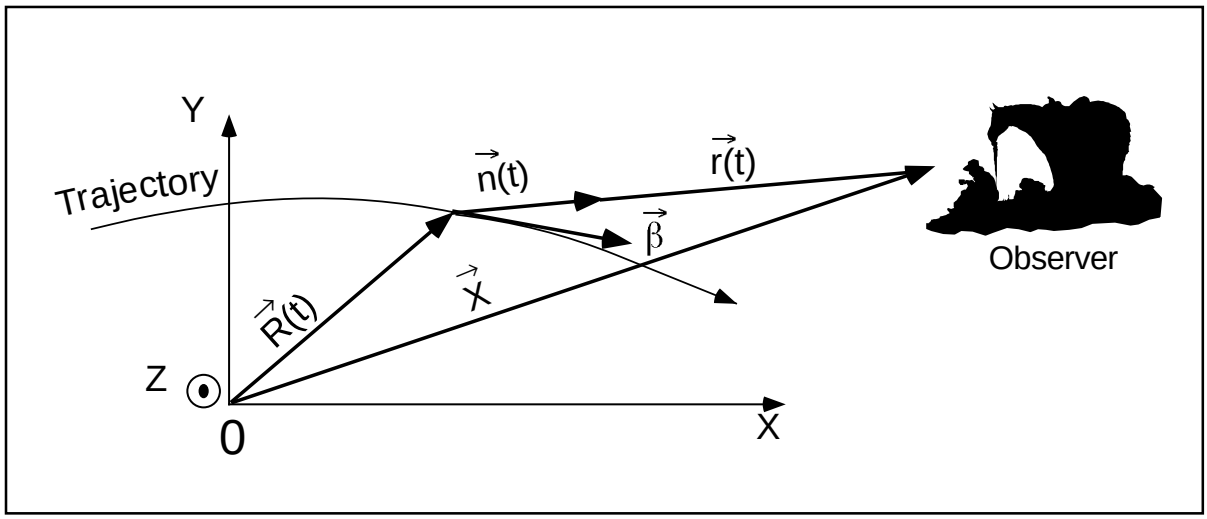


Figure 8: A scheme of the reference frame in **zgoubi** together with the vectors entering in the definition of the electric field radiated by the accelerated particle:
(x, y): horizontal plane; z: vertical axis.
$\vec{R}(t)$ = particle position in the fixed frame (O, x, y, z);
$\vec{X}$ (time-independent) = position of the observer in the (O, x, y, z) frame;
$\vec{r}(t) = \vec{X} - \vec{R}(t)$ = position of the particle with respect to the observer;
$\vec{n}(t)$ = (normalized) direction of observation = $\vec{r}(t)/|\vec{r}(t)|$;
$\vec{\beta}$ = normalized velocity vector of the particle $\vec{v}/c = (1/c)d\vec{R}/dt$.

[5]These procedures are for the moment implemented in the post-processor **zpop**

6.2.1 Calculation of the radiated electric field

The expression for the radiated electric field $\vec{\mathcal{E}}(\vec{n},\tau)$ as seen by the observer in the long distance approximation is [13]

$$\vec{\mathcal{E}}(\vec{n},\tau) = \frac{q}{4\pi\varepsilon_0 c}\,\frac{\vec{n}(t)\times\left[\left(\vec{n}(t)-\vec{\beta}(t)\right)\times d\vec{\beta}/dt\right]}{r(t)\left(1-\vec{n}(t)\cdot\vec{\beta}(t)\right)^3} \tag{6.2.1}$$

where t is the time in which the particle motion is described and τ is the observer time. Namely, when at position $\vec{r}(t)$ with respect to the observer [or as well at position $\vec{R}(t)=\vec{X}-\vec{r}(t)$ in the (O,x,y,z) frame] the particle emits a signal which reaches the observer at time τ, such that $\tau = t + r(t)/c$ where $r(t)/c$ is the delay necessary for the signal to travel from the emission point to the observer, which also leads by differentiation to the well-known relation

$$d\tau/dt = 1-\vec{n}(t)\cdot\vec{\beta}(t) \tag{6.2.2}$$

The vectors $\vec{R}(t)$ and $\vec{\beta}(t) = \frac{v}{c}\vec{u}$ (eq. 4.1.5) that describe the motion are obtained from the ray-tracing (eqs. 4.1.1). The acceleration is calculated from (eq. 4.0.1)

$$d\vec{\beta}/dt = (q/m)\,\vec{\beta}(t)\times\vec{b}(t) \tag{6.2.3}$$

Then, given the observer position $\vec{X}$ in the fixed frame, it is possible to calculate

$$\vec{r}(t) = \vec{X} - R(t) \text{ and } \vec{n}(t) = \vec{r}(t)/|\vec{r}(t)| \tag{6.2.4}$$

The calculation of $\vec{n}-\vec{\beta}$ and $1-\vec{n}\cdot\vec{\beta}$

Owing to computer precision the crude computation of $\vec{n}-\vec{\beta}$ and $1-\vec{n}\cdot\vec{\beta}$ may lead to

$$\vec{n}-\vec{\beta}=0 \text{ and } 1-\vec{n}\cdot\vec{\beta}=0$$

since the preferred direction of observation is generally almost parallel to $\vec{\beta}$ (exactly parallel in the sense of computer precision), while $\beta\approx 1$ as soon as particle energies of a few hundred times the rest mass are concerned.
It is therefore necessary to express $\vec{n}-\vec{\beta}$ and $1-\vec{n}\cdot\vec{\beta}$ in an adequate form for achieving accurate software computation.
The expression for $\vec{n}$ is

$$\begin{aligned}\vec{n} &= (n_x,n_y,n_z) = (\cos\psi\,\cos\phi, \cos\psi\,\sin\phi, \sin\psi)\\ &= \left[1-2(\sin^2\phi/2+\sin^2\psi/2)+4\sin^2\phi/2\sin^2\psi/2, \sin\phi(1-2\sin^2\psi/2), \sin\psi\right]\end{aligned} \tag{6.2.5}$$

where ϕ and ψ are the observation angles, given by

$$\phi = \text{Atg}\left(\frac{r_y}{r_x}\right) \text{ and } \psi = \text{Atg}\left(\frac{r_z}{\sqrt{r_x^2+r_y^2}}\right) \tag{6.2.6}$$

with $\vec{r} = (r_x, r_y, r_z)$, while $\vec{\beta}$ can be written under the form

$$\begin{aligned}\vec{\beta} = (\beta_x, \beta_y, \beta_z) &= \left[\sqrt{(\beta^2 - \beta_y^2 - \beta_z^2)}, \beta_y, \beta_z\right] \\ &= \left[\sqrt{(1 - 1/\gamma^2 - \beta_y^2 - \beta_z^2)}, \beta_y, \beta_z\right] = (1 - a/2 + a^2/8 - a^3/16 + ..., \beta_y, \beta_z) \quad (6.2.7)\end{aligned}$$

where $a = 1/\gamma^2 + \beta_y^2 + \beta_z^2$. This leads to

$$n_x = 1 - \varepsilon_x \text{ and } \beta_x = 1 - \xi_x$$

with

$$\varepsilon_x = 2(\sin^2 \phi/2 + \sin^2 \psi/2) - 4 \sin^2 \phi/2 \sin^2 \psi/2$$

and

$$\xi_x = a/2 - a^2/8 + a^3/16 + ...$$

All this provides, on the one hand,

$$\vec{n} - \vec{\beta} = (-\varepsilon_x + \xi_x, n_y - \beta_y, n_z - \beta_z)\ , \quad (6.2.8)$$

whose components are combinations of terms of the same order of magnitude (ε_x and $\xi_x \sim 1/\gamma^2$ while n_y, β_y, n_z and $\beta_z \sim 1/\gamma$) and, on the other hand,

$$1 - \vec{n} \cdot \vec{\beta} = \varepsilon_x + \xi_x - n_y\beta_y - n_z\beta_z - \varepsilon_x\xi_x\ , \quad (6.2.9)$$

that combines terms of the same order of magnitude ($\varepsilon_x, \xi_x, n_y\beta_y$ and $n_z\beta_z \sim 1/\gamma^2$), plus $\varepsilon_x\beta_x \sim 1/\gamma^4$.
The precision of these expressions is directly related to the order at which the series

$$\xi_x = a/2 - a^2/8 + a^3/16 + ... \qquad (a = 1/\gamma^2 + \beta_y^2 + \beta_z^2)$$

is pushed, however the convergence is fast since $a \sim 1/\gamma^2 \ll 1$.

6.2.2 Calculation of the Fourier transform of the electric field

The Fourier transforms

$$FT_\omega[\vec{\mathcal{E}}(\tau)] = \int \vec{\mathcal{E}}(\tau) e^{-i\omega\tau} d\tau$$

of the σ and π electric field components provide the spectral angular energy density

$$\partial^3 W/\partial\phi\,\partial\psi\,\partial\omega = 2r^2 \left| FT_\omega \left(\vec{\mathcal{E}}(\tau)\right)\right|^2 /\mu_0 c \quad (6.2.10)$$

They are calculated in a regular way, without use of FFT technics, namely from

$$FT_\omega \left[\vec{\mathcal{E}}(\tau)\right] \approx \sum \vec{\mathcal{E}}(\tau_k)\, e^{-i\omega\tau_k}\, \Delta\tau_k \quad (6.2.11)$$

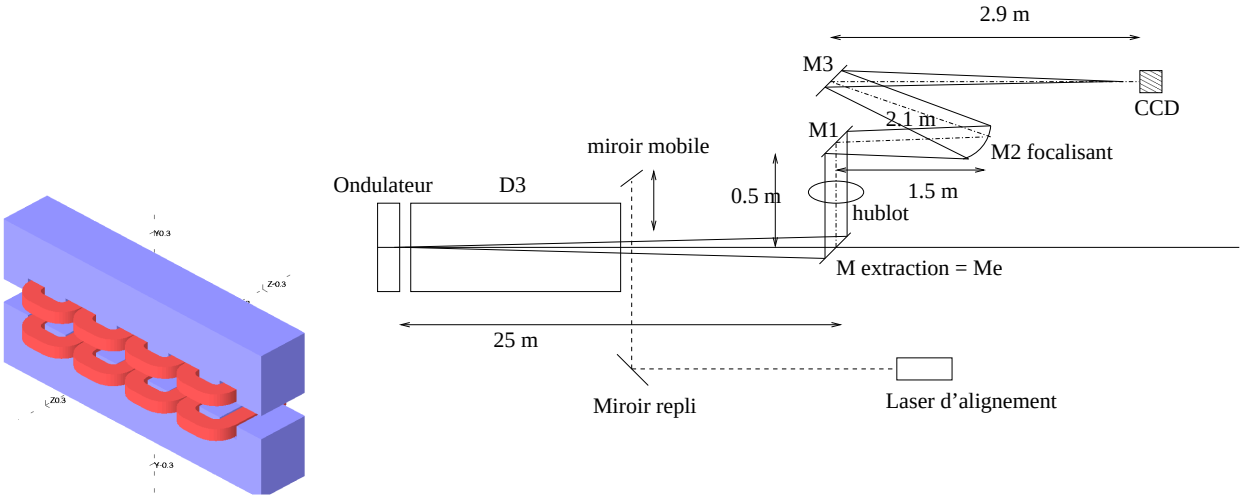
for two reasons. On the one hand, the number of integration steps Δs that define the trajectory (eqs. 4.1.1), is arbitrary and therefore in general not of order 2^n. On the other hand, the integration step defines a constant time differential element $\Delta t_k = \Delta s/\beta c$ which results in the observer differential time element $\Delta\tau_k$, which is also the differential element of the Fourier transform, being non-constant, since both are related by eq. 6.2.2 in which $\vec{\beta}$ and $\vec{n}$ vary as a function of the integration step number k.

Another major point is that $\Delta\tau_k$ may reach drastically small values in the region of the central peak of the electric impulse emitted in a dipole ($1 - \vec{n}(t) \cdot \vec{\beta}(t) \rightarrow 1/2\gamma^2$), whereas the total integrated time $\sum_{k=1}^{N} \Delta\tau_k$ may be several orders of magnitude larger. In terms of the physical phenomenon, the total duration of the electric field impulse as seen by the observer corresponds to the time delay $\sum_{k=1}^{N} \Delta\tau_k$ that separates photons emitted at the entrance of the magnet from photons emitted at the exit, but the significant part of it (in terms of energy density) which can be represented by the width $2\,\tau_c = \dfrac{2(1+\gamma^2\psi^2)^{3/2}\ 2\rho}{3\gamma^3\ c}$ of the radiation peak [14], is a very small fraction of $\sum_{k=1}^{N} \Delta\tau_k$.

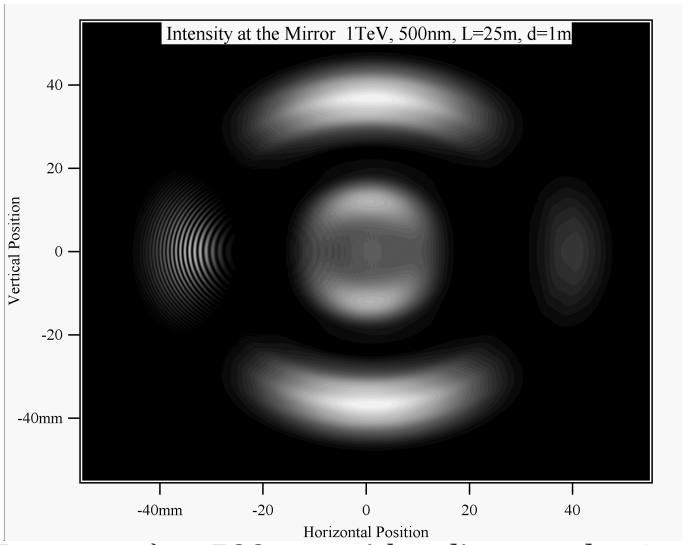
The consequence is that, once again in relation with computer precision, the differential element $\Delta\tau_k$ involved in the computation of eq. 6.2.11 cannot be derived from such relation as $\Delta\tau_k = \sum_{k=1}^{n} \Delta\tau_k - \sum_{k=1}^{n-1} \Delta\tau_k$ but instead must be stored as such beforehand in the couorse of the ray-tracing process.

Example - Design of the SR beam diagnostics installations at LHC [12]

This experince has been fully designed using **zgoubi** , and then checked using SRW (ESRF).



LHC undulator upstream of a long dipole, and the optical system, drawn on that of LEP.



Intensity emitted (horizontal component) by 1 TeV protons, λ = 500 nm, with a distance d = 1 m between the two sources, simulated with Zgoubi (left) and with SRW (right).

7 DESCRIPTION OF THE AVAILABLE PROCEDURES

7.1 Introduction

This chapter gives a detailed description of how the **zgoubi** procedures work, and their associated keywords. It has been split into several sections. Sections 7.2 to 7.5 explain the underlying content and functioning of all available keywords. Section 7.6 is dedicated to the description of some general procedures that may be accessed by means of special data or flags (such as negative integration steps), or through the available keywords (such as multiturn tracking with *REBELOTE*).

7.2 Definition of an Object

The description of the object, *i.e.*, initial coordinates of the beam, must be the first element of the input data to **zgoubi** .

Several types of automatically generated objects are available, as described in the following pages.

MCOBJET: Monte-Carlo generation of a 6-D object

MCOBJET generates a set of up to 10^4 random 6-D initial conditions. It can be used in conjunction with the keyword *REBELOTE*, which moreover allows generating an arbitrarily high number of initial conditions.

The first datum is the reference rigidity (negative value allowed)

$$BORO = \frac{p_0}{q} \text{ (kG.cm)}$$

Depending on the value of the next datum, *KOBJ*, the *IMAX* ($\leq 10^4$) particles have their initial random conditions Y, T, Z, P, X and D (relative momentum) generated on 3 different types of supports, as described below.
Next come the data

$$KY, KT, KZ, KP, KX$$

that specify the type of probability density for the 6 coordinates.
KY, KT, KZ, KP, KX can take the following values:

1. uniform density, $p(x) = 1$ if $-\delta x \leq x \leq \delta x$, $p(x) = 0$ elsewhere,
2. Gaussian density, $p(x) = \frac{1}{\delta x\sqrt{2\pi}} e^{-\frac{x^2}{2\delta x^2}}$,
3. parabolic density, $p(x) = \frac{3}{4\delta x}(1 - \frac{x^2}{\delta x^2})$ if $-\delta x \leq x \leq \delta x$, $p(x) = 0$ elsewhere.

KD can take the following values:

1. uniform density, $p(D) = 1$ if $-\delta D \leq D \leq \delta D$, $p(D) = 0$ elsewhere,
2. exponential density, $p(D) = N_0 \exp(C_0 + C_1 l + C_2 l^2 + C_3 l^3)$ with $0 \leq l \leq 1$ and $-\delta D \leq D \leq \delta D$,
3. $p(D)$ is determined by a kinematic relation, namely, with T = horizontal angle, $D = \delta D * T$.

Next come the central value for the random sorting,

$$Y_0, \quad T_0, \quad Z_0, \quad P_0, \quad X_0, \quad D_0$$

namely, the probability density laws $p(x)$ ($x = Y, T, Z, P$ or X) and $p(D)$ described above apply to the variables $x - x_0$ ($\equiv Y - Y_0$, $T - T_0$, ...) and $D - D_0$ respectively. Negative value for D_0 is allowed (see section 7.6.9).

KOBJ = 1: Random generation of *IMAX* particles in a hyper-window with widths (namely the half-extent for uniform or parabolic distributions ($KY, KT, ... = 1$ or 3), and the r.m.s. width for Gaussian distributions ($KY, KT, ... = 2$))

$$\delta Y, \quad \delta T, \quad \delta Z, \quad \delta P, \quad \delta X, \quad \delta D$$

Then follow the cut-off values, in units of the r.m.s. widths δY, δT, ... (used only for Gaussian distributions, $KY, KT, ... = 2$)

$$N_{\delta Y}, \quad N_{\delta T}, \quad N_{\delta Z}, \quad N_{\delta P}, \quad N_{\delta X}, \quad N_{\delta D}$$

The last data are the parameters

$$N_0, \quad C_0, \quad C_1, \quad C_2, \quad C_3$$

needed for generation of the D coordinate upon option $KD = 2$ (unused if $KD = 1,\ 3$) and a set of three integer seeds for initialization of random sequences,

$$IR1, \quad IR2, \quad IR3 \quad (\text{all} \simeq 10^6)$$

All particles generated by *MCOBJET* are tagged with a (non-S) character, for further statistic purposes (e.g., with *HISTO* and *MCDESINT*).

KOBJ = 2: Random generation of $IY * IT * IZ * IP * IX * ID$ particles (maximum 10^4) in a hyper-grid. The input data are the number of bars in each coordinate

$$IY, \quad IT, \quad IZ, \quad IP, \quad IX, \quad ID$$

the spacing of the bars

$$PY, \quad PT, \quad PZ, \quad PP, \quad PX, \quad PD$$

the width of each bar

$$\delta Y, \quad \delta T, \quad \delta Z, \quad \delta P, \quad \delta X, \quad \delta D$$

the cut-offs, used with Gaussian densities (in units of the r.m.s. widths)

$$N_{\delta Y}, \quad N_{\delta T}, \quad N_{\delta Z}, \quad N_{\delta P}, \quad N_{\delta X}, \quad N_{\delta D}$$

This is illustrated in Fig. 9.

The last two sets of data in this option are the parameters

$$N_0, \quad C_0, \quad C_1, \quad C_2, \quad C_3$$

needed for generation of the D coordinate upon option KD= 2 (unused if KD= 1, 3) and a set of three integer seeds for initialization of random sequences, $IR1$, $IR2$, and $IR3$ (all $\simeq 10^6$). All particles generated by *MCOBJET* are tagged with a (non-S) character, for further statistic purposes (see *HISTO and MCDESINT*).

KOBJ = 3: Distribution of *IMAX* particles inside a 6-D ellipsoid defined by the three sets of data (one set per 2-D phase-space)

$$\alpha_Y,\ \beta_Y,\ \frac{\varepsilon_Y}{\pi},\ N_{\varepsilon_Y}\ [,\ N'_{\varepsilon_Y},\ \text{if } N_{\varepsilon_Y}<0]$$
$$\alpha_Z,\ \beta_Z,\ \frac{\varepsilon_Z}{\pi},\ N_{\varepsilon_Z}\ [,\ N'_{\varepsilon_Z},\ \text{if } N_{\varepsilon_Z}<0]$$
$$\alpha_X,\ \beta_X,\ \frac{\varepsilon_X}{\pi},\ N_{\varepsilon_X}\ [,\ N'_{\varepsilon_X},\ \text{if } N_{\varepsilon_X}<0]$$

where α, β are the ellipse parameters and ε/π the emittance, corresponding to an elliptical frontier $\frac{1+\alpha_Y^2}{\beta_Y}Y^2+2\alpha_Y YT+\beta_Y T^2=\varepsilon_Y/\pi$ (idem for the (Z,P) or (X,D) planes). N_{ε_Y}, N_{ε_Z} and N_{ε_X} are the sorting cut-offs (used only for Gaussian distributions, $KY, KT, ...=2$). The sorting is uniform in surface (for $KY=1$, or $KZ=1$ or $KX=1$) or Gaussian ($KY=2$ or $KZ=2$), and so on, as described above. A uniform sorting has the ellipse above for support. A Gaussian sorting has the ellipse above for r.m.s. frontier, leading to $\sigma_Y=\sqrt{\beta_Y\varepsilon_Y/\pi}$, $\sigma_T=\sqrt{\frac{(1+\alpha_Y^2)}{\beta_Y}\varepsilon_Y/\pi}$, and similar relations for σ_Z, σ_X.

If N_ε is negative, thus the sorting fills the elliptical ring that extends from $|N_\varepsilon|$ to N'_ε (rather than the inner region determined by the N_ε cut-off, as addressed above).

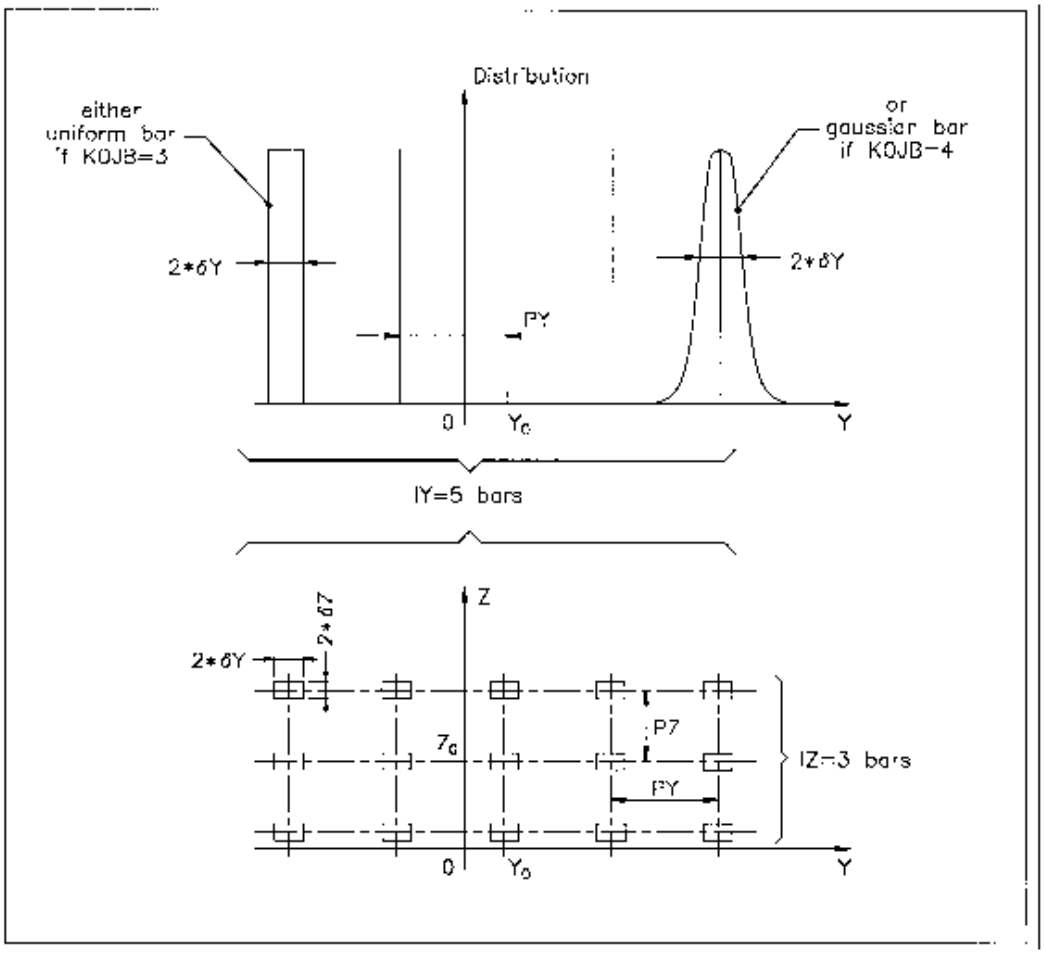


Figure 9: Scheme of the input parameters to *MCOBJET* when *KOBJ* = 3, 4
A: A distribution of the Y coordinate
B: A 2-D grid in (Y, Z) space.

EXERCISE #7 - Generate random objects

• 6.1 - Generate a 2-D gaussian 10000 particle beam with rms sizes $\sigma_Y = 10$ mm, $\sigma_T = 1$ mrad, $\sigma_Z = 3$ mm, $\sigma_P = 0.7$ mrad.
- Check your results using 'HISTO'
- Check your results using zpop, either Menu7 or Menu8

• 6.2 - Make a comb in dp/p space, e.g. dp/p : -0.10, +0.10, step 0.02, gaussian teeth with widths 0.005.
- Check your results using HISTO or zpop.

• 6.2 - Make a grid in Y,Z plane.
- Check your results using HISTO or zpop.

OBJET: Generation of an object

OBJET is dedicated to the determination of the initial coordinates, in several ways.

The first datum is the reference rigidity (a negative value is allowed)

$$BORO = \frac{p_0}{q}$$

At the object, the beam is defined by a set of particles (maximum 10^4) with the initial conditions (Y, T, Z, P, X, D) where D is the relative momentum.
Depending on the value of the next datum *KOBJ*, these initial conditions may be generated in six different ways:

KOBJ = 1: Defines a grid in the Y, T, Z, P, X, D space. One gives the number of points desired,

$$IY, \quad IT, \quad IZ, \quad IP, \quad IX, \quad ID$$

(maximum 41 in each coordinate: $IY \leq 41 \ \ldots ID \leq 41$ and such that $IY*IT*...*ID \leq 10^4$) and the sampling size

$$PY, \quad PT, \quad PZ, \quad PP, \quad PX, \quad PD$$

zgoubi then generates $IY * IT * IZ * IP * IX * ID$ ($\leq 10^4$) initial conditions with the following coordinates

$$\begin{aligned}
&0, \quad \pm PY, \quad \pm 2 * PY, \quad \ldots, \quad \pm IY/2 * PY,\\
&0, \quad \pm PT, \quad \pm 2 * PT, \quad \ldots, \quad \pm IT/2 * PT,\\
&0, \quad \pm PZ, \quad \pm 2 * PZ, \quad \ldots, \quad \pm IZ/2 * PZ,\\
&0, \quad \pm PP, \quad \pm 2 * PP, \quad \ldots, \quad \pm IP/2 * PP,\\
&0, \quad \pm PX, \quad \pm 2 * PX, \quad \ldots, \quad \pm IX/2 * PX,\\
&0, \quad \pm PD, \quad \pm 2 * PD, \quad \ldots, \quad \pm ID/2 * PD,
\end{aligned}$$

In this option relative momenta will be classified automatically for the purpose of the use of *IMAGES* for momentum analysis.
The particles are tagged with an index *IREP* possibly indicating a symmetry with respect to the (X,Y) plane, as explained in option *KOBJ* = 3. If two trajectories have mid-plane symmetry, only one will be ray-traced, while the other will be deduced using the mid-plane symmetries. This is done for the purpose of saving computing time. It may be incompatible with the use of some procedures (e.g. *MCDESINT*, which involves random processes).
The last datum is the reference of the problem (YR, TR, ZR, PR, XR, DR). For instance the reference rigidity is $DR* BORO$, resulting in the rigidity of a particle of initial condition $I * PD$ to be $(DR + I * PD) * BORO$.

KOBJ = 1.1: Same as *KOBJ* = 1 except for the Z symmetry. The initial Z and P conditions are the following

$$0,\ \ \pm PZ,\ \ \pm 2 * PZ,\ \ \ldots,\ \ \pm(IZ-1) * PZ,$$

$$0,\ \ \pm PP,\ \ \pm 2 * PP,\ \ \ldots,\ \ \pm(IP-1) * PP,$$

This object results in shorter outputs/CPU-time when studying problems with Z symmetry.

KOBJ = 2: Next data: *IMAX* , *IDMAX* . Initial coordinates are entered explicitly for each trajectory. *IMAX* is the total number of particles (*IMAX* $\leq 10^4$). These may be classified in groups of equal number for each value of momentum, in order to fulfill the requirements of image calculations by *IMAGES*. *IDMAX* is the number of groups of momenta. The following initial conditions defining a particle are specified for each one of the *IMAX* particles

$$Y,\quad T,\quad Z,\quad P,\quad X,\quad D,\quad {}'A'$$

where $D * BORO$ is the rigidity (negative value allowed) and $'A'$ is a (arbitrary) tagging character.

The last record *IEX* (I=1, *IMAX*) contains *IMAX* times either the string “1” (which indicates that the particle will be tracked) or the string “-9” (indicates that the particle should not be tracked).

This option *KOBJ* = 2 may be be useful for the definition of objects including kinematic effects.

KOBJ $= 3$**:** This option allows the reading of initial conditions from an external input file *FNAME*.

The next three data lines are:

IT1, IT2, ITStep

IP1, IP2, IPStep

YF,TF,ZF,PF,SF,DPF,TiF,TAG

YR,TR,ZR,PR,SR,DPR,TiR

InitC

followed by the storage file name *FNAME*.

IT1, IT2, ITStep tell the code to read coordinates of particles number IT1 through IT2 by step ITStep.

IP1, IP2, IPStep tell the code to read coordinates belonging in the sole pass IP1 through IP2 by step IPStep. Indeed, IP2 $>$ IP1 assumes prior filling of *FNAME* in the course of a run (e.g., multiturn tracking) involving the keyword *REBELOTE*.

YF, TF, ZF, PF, SF, DPF, TiF are references added to the values of respectively Y, T, Z, P, S, DP as read from *FNAME*, whereas YR, TR, ZR, PR, SR, DPR, TiR are scaling factors, so that any coordinate $C = Y, T, Z...$ is chenged into CF*C + CR. In addition a flag character TAG allows retaining only particles with identical tagging letter LET, unless TAG='*' in which case it has no selection effect - for instance TAG='S' can be used to retain only secondary particles following in-flight decay simulations.

If InitC$= 1$ ray-tracing starts from the current coordinates$F(J, I)$,
if InitC$= 0$ ray-tracing starts from the initial coordinates $FO(J, I)$ as read from *FNAME*.

The file *FNAME* must be formatted so as to fit the following *FORTRAN* sequence

```
        OPEN (UNIT = NL, FILE = FNAME, STATUS = 'OLD')
 DO 1 I = 1, IMAX
    READ (NL,100) LET (I), IEX(I), (FO(J,I),J=1,6), (F(J,I),J=1,6), I, IREP(I),
     >      LET(I),IEX(I),-1.D0+FO(1,I),(FO(J,I),J=2,MXJ),
     >      -1.D0+F(1,I),F(2,I),F(3,I),
     >      (F(J,I),J=4,MXJ),ENEKI,
     >      ID,I,IREP(I), SORT(I),D,D,D,D,RET(I),DPR(I),
     >      D, D, D, BORO, IPASS, KLEY,LBL1,LBL2,NOEL
 100        FORMAT(1X,
C1   LET(IT),KEX,   1.D0-FO(1,IT),(FO(J,IT),J=2,MXJ),
     1           A1,1X,I2,1P,7E16.8,
C2   1.D0-F(1,IT),(FO(J,IT),J=2,MXJ),
     2         /,3E24.16,
C3   Z,P*1.D3,SAR,     TAR,     DS,
     3         /,4E24.16,E16.8,
C4   KART,  IT,IREP(IT),SORT(IT),X, BX,BY,BZ, RET(IT), DPR(IT),
     4         /,I1,2I6,7E16.8,
C5        EX,EY,EZ, BORO,  IPASS,  KLEY,  (LABEL(NOEL,I),I=1,2),NOEL
     5  /,4E16.8,          I6,1X,  A8,1X,  2A10,               I5)
    1    CONTINUE
```

where the meaning of the parameters (apart from D=dummy real, ID=dummy integer) is the following

LET(I) : one-character string (for tagging)

IEX(I) : flag, see *KOBJ* = 2

FO(1-6,I) : coordinates D, Y, T, Z, P and path length of the particle number I, at the origin. $D * BORO$ = rigidity

F(1-6,I) : idem, at the current position.

IREP is an index which indicates a symmetry with respect to median plane. For instance, if $Z(I+1) = -Z(I)$, then normally $IREP(I+1) = IREP(I)$. Consequently the coordinates of particle $I+1$ will not be obtained from ray-tracing but instead deduced without ray-tracing from those of particle I by simple symmetry. This results in gain of computing time.

KOBJ $= 3$ can be used directly for reading files filled by *FAISCNL*, *FAISTORE*.
If more than 10^4 particles are to be read from a file, use *IMAX* $\leq 10^4$ in conjunction with *REBELOTE*.

KOBJ = 3.1: Same as **KOBJ = 3**, except for the formatting of trajectory coordinate data in *FNAME* which is much simpler, namely, according to the following *FORTRAN* sequence

```
      OPEN (UNIT = NL, FILE = FNAME, STATUS = 'OLD')
 1    CONTINUE
      READ (NL,*,END=10,ERR=99) Y, T, Z, P, S, D
      GOTO 1
10    CALL ENDFIL
99    CALL ERREAD
```

EXERCISE #8 - Using OBJET/KOBJ=3 for determining EMMA beam path in tune diagram

- Imagine how to do that.
- You'll need to run two .dat problems for that. Set up the two data files, run them, plot beam path in tune diagram.

KOBJ = 5: Mostly dedicated to the calculation of first order transfer matrix and various other optical parameters in conjunction with *MATRIX* or with *TWISS*. The input data are the step-sizes

$$PY, \quad PT, \quad PZ, \quad PP, \quad PX, \quad PD$$

The code generates 11 particles

$$0, \quad \pm PY, \quad \pm PT, \quad \pm PZ, \quad \pm PP, \quad \pm PX, \quad \pm PD$$

These values should be small enough, so that the paraxial ray approximation be valid.
The last data are the initial coordinates of the reference trajectory [normally $(YR, TR, ZR, PR, XR, (0, 0, 0, 0, 0, 1)$]. The reference rigidity is $DR * BORO$ (negative value allowed).

KOBJ = 5.1: Same as **KOBJ = 5**, except for an additional data line giving initial beam ellipse parameters $\alpha_Y, \beta_Y, \alpha_Z, \beta_Z, \alpha_X, \beta_X$, for further transport of these using *MATRIX*, or for possible use by the *FIT* procedure.

KOBJ = 6: Mostly dedicated to the calculation of first, second and other higher order transfer coefficients and various other optical parameters, in conjunction with *MATRIX* or with *TWISS*. The input data are the step sizes

$$PY, \quad PT, \quad PZ, \quad PP, \quad PX, \quad PD$$

to allow the building up of an object containing 61 particles. The last data are the initial coordinates of the reference trajectory [normally $(YR, TR, ZR, PR, XR, DR) = (0, 0, 0, 0, 0, 1)$]. The reference rigidity of the beam is $DR * BORO$.

KOBJ = 7: Object with kinematics
The data and functioning are the same as for *KOBJ* = 1, except for the following

- ID is not used,
- PD is the kinematic coefficient, such that for particle number I, the initial relative momentum D_I is calculated from the initial angle T_I following

  $$D_I = DR + PD * T_I$$

  while T_I is in the range

  $$0, \quad \pm PT, \quad \pm 2 * PT, \quad \ldots, \quad \pm IT/2 * PT$$

  as stated under *KOBJ* = 1

KOBJ = 8: Generation of phase-space coordinates on ellipses.
The ellipses are defined by the three sets of data (one set per ellipse)

$$\alpha_Y, \quad \beta_Y, \quad \varepsilon_Y/\pi$$

$$\alpha_Z, \quad \beta_Z, \quad \varepsilon_Z/\pi$$

$$\alpha_X, \quad \beta_X, \quad \varepsilon_X/\pi$$

where α, β are the ellipse parameters and ε/π is the emittance encompassed, corresponding to an ellipse with equation $\frac{1+\alpha_Y^2}{\beta_Y}Y^2 + 2\alpha_Y YT + \beta_Y T^2 = \varepsilon_Y/\pi$ (idem for the (Z, P) or (X, D) planes).
The ellipses are centered respectively on (Y_0, T_0), (Z_0, P_0), (X_0, D_0).
The number of samples per plane is respectively IX, IY, IZ. If that value is zero, the central value above is assigned.

OBJETA: Object from Monte-Carlo simulation of decay reaction [15]

This generator simulates the reactions

$$M_1 + M_2 \longrightarrow M_3 + M_4$$

and then

$$M_4 \longrightarrow M_5 + M_6$$

where M_1 is the mass of the incoming body; M_2 is the mass of the target; M_3 is an outgoing body; M_4 is the rest mass of the decaying body; M_5 and M_6 are decay products. Example:

$$p + d \longrightarrow {}^3\text{He} + \eta$$
$$\eta \longrightarrow \mu^+ + \mu^-$$

The first input data are the reference rigidity

$$BORO = \frac{p_0}{q}$$

an index *IBODY* which specifies the particle to be ray-traced, namely M3 (*IBODY* = 1), M5 (*IBODY* = 2) or M6 (*IBODY* = 3). In this last case, initial conditions for M6 must be generated by a first run of *OBJETA* with *IBODY* = 2; they are then stored in a buffer array, and restored as initial conditions at the next occurrence of *OBJETA* with *IBODY* = 3. Note that **zgoubi** by default assumes positively charged particles.

Another index, *KOBJ* specifies the type of distribution for the initial transverse coordinates Y, Z; namely either uniform (*KOBJ* = 1) or Gaussian (*KOBJ* = 2). The other three coordinates T, P and D are deduced from the kinematic of the reactions.

The next data are the number of particles to be generated, *IMAX* , and the masses involved in the two previous reactions.

$$M_1, \quad M_2, \quad M_3, \quad M_4, \quad M_5, \quad M_6$$

and the kinetic energy T_1 of the incoming body (M_1).
Then one gives the central value of the distribution for each coordinate

$$Y_0, \quad T_0, \quad Z_0, \quad P_0, \quad D_0$$

and the width of the distribution around the central value

$$\delta Y, \quad \delta T, \quad \delta Z, \quad \delta P, \quad \delta D$$

so that only those particles in the range

$$Y_0 - \delta Y \leq Y \leq Y_0 + \delta Y \qquad \ldots \qquad D_0 - \delta D \leq D \leq D_0 + \delta D$$

will be retained. The longitudinal initial coordinate is uniformly sorted in the range

$$-XL \leq X_0 \leq XL$$

The random sequences involved may be initialized with different values of the two integer seeds IR_1 and IR_2 ($\simeq 10^6$).

7.3 Declaration of options

These options allow the control of procedures that affect certain functions of the code. Some options are normally declared right after the object definition (e.g. *SPNTRK* - spin tracking, *MCDESINT* - in-flight decay), others are normally declared at the end of the data pile (e.g. *END* – end of a problem, *REBELOTE* – for tracking more than 10^4p̃articles or for multi-turn tracking, *FIT* – fitting procedure).

BINARY: *BINARY/FORMATTED* data converter

This procedure translates field map data files from “BINARY” to “FORMATTED” – in the *FORTRAN* sense, or the other way.

The keyword is followed, next line, by NF.NCOL ($NF \leq 9$, $NCOL \leq 9$), the number of files to be translated and of data columns in the file. NCOL should be consistant with the following FORTRAN READ statement :

```
READ (unit=ln,*) (X7(I),I=1,NCOL)
```

The first data line in a field map file is a header, and contains $NCOL$ reals as reference X coordinate, mesh step δX, reference Y coordinate, mesh step δY, and three others (see the FMAPW FORTRAN procedure for more datails).
Then follow, line per line, the NF names of the files to be translated.

Iff a file name begins with the prefix “B_” or “b_”, it is presumed “binary”, and hence converted to “formatted”, and given the same name after suppression of the prefix “B_” or “b_”. Conversely, *iff* the file name does not begin with “B_” or “b_”, the file is presumed “formatted” and hence translated to “binary”, and is given the same name after addition of the prefix “B_”.

In its present state, the procedure *BINARY* only supports a limited number of output formatting, e.g. from *TOSCA* magnet code (see keyword *TOSCA*).

END or FIN: End of input data list ; see FIN

The end of a problem, or of a set of several problems stacked in the data file, should be stated by means of the keywords *FIN* or *END*.

Any information following these keywords will be ignored.

FIT: Fitting procedure

The keyword *FIT* allows the automatic adjustment of up to 20 variables, for fitting up to 20 constraints. It has been realized after existing routines used in the matrix transport code BETA [17]. Any physical parameter of any element (*i.e.* keyword) may be varied. Available constraints are, amongst others: any of the 6×6 coefficients of the first order transfer matrix $[R_{ij}]$ as defined in the keyword *MATRIX*, and its horizontal $(R_{11}R_{22} - R_{12}R_{21})$ and vertical $(R_{33}R_{44} - R_{34}R_{43})$ determinants; horizontal and vertical tunes (if periodical structure); any of the $6 \times 6 \times 6$ coefficients of the second order array $[T_{ijk}]$ as defined in *MATRIX* ; any of the 2×4 coefficients of the σ-matrix as defined by

$$[\sigma_{ij}] = \begin{pmatrix} \sigma_{11} & \sigma_{12} & & \\ \sigma_{21} & \sigma_{22} & & \\ & & \sigma_{33} & \sigma_{34} \\ & & \sigma_{43} & \sigma_{44} \end{pmatrix}$$

and any trajectory coordinates $F(J, I)$ as defined in *OBJET* (I = particle number, J = coordinate number = 1 to 6 for respectively D, Y, T, Z,P or S =path length).

Tunes $\nu_{Y,Z}$ and Twiss periodic functions $\beta_{Y,Z}, \alpha_{Y,Z}, \gamma_{Y,Z}$ are adjustable as well; they are defined by identification of the full optical structure transfer matrix $[R_{ij}]$ with the Twiss matrix, following $[R_{ij}] = Icos(2\pi\nu_{Y,Z}) + Jsin(2\pi\nu_{Y,Z})$ wherein $J = \begin{pmatrix} \alpha & \beta \\ -\gamma & -\alpha \end{pmatrix}$.

VARIABLES

The first input data in *FIT* are the number of variables *NV* , and for each one of them, the following parameters

$IR =$ number of the varied element in the structure

$IP =$ number of the physical parameter to be varied in this element

$XC =$ coupling parameter. Normally $XC = 0$. If $XC \neq 0$, coupling will occur (see below)

$DV =$ allowed relative range of variation of the physical parameter IP.

Numbering of the elements (*IR*):

The elements (*DIPOLE, QUADRUPO,* etc.) are numbered following their sequence in the **zgoubi** input data file, for the purpose of the *FIT* procedure. The number of any element just identifies with its position in the data sequence. However, a simple way to get IR is to make a preliminary run: **zgoubi** will then print the whole structure into the file zgoubi.res with all elements numbered.

Numbering of the physical parameters (*IP*):

In the elements *DIPOLE*, *AIMANT* and *EBMULT*, *ELMULT*, *MULTIPOL*, the numbering of the physical parameters just follows their sequence, as it is shown here after for *DIPOLE-M*: the left column below represents the input data, the right one the corresponding numbering to be used for the *FIT* procedure.

Input data	**Numbering for FIT**
DIPOLE-M	
NFACE, IC*,* IL	1, 2, 3
IAMAX, IRMAX	4, 5
B_0, N, B, G	6, 7, 8, 9
AT, ACENT, RM, RMIN, RMAX	10, 11, 12, 13, 14
λ , ξ	15,16
NC , C_0, C_1, C_2, C_3, C_4, C_5 shift	17, 18, 19, 20, 21, 22, 23, 24
ω, θ, R_1, U_1, U_2, R_2	25, 26, 27, 28, 29, 30
etc.	etc.

Parameters in *SCALING* also have a specific numbering, as follows.

Input data	**Numbering for FIT**
SCALING	
IOPT, NFAM	
NAMEF	
NT_1	
$SCL(I)$, $I = 1$, NT_1	10 $[, ..., \ 10 + NT_1]$
$TIM(I)$, $I = 1$, NT_1	10 $[, ..., \ 10 + 2 * NT_1]$
NAMEF	
NT_2	
$SCL(I)$, $I = 1$, NT_2	20 $[, ..., \ 20 + NT_2]$
$TIM(I)$, $I = 1$, NT_2	20 $[, ..., \ 20 + 2 * NT_2]$
...	
etc. up to *NFAM*	etc.

For all other keywords, the parameters are numbered in the following way

Input data	**Numbering for FIT**
KEYWORD	
first line	1, 2, 3,...
second line	10, 11, 12, 13,...
this is a comment	a line of comments is skipped
next line	20, 21, 22,...
and so on...	30, 31, 32, 33,...

The examples of *QUADRUPO* (quadrupole) and *TOSCA* (Cartesian or cylindrical mesh field map) are given below.

Input data	**Numbering for FIT**
QUADRUPO	
IL	1
XL, R_0, B	10, 11, 12
X_E, λ_E	20, 21
NCE , C_0, C_1, C_2, C_3, C_4, C_5	30, 31, 32, 33, 34, 35, 36
X_S, λ_S	40, 41
NCS , C_0, C_1, C_2, C_3, C_4, C_5	50, 51, 52, 53, 54, 55, 56
XPAS	60
KPOS, XCE, YCE, ALE	70, 71, 72, 73
TOSCA	
IC, IL	1, 2
BNORM, X- [, Y-, Z-]NORM	10, 11 [, 12, 13]
TIT	This is text
IX, IY, IZ, *MOD*	20, 21, 22, 23
FNAME	This is text
ID, A, B, C [A', B', C', etc. if $ID \geq 2$]	30, 31, 32, 33 [34, 35, 36 [, 37, 38, 39] if $ID \geq 2$
IORDRE	40
XPAS	50
KPOS, XCE, YCE, ALE	60,61,62,63

Coupled variables (XC)

Coupling a variable parameter to any other parameter in the structure is possible. This is done by giving XC a value of the form $r \cdot pp$ where the integer part r is the number of the coupled element in the structure (equivalent to IR, see above), and the decimal part pp is the number of its parameter of concern (equivalent to IP, see above) (if the parameter number is in the range 1,...,9, then pp must take the form $0p$). For example, $XC = 20 \cdot 01$ is a request for coupling with the parameter number 1 of element number 20 of the structure, while $XC = 20 \cdot 10$ is a request for coupling with the parameter number 10 of element 20.

An element of the structure which is coupled (by means of $XC \neq 0$) to a variable declared in the data list of the *FIT* keyword, needs not appear as one of the *NV* variables in that data list (this would be redundant information).

XC can be either positive or negative. If $XC > 0$, then the coupled parameter will be given the same value as the variable parameter (for example, symmetric quadrupoles in a lens triplet will be given the same field). If $XC < 0$, then the coupled parameter will be given a variation opposite to that of the variable, so that the sum of the two parameters stays constant (for example, an optical element can be shifted while preserving the length of the structure, by coupling together its upstream and downstream drift spaces).

Variation range (DV**)**

For a parameter IP of initial value p, the *FIT* procedure is allowed to explore the range $p(1 \pm DV)$.

IC = type of constraint (see table below).

I, J = constraint (*i.e.* R_{ij}, determinant, tune; T_{ijk}; σ_{ij}; trajectory $\#I$ and coordinate $\#J$)

IR = number of the element in the **zgoubi** input data file, right after which the constraint applies

V = desired value of the constraint

W = weight of the constraint (smaller W for higher weight)

CONSTRAINTS

The next input data in *FIT* are the number of constraints, *NC* , and for each one of them the following parameters.

IC=0 : The coefficients σ_{11} (σ_{33}) = horizontal (vertical) beta values and σ_{22} (σ_{44}) = horizontal (vertical) derivatives ($\alpha = -\beta'/2$) are obtained by transport of their initial values at line start as introduced using for instance *OBJET, KOBJ=5.1*.

IC=0.1 : Twiss functions: $\sigma_{11} = \beta_Y, \sigma_{12} = \sigma_{21} = -\alpha_Y, \sigma_{22} = \gamma_Y$, $\sigma_{33} = \beta_Z, \sigma_{34} = \sigma_{43} = -\alpha_Z, \sigma_{44} = \gamma_Z$; periodic dispersion: $\sigma_{16} = D_Y$, $\sigma_{26} = D'_Y$, $\sigma_{36} = D_Z, \sigma_{46} = D'_Z$, all quantities derived by assuming periodic structure and identifying the first order transfer matrix to its Twiss form.

IC=1, 2 : The coefficients R_{ij} and T_{ijk} are calculated following the procedures described in *MATRIX*, option *IFOC* = 0. The fitting of the $[R_{ij}]$ matrix coefficients or determinants supposes the tracking of particles having initial coordinates sampled as described in *MATRIX* (these particles are normally defined with *OBJET*, *KOBJ* = 5 or 6). The same is true for the T_{ijk} second order coefficients (Initial coordinates normally defined with *OBJET*, *KOBJ* = 6).

IC=3 : If $1 < I <$ *IMAX* then the value of coordinate type J ($J = 1, 6$ for respectively D, Y, T, Z, P, S) of particle number I ($1 < I <$ *IMAX*) is constrained. If $I = -1$ the constraint is the mean value of coordinate of type J.

IC=4 : The coefficients σ_{11} (σ_{33}) = horizontal (vertical) beta values and σ_{22} (σ_{44}) = horizontal (vertical) derivatives ($\alpha = -\beta'/2$) are derived from an ellipse match of the current particle population (as generated for instance using *MCOBJET, KOBJ=3*).

Type of constraint	Parameters defining the constraints				Object definition (recommended)
	IC	I	J	Constraint	
σ-matrix	0	1 - 6	1 - 6	σ_{IJ} ($\sigma_{11} = \beta_Y, \sigma_{12} = \sigma_{21} = \alpha_Y$, etc.)	*OBJET, KOBJ=5, 6*
Periodic (Twiss) coefficients	0.1	1 - 6	1 - 6	σ_{IJ} ($\sigma_{11} = \cos\mu_Y + \alpha_Y \sin\mu_Y$, etc.)	*OBJET, KOBJ=5, 6*
		7	any	Y-tune = $\mu_Y/2\pi$	
		8	any	Z-tune = $\mu_Z/2\pi$	
		9	any	$\cos(\mu_Y)$	
		10	any	$\cos(\mu_Z)$	
First order parameters	1	1 - 6	1 - 6	Transport coeff. R_{IJ}	*OBJET, KOBJ=5*
		7	any	Y-determinant	
		8	any	Z-determinant	
Second order parameters	2	1 - 6	11 - 66	Transport coeff. $T_{I,j,k}$ $(j = [J/10], k = J - 10[J/10])$	*OBJET, KOBJ=6*
Trajectory coordinates	3	1 - IMAX	1 - 6	$F(J, I)$	*OBJET* or *MCOBJET*
		-1	1 - 6	$< F(J, i) >_{i=1,IMAX}$	
		-2	1 - 6	$Sup(\|F(J, i)\|)_{i=1,IMAX}$	
Matched ellipse parameters	4	1 - 6	1 - 6	σ_{IJ} ($\sigma_{11} = \beta_Y, \sigma_{12} = \sigma_{21} = \alpha_Y$, etc.)	*OBJET, KOBJ=8; MCOBJET/KOBJ=3*
Number of particles	5	-1	any	Ratio $N_{survived}/IMAX$	*OBJET* or *MCOBJET*
		1 - 3	any	$N_{through\ \epsilon_{Y,Z,X}}/N_{survived}$	

Table 1: This table shows the constraints available, depending on the values of IC, I and J. [ ] denotes the integer part. When $IC = 3$, I designates the particle number and J the coordinate number (i.e., D, Y, T, Z, P or X).

The fitting of the $[\sigma_{ij}]$ coefficients supposes the tracking of a relevant population of particles within an adequate emittance.

IC=5 : If $I = -1$ then the constraint value is the ratio of particles still on the run. If $I \geq 1$ then the constraint value is the ratio of particles encompassed within a given I-type ($I = 1 - 3$ for respectively $Y,\ Z,\ D$) phase-space surface.

OBJECT DEFINITION

Depending on the type of constraint (see Table), constraint calculations are performed either from transport coefficient calculation and in such case need *OBJET* with either *KOBJ* = 5 or *KOBJ* = 6, or from particle distributions and in this case need object definition using for instance *OBJET* with *KOBJ* = 8, *MCOBJET* with either *KOBJ* = 3.

THE FITTING METHOD [17]

The numerical procedure is a direct sequential minimization of the quadratic sum of all errors (*i.e.*, differences between desired and actual values of the *NC* constraints), each normalized by its specified weight W (the smaller W, the stronger the constraint).

The step sizes for the variation of the physical parameters depend on their initial values, and cannot be accessed by the user. At each iteration, the optimum value of the step size, as well as the optimum direction of variation, is determined for each one of the NV variables. Then follows an iterative global variation of all NV variables, until the minimization fails which results in a next iteration on the optimization of the step sizes.

Let's visit the fitting routines.

There is quite a number of useful constraints, however is not difficult to add even more. That stands in the FUNCTION FF, file ff.f.

```
      FUNCTION FF(IV)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/CONST/ CL9,CL ,PI,RAD,DEG,QE ,AMPROT, CM2M
      PARAMETER (MXV=40)
      COMMON/CONTR/ VAT(MXV),XI(MXV)
      INCLUDE "MAXTRA.H"
      INCLUDE "MAXCOO.H"
      COMMON/FAISC/ F(MXJ,MXT),AMQ(5,MXT),IMAX,IEX(MXT),IREP(MXT)
      COMMON/FINIT/ FINI
      COMMON/OBJET/ FO(MXJ,MXT),KOBJ,IDMAX,IMAXT
      COMMON/UT/ U(6,6), T(6,6,6)
      COMMON/VARY/ NV,IR(MXV),NC,I1(MXV),I2(MXV),V(MXV),IS(MXV),W(MXV),
     >IC(MXV),I3(MXV),XCOU(MXV),CPAR(MXV,7)

      CHARACTER BLANC*1,AST*10
      PARAMETER (BLANC=' ')
C      PARAMETER (AST  ='*')
      PARAMETER (AST  ='improving!')
      PARAMETER (NMAIL  = 1)

      DIMENSION F0P(6,6), F0PD(6,6), F0MD(6,6)

      DIMENSION IC2(MXV)
      SAVE IC2

      DIMENSION RPD(6,6), RMD(6,6)

      M=1
      CALL REMPLI(M)
      Z=0D0
      KK=0
      DO 3 I=1,NC
         ICONT=IC(I)
         ICONT2=IC2(I)
         K=I1(I)
         L=I2(I)
         IF(I3(I) .NE. KK) THEN
            KK=I3(I)
C----------- Compute present value of constraint
            CALL ZGOUBI(1,KK,.FALSE.,.TRUE.)
         ENDIF

         IF     (ICONT .EQ. 0) THEN
C----------- Constrain BEAM matrix [sigma_ij]. Normally used with OBJET, KOBJ=5 or 6
C 11=beta_y, 12=21=-alpha_y, 22=gamma_y ; 3-4 for ._z ; 5-6 for dp-s
           IORD=1

           IF(KOBJ .EQ. 6) IORD=2

           IF    (ICONT2 .EQ. 0) THEN
             CALL COEFFS(1,IORD,U,T,F0P,1)
             VAL= F0P(K,L)

           ELSEIF(ICONT2 .GE. 1) THEN
C------------ Periodic case :  Twiss coefficients or tunes
C  16=periodic dispersion D_y, 26= its derivative D'_y ; 36=D_z, 46=D'_z
C ICONT2 refers to the reference of concern when using OBJET/KOBJ=5.2
             CALL COEFFS(0,IORD,U,T,F0P,ICONT2)
             CALL TUNES(U,F0P,NMAIL,IERY,IERZ,.FALSE.,
     >                                        YNU,ZNU,CMUY,CMUZ)
             IF(K .LE. 6 .AND. L .LE. 6 ) THEN
               VAL= F0P(K,L)
             ELSEIF( K .EQ. 7 ) THEN
C--------------Constraint  Y  tune
               VAL= YNU
             ELSEIF( K .EQ. 8 ) THEN
               VAL= ZNU
             ELSEIF( K .EQ. 9 ) THEN
C--------------Constraint  cos(muY) = trace/2
               VAL= CMUY
             ELSEIF( K .EQ. 10 ) THEN
               VAL= CMUZ
             ENDIF
           ENDIF
         ELSE IF(ICONT .EQ. 1) THEN
C-----------Contraints are first order transport coeffs

           IORD=1
           IF(KOBJ .EQ. 6) IORD=2
           CALL COEFFS(0,IORD,U,T,F0P,1)
           IF(K .LE. 6 .AND. L .LE. 6 ) THEN
             VAL= U(K,L)
           ELSEIF( K .EQ. 7 ) THEN
C-------------Contraint determinant H
             VAL= U(1,1)*U(2,2)-U(1,2)*U(2,1)
           ELSEIF( K .EQ. 8 ) THEN
C-------------Contraint determinant V
             VAL= U(3,3)*U(4,4)-U(3,4)*U(4,3)
           ENDIF

         ELSE IF(ICONT .EQ. 2) THEN
C-----------Contraints are second order transport coeffs

           IORD=2
           IF    (ICONT2 .EQ. 0) THEN
             CALL COEFFS(0,IORD,U,T,F0P,1)
             L1=L/10
             L2=L-10*L1
             VAL= T(K,L1,L2)
           ELSEIF(ICONT2 .EQ. 1) THEN
             CALL COEFFS(0,IORD,U,T,F0P,1)
             CALL MAT2P(RPD,DP)
             CALL TUNES(RPD,F0PD,NMAIL,IERY,IERZ,.TRUE.,
     >                                          YNUP
             CALL MAT2M(RMD,DP)
             CALL TUNES(RMD,F0MD,NMAIL,IERY,IERZ,.TRUE.,
     >                                          YNUM
             DNUYDP = (YNUP-YNUM)/2.D0/DP
             DNUZDP = (ZNUP-ZNUM)/2.D0/DP
             WRITE(*,FMT='(/,34X,''Chromaticities:'',
     >       //,30X,''dNu_y / dp/p = '',G14.8,/,
     >         30X,''dNu_z / dp/p = '',G14.8)') DNUYDP,
           ENDIF
         ELSE IF(ICONT .EQ. 3) THEN
C----------- Constraints on particle coordinates or bund
           IF    (ICONT2.EQ.0) THEN
             IF(K .GT. 0) THEN
C-------------- Constraint is coordinate L of particle K
               VAL=F(L,K)
             ELSEIF(K.EQ.-1) THEN
C------------- Constraint on beam : average value of coor
               VAL=FITLAV(L)
             ELSEIF(K.EQ.-2) THEN
C------------- Constraint on beam : max value of coordina
               VAL=FITLMA(L)
             ELSEIF(K.EQ.-3) THEN
C------------ Constraint : minimze distance between patic
C              List of particles concerned is entered via
               CALL DIST2(L,
     >                    VAL)
             ENDIF
           ELSEIF(ICONT2.EQ.1) THEN
C------------ Constraint on closed orbit :
C              e.g., equal position values  (L=2,4 or oth
C                  at ends of cell
C                    (hence expected constraint value in
             VAL=ABS(F(L,K) - FO(L,K))
             write(*,*) ' val   vi  : ',F(L,K),FO(L,K),
           ELSEIF(ICONT2.EQ.2) THEN
C------------ Constraint on closed orbit :
C              or opposite angle values (L=3,5)
C                  at ends of cell
C                    (hence expected constraint value in
             VAL=ABS(F(L,K) + FO(L,K))
c            write(*,*) ' val   vi  : ',F(L,K),FO(L,K),
           ENDIF
```

```
            ELSE IF(ICONT .EQ. 4) THEN
      C----------- Constraint ellipse parameters
               IF    (K.LE.2) THEN
                 K2=1
               ELSEIF(K.LE.4) THEN
                 K2=2
               ELSEIF(K.LE.6) THEN
                 K2=3
               ENDIF
               CALL LPSFIT(NRES,NOEL,K2,
           >                              EMIT,ALP,BET,XM,XPM)
               IF(K.EQ.L) THEN
                 IF(K.EQ.2*K2) THEN
                   VAL = (1.D0+ALP*ALP)/BET
                 ELSE
                   VAL = BET
                 ENDIF
               ELSE
                 VAL = ALP
               ENDIF
            ELSE IF(ICONT .EQ. 5) THEN
               IF( K .EQ. -1 ) THEN
      C------------- Ratio of iex>0 particles,  (IMAX-Nstopped)/IMAX
                 NLIV = 0
                 DO 5 IT=1,IMAX
       5           IF(IEX(IT).GE.-1) NLIV = NLIV+1
                 NJUMP = NLIV/100
                 IF(NJUMP.EQ.0) NJUMP=1
                 VAL = ((DBLE(NJUMP)*(NLIV/NJUMP))/DBLE(IMAX))
               ELSE IF( K .GE. 1) THEN
                 IF( K .LE. 3) THEN
      C---------------- Ratio N_InLips/IMAX of particles within ph-space ellipse ;
      C                  ellipse surface is entered as parameter in constraint data list
                   CALL ACCEN(K,
           >                         RATIN)
                   VAL=RATIN
      C                  VAL = 1.D0/VAL
      CCCCCCC     ?? remplace VAL = 1.D0/VAL par VAL = 1.D0-VAL dans FF ????
                 ELSEIF( K .LE. 6) THEN
      C---------------- Maximum ratio N_InLips/IMAX  of particles within a
      C                  ph-space ellipse which is found at best in region of rms ellipse ;
      C                  ellipse surface is entered as parameter in constraint data list
                   CALL ACCEP(K-3,
           >                           EMIT,ALP,BET,XM,XPM,NLIV,MXINL)
      C     >                           EMIT,ALP,BET,XM,XPM,NLIV,NINL)
      C---------------- For a follow-up of the optimized ellipse, using zpop (Menu8/14/7/6) ------
      c                  WRITE(88,FMT='(1P,I7,5G12.4,A13)')
      c     >              MXINL,EMIT,ALP,BET,XM,XPM,'  Function FF'
      c                  BACKSPACE(88)
      c                  READ(88,*) DUM
      C-------------------------------------------------------------------------------------------
                   VAL=DBLE(MXINL)/DBLE(IMAX)
      C                  VAL=DBLE(NINL)/DBLE(IMAX)
                 ELSE
                   STOP ' *** Error, FCT FF -> No such constraint 5.K'
                 ENDIF
               ELSE
                 STOP ' *** Error, FCT FF -> No such constraint 5.K'
               ENDIF
            ELSE IF(ICONT .EQ. 6) THEN
      C    constraint rms  emittance

               CALL LPSFIT(NRES,NOEL,K,
           >                              EMIT,ALP,BET,XM,XPM)
               VAL=EMIT
            ENDIF
            Z=Z+((VAL-V(I))/W(I))**2
            VAT(I)=VAL
      3     CONTINUE

            IF(FINI.LE.Z) THEN
      C        WRITE(6,100) CHAR(13),IV,FINI,Z,BLANC
      C100      FORMAT(1H+,A1,I3,1P,2E20.5,1X,A10,$)
            ELSE
      C         WRITE(6,100) CHAR(13),IV,FINI,Z,AST
               FINI=Z
            ENDIF
            FF=Z
            RETURN

            ENTRY FFI(IC2IN,IIC2)
            IC2(IIC2) = IC2IN
            FFI=0.D0
            RETURN
            END
```

EXERCISE #9 - Look for a particular position on the beam path in tune diagram

The reason for doing that might be, for instance, to check beam behavior in the region of that particular (ν_x, ν_z) value.

We will start from Exercise #4 data file (TOF in EMMA, 'exercise4.dat').

• Make sure it runs well (run it, check TOFs).

• Get the tunes on the 13 closed orbits, using 'MATRIX'. Find the two trajectories that encompass $\nu_x = 1/4$.

• Now, we set up the 'FIT' data, progressively. Only retain in the .dat file those two particles that encompass 0.25.

- sightly change Y_0 of first particle. Use FIT to get back to closed orbit
- get the following fit : first trajectory tuned to 0.251
- get the following fit : simultaneously, first trajectory tuned to 0.251, second trajectory tuned to 0.249.

Responses

- Get the tunes on the 13 closed orbits.

  Find the two trajectories that encompass $\nu_x = 1/4$.

  Responses :

  Use KOBJ = 5.13 and MATRIX to get the tunes.

  Figure out what 'KOBJ = 5.13' means.

  See in the fortran where '5.13' is handled

  The two trajectories are :

```
'OBJET'
+5.171103865921708e+01
 5.02
.001 .001 .001 .001 .001 .0001
 -2.84562213E-02 -1.049408E+00  1.0E-04  0.0E+00  0.0E+00  9.70457160E-01 '4'        14.54 MeV
  2.296861E-02    1.144756E+00  1.0E-04  0.0E+00  0.0E+00  1.00000000E+00 'R'        15.00 MeV
```

• Now, we set up the 'FIT' data, progressively. We first try to run FIT with a simple problem :

- 1/ slightly change Y_0 of first particle. 2/ Now, use FIT to get back to closed orbit

It is sufficient to vary Y_0, with constraint '$Y = Y_0$' after 1 turn ('$T = T_0$' will ensue). It yields the following :

```
Data generated by searchCO
'OBJET'
+5.171103865921708e+01
 5.02
.001 .001 .001 .001 .001 .0001
 -2.845012 -1.049408E+00  1.0E-04  0.0E+00  0.0E+00  9.70457160E-01 '4'        14.54 MeV
  2.296861E-02  1.144756E+00  1.0E-04  0.0E+00  0.0E+00  1.00000000E+00 'R'        15.00 MeV
 'PARTICUL'
 0.51099892 1.60217653e-19 0.0 0.0 0.0
      'CHAMBR'
       1
      1  5.0000   5.0000  0.  0.
 'DRIFT'     DD1
 10
  'CHANGREF'                          Allows positionning of the magnet
   0 -0.1111304250932937 1.47531616464
   'MULTIPOL'  QF                     A multipole, whose sole B_dipole and B_quad are non-zero
 ...................
  'CHANGREF'                          Allows positionning of the magnet
   0 0.1111304250932937 1.47531616464
 'DRIFT'     DQ
 3.9
   'CHANGREF'
   0 1.707551921352993 -5.7610304504
   'MULTIPOL'  QD
 ......................
   'CHANGREF'
   0 -1.707551921352993 -5.7610304504
 'DRIFT'     DD2
 9.9
 'FAISCEAU'     #E
'FIT'
1
1 30 0.0 2.    variable is in OBJET (1), Y coord. in first particle (30), non-coupled (0.0), 20\% variation allowed
1
3.1 1 2 13 0 1. 0  constraint is diff. in coordinates (3.1), particle 1 (I=1), coordinate Y (J=1)
 'END'
```

- Now, get the following fit : first trajectory tuned to 0.251

Response :

’OBJET’ and ’FIT’ data :

```
Data generated by searchCO
’OBJET’
+5.171103865921708e+01
 5.02
.001 .001 .001 .001 .001 .0001
 -2.84562213E-02 -1.049408E+00  1.0E-04  0.0E+00  0.0E+00  9.70457160E-01 ’4’        14.54 MeV
  2.296861E-02    1.144756E+00  1.0E-04  0.0E+00  0.0E+00  1.00000000E+00 ’R’        15.00 MeV
 ’PARTICUL’


’FIT’
2
1 30 0.0 .5    variable is in OBJET (1), Y coord. in first particle (30), non-coupled (0.0), 20\% variation
1 35 0.0 .1    variable is in OBJET (1), Y coord. in first particle (30), non-coupled (0.0), 20\% variation
2
3.1 1 2 13 0 1. 0  constraint is diff. in coordinates (3.1), particle 1 (I=1), coordinate Y (J=1)
0.1 7 0 13 0.251 1. 0  constraint is diff. in coordinates (3.1), particle 1 (I=1), coordinate Y (J=1)
```

- Now, get the following fit : simultaneously, first trajectory tuned to 0.251, second trajectory tuned to 0.249.

Response :

’OBJET’ and ’FIT’ data :

```
'OBJET'
+5.171103865921708e+01
 5.02
.001 .001 .001 .001 .001 .0001
 -2.84562213E-02 -1.049408E+00  1.0E-04  0.0E+00  0.0E+00  9.70457160E-01 '4'        14.54 MeV
  2.296861E-02    1.144756E+00  1.0E-04  0.0E+00  0.0E+00  1.00000000E+00 'R'        15.00 MeV


'FIT'
6
1 30 0.0 4.    variable is in OBJET (1), Y coord. in first particle (30), non-coupled (0.0), 20\% variation
1 31 0.0 4.    variable is in OBJET (1), T coord. in first particle (31), non-coupled (0.0), 20\% variation
1 35 0.0 .2    variable is in OBJET (1), D coord. in first particle (30), non-coupled (0.0), 20\% variation
1 40 0.0 4.    variable is in OBJET (1), Y coord. in first particle (40), non-coupled (0.0), 20\% variation
1 41 0.0 4.    variable is in OBJET (1), T coord. in first particle (41), non-coupled (0.0), 20\% variation
1 45 0.0 .2    variable is in OBJET (1), D coord. in first particle (45), non-coupled (0.0), 20\% variation
6
3.1 1  2 13 0     1. 0  constraint is diff. in coordinates (3.1), particle 1 (I=1), coordinate Y (J=2), val
3.1 1  3 13 0     1. 0  constraint is diff. in coordinates (3.1), particle 1 (I=1), coordinate T (J=3), val
0.1 7  0 13 0.251 .01 0  constraint is tunes (7) for MATRIX block #1 (0.1), applies after elmnt # 13
3.1 12 2 13 0     1. 0  constraint is diff. in coordinates (3.1), particle 12 (I=12), coordinate Y (J=2), v
3.1 12 3 13 0     1. 0  constraint is diff. in coordinates (3.1), particle 12 (I=12), coordinate T (J=3), v
0.2 7  0 13 0.249 .01 0  constraint is tunes (7) for MATRIX block #2 (0.2), applies after elmnt # 13
```

GASCAT: Gas scattering

Modification of particle momentum and velocity vector, performed at each integration step, under the effect of scattering by residual gas.

To be documented

MCDESINT: Monte-Carlo simulation of in-flight decay[18]

As soon as *MCDESINT* appears in a structure (normally, after *OBJET* or after *CIBLE*), in-flight decay simulation starts. It must be preceded by *PARTICUL* for the definition of mass M_1 and *COM* lifetime τ_1.
The two-body decay simulated is

$$1 \longrightarrow 2 + 3$$

The decay is isotropic in the center of mass. 1 is the incoming particle, with mass M_1, momentum $p_1 = \gamma_1 M_1 \beta_1 c$ (relative momentum $D_1 = \frac{p_1}{q}\frac{1}{BORO}$ with *BORO*= reference rigidity, see *OBJET*), and position Y_1, Z_1 in the **zgoubi** frame. 2 and 3 are decay products with respective masses and momenta M_2, M_3 and $p_2 = \gamma_2 M_2 \beta_2 c$, $p_3 = \gamma_3 M_3 \beta_3 c$.
The decay length s_1 of particle 1 is related to its center of mass lifetime τ_1 by

$$s_1 = c\tau_1\sqrt{\gamma_1^2 - 1}$$

The path length s up to the decay point is then calculated from a random number $0 < R_1 \leq 1$ by using the exponential decay formula

$$s = -s_1 \ell n R_1$$

After decay, particle 2 will be ray-traced with assumed positive charge, while particle 3 is discarded. Its scattering angles in the center of mass θ^* and ϕ are generated from two other random numbers R_2 and R_3.

ϕ is a relativistic invariant, and θ in the laboratory frame (Fig. 10) is given by

$$\tan\theta = \frac{1}{\gamma_1}\frac{\sin\theta^*}{\frac{\beta_1}{\beta_2^*} + \cos\theta^*}$$

β_2^* and momentum p_2 are given by

$$\begin{aligned}
\gamma_2^* &= \frac{M_1^2 + M_2^2 - M_3^2}{2M_1M_2}\\
\beta_2^* &= \left(1 - \frac{1}{\gamma^2}\right)^{1/2}\\
\gamma_2 &= \gamma_1\gamma_2^*\left(1 + \beta_1\beta_2^*\cos\theta^*\right)\\
p_2 &= M_2\sqrt{\gamma_2^2 - 1}
\end{aligned}$$

Finally, θ and ϕ are transformed into the angles T_2 and P_2 in the **zgoubi** frame, and the relative momentum takes the value $D_2 = \frac{p_2}{q}\frac{1}{BORO}$ (where *BORO*is the reference rigidity, see *OBJET*), while the starting position of M_2 is $Y_2 = Y_1$ and $Z_2 = Z_1$.

The decay simulation by **zgoubi** obeys the following procedures. In optical elements and field maps, after each integration step *XPAS*, the actual path length of the particle, $F(6, I)$, is compared to its limit path length s. If s is passed, then the particle is considered as having

decayed at $F(6,I) - \dfrac{XPAS}{2}$, at a position obtained by a linear translation from the position at $F(6,I)$. [Presumably, the smaller *XPAS*, the smaller the error on position and angles at the decay point].

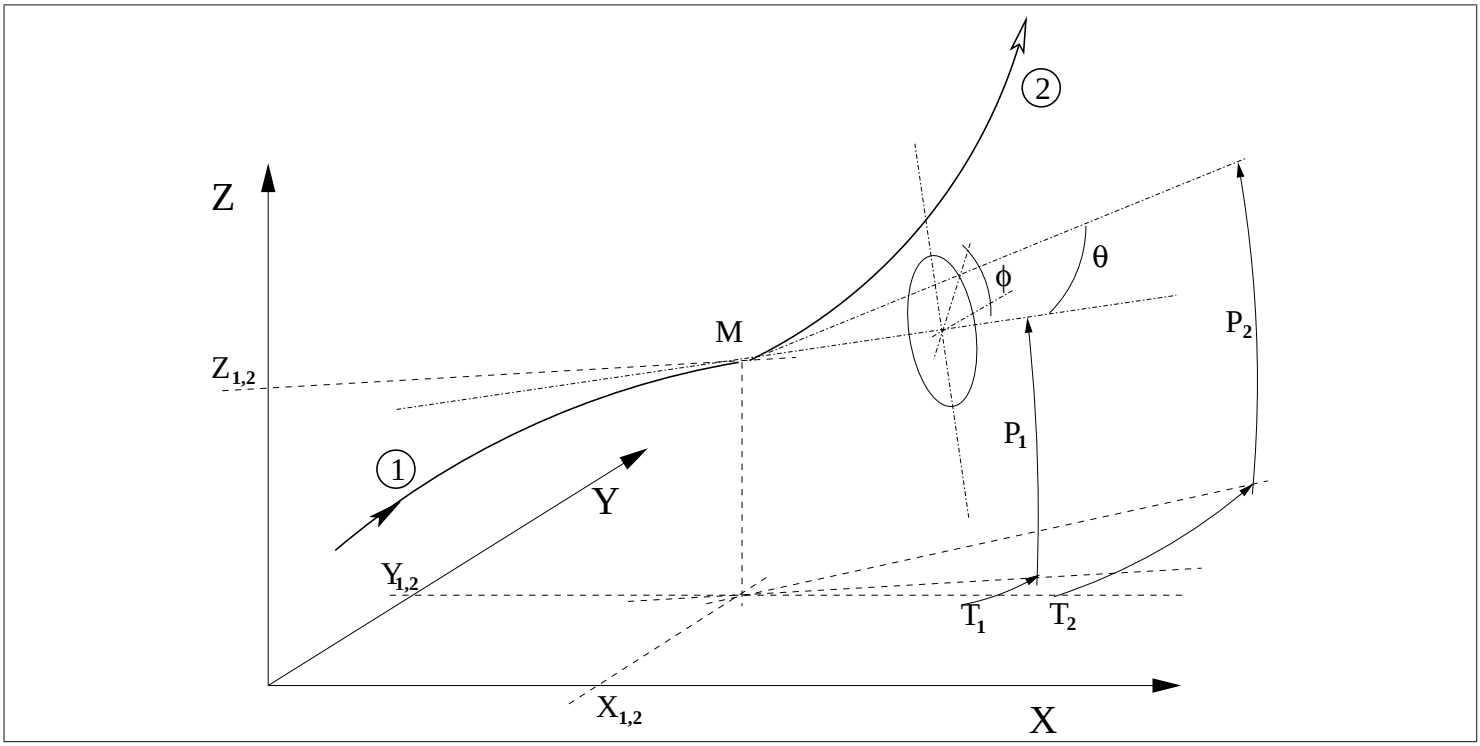


Figure 10: At position $M(X_1, Y_1, Z_1)$, particle 1 decays into 2 and 3; **zgoubi** then calculates the trajectory of 2, while 3 is discarded. θ and ϕ are the scattering angles of particle 2 relative to the direction of the incoming particle 1; they transform to T_2 and P_2 in **zgoubi** frame.

In *ESL* and *CHANGREF*, $F(6,I)$ is compared to s at the end of the element. If the decay occurs inside the element, the particle is considered as having decayed at its actual limit path length s, and its coordinates at s are recalculated by translation.

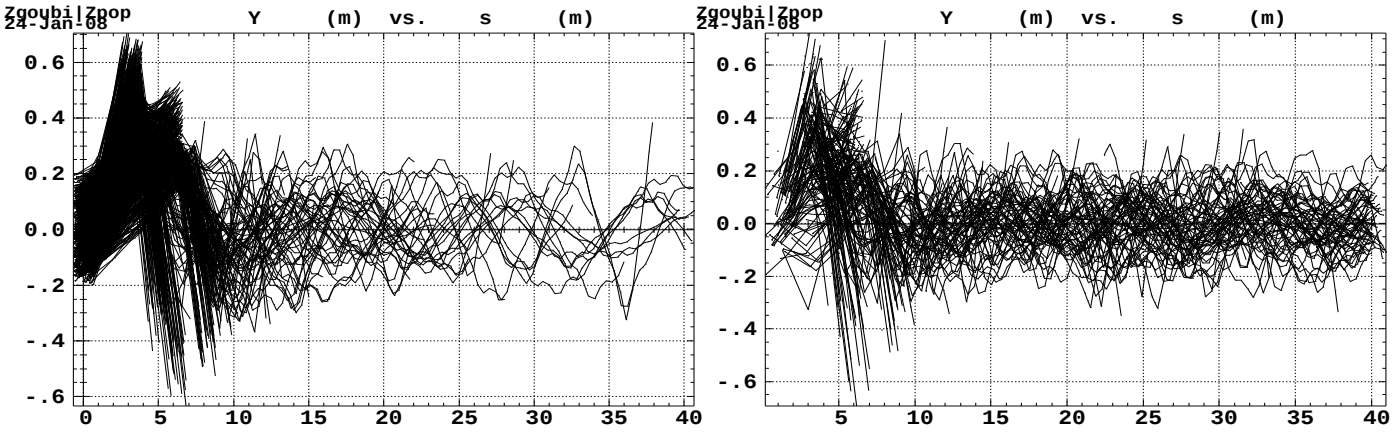
The limit path length of all particles ($I = 1$, *IMAX*) is stored in the array *FDES*$(6,I)$, for further statistical purposes. For the same purpose (e.g., use of *HISTO*), any particle of type 2 (resulting from decay of 1) will be tagged with an S standing for "secondary". When a particle decays, its coordinates D, Y, T, Z, P at the decay point are stored in *FDES*(J,I), $J = 1, 5$.

NOTE on negative drifts:
The use of negative drifts with *MCDESINT* is allowed and correct. For instance, negative drifts may occur in a structure for some of the particles when using *CHANGREF* (due to the Z-axis rotation or negative *XCE)*, or when using *DRIFT* with $XL < 0$. Provision has been made to take it into account during the *MCDESINT* procedure, as follows.

If, due to a negative drift, a secondary particle reaches back the decay spot of the primary particle from which it originated, then that primary particle is regenerated with its original coordinates at that spot. Then the secondary particle is discarded while ray-tracing resumes in a regular way for the primary particle which is again susceptible of decay at the same time-of-flight. This procedure is made possible by prior storage of the coordinates of the primary particles (in array $FDES(J, I)$) each time a decay occurs.
Negative steps ($XPAS < 0$) in optical elements are not compatible with *MCDESINT*.

EXERCISE #10 - In-flight decay in the Neutrino Factory collect channel

- Figure out what goes on in “inFlightDecay.dat“.
- Run “inFlightDecay.dat“.
- Using zpop plot pion trajectories and muon trajectories, separately.



Left : pions. Right muons.

“inFlightDecay.dat“ data file

```
NuFactory. New Funnel.  Simone object
  'MCOBJET'                                                                    1
  833.910238                      !!! Optical tuning Brho is 1000. !!!
  3
 1000
 2 2 2 2 1 1
0. 0. .0  .0   0.  1.246756
    0.     4.   0.01  1
   .0      4.   0.01  1
    0.   1.     0.4539363675   1               Etot_pi in [200,500]MeV
 123456 234567 345678
'FAISTORE'
b_zgoubi.fai    #START FUN CELLQ DRF DRIF #END CHAMB
1
   'PARTICUL'     #START
   139.567 1.60217733D-19 0. 26.03E-9 0.        PION MASS AND LIFE TIME
   'MCDESINT'                                                              4
   105.66   0.                                  PION -> MUON + NEUTRINO DECAY
   136928 768370 548375
'DRIFT'  In order to set particle time, now that PARTICUL is passed...
1.D-20
'SCALING'
 1    1
  MULTIPOL
  1
 1.
  1
'CHAMBR'
1
2 100.01 100.01 0. 0.
'MULTIPOL' DUM
0000  00002  00  .Quad
  1e-10 10.00    0.0000000000 1e-10   0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 0. 0.   1.00 0.00 0.00 0.00 0. 0. 0. 0.
6 -.010967  5.464823  .996848 1.568787 -5.671630 18.505734
 0. 0.   1.00 0.00 0.00 0.00 0. 0. 0. 0.
6 -.010967  5.464823  .996848 1.568787 -5.671630 18.505734
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
#10|3|10    Quad  Q1
1 0. 0. 0.
'DRIFT'    DRIF      DRF1
     10.0000
'MULTIPOL'   FUNA  HKIC      DQ1
0000  00002  00  .kicker
  100.1 10.00  -9.45212430E-01  2.91536125E-01  0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 29. 18.  1.00 0.00 0.00 0.00 0.00 0. 0. 0. 0.
6 -.010967  5.464823  .996848 1.568787 -5.671630 18.505734
 29. 18.  1.00 0.00 0.00 0.00 0.00 0. 0. 0. 0.
6 -.010967  5.464823  .996848 1.568787 -5.671630 18.505734
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
#20|20|20  Kick
1 0. 0. 0.
'DRIFT'    DRIF      DRF2
   40.0000
'MATRIX'
1 0
'MULTIPOL'   FUNB  QUAD      Q2
0000  00002  00  .Quad
 100. 10.00    0.0000000000 -6.20657416E-01  0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 29. 18.  1.00 0.00 0.00 0.00 0.00 0. 0. 0. 0.
6 -.010967  5.464823  .996848 1.568787 -5.671630 18.505734
 29. 18.  1.00 0.00 0.00 0.00 0.00 0. 0. 0. 0.
6 -.010967  5.464823  .996848 1.568787 -5.671630 18.505734
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
#20|20|20    Quad  Q2
1 0. 0. 0.
'DRIFT'    DRIF      DRF3                                                       8
  65.
'MATRIX'
1 0
'MULTIPOL'   FUNC  QUAD      Q3
0000  00002  00  .Quad
 100.1 10.00 -3.29102108E-01  8.46280488E-01 0.0 0.0 0.0 0.0 0.0
 29. 18.  1.00 0.00 0.00 0.00 0.00 0. 0. 0. 0.
6 -.010967  5.464823  .996848 1.568787 -5.671630 18.505734
 29. 18.  1.00 0.00 0.00 0.00 0.00 0. 0. 0. 0.
6 -.010967  5.464823  .996848 1.568787 -5.671630 18.505734
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
#20|20|20    Quad  Q3
1 0. 0. 0.
'DRIFT'    DRIF      DRF2                                                       7
  40.0000
'MATRIX'
1 0
'MULTIPOL'   FUND  QUAD      Q4
0000  00002  00  .Quad
 100. 10.00    0.0000000000 -7.63601486E-01  0.0 0.0 0.0 0.0 0.0
 29. 18.  1.00 0.00 0.00 0.00 0.00 0. 0. 0. 0.
6 -.010967  5.464823  .996848 1.568787 -5.671630 18.505734
 29. 18.  1.00 0.00 0.00 0.00 0.00 0. 0. 0. 0.
6 -.010967  5.464823  .996848 1.568787 -5.671630 18.505734
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
#20|20|20    Quad  Q4
1 0. 0. 0.
'DRIFT'    DRIF      DRF4
  40.0000
.........................
.........................
.........................
.........................                                                    4
.........................
.........................                                                     5
'MULTIPOL' CELLQ B-E QUADC1      QM
0000  00002  Quad
   40. 10.00    0.  -3.00591 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
19.00 12. 1.00 0.00 0.00 0.00 0.00 0. 0. 0. 0.
6  .1122 6.2671 -1.4982 3.5882 -2.1209 1.723
19.00 12. 1.00 0.00 0.00 0.00 0.00 0. 0. 0. 0.
6  .1122 6.2671 -1.4982 3.5882 -2.1209 1.723
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
#20|20|20  Quad  Q3
1 0. 0. 0.                                                                    6
'DRIFT'    DRF 18
 18.66435
'FAISCEAU' #END
 'END'                                                                         7
```

ORDRE: Taylor expansions order

The position $\vec{R}$ and velocity $\vec{u}$ of a particle are obtained from Taylor expansions as described in eq. (4.1.1). By default, these expansions are up to the fourth order derivative of $\vec{u}$,

$$\vec{R}_1 \approx \vec{R}_0 + \vec{u}_0 \Delta s + ... + \vec{u}_0^{(4)} \frac{\Delta s^5}{5!}$$
$$\vec{u}_1 \approx \vec{u}_0 + \vec{u}_0' \Delta s + \ldots + \vec{u}_0^{(4)} \frac{\Delta s^4}{4!}$$

which corresponds to third order derivatives of $\vec{B}$, since (eq. (4.1.7))

$$\vec{u}^{(4)} = \vec{u}''' \times \vec{B} + 3\vec{u}'' \times \vec{B}' + 3\vec{u}' \times \vec{B}'' + \vec{u} \times \vec{B}'''$$

and to the third order derivatives of $\vec{E}$ (eq. (4.1.10)) as well.

However $\vec{B}'''$, or $\vec{E}'''$, and higher order derivatives may be zero in second order type optical elements, for instance in a sharp edge quadrupole. Also, in several elements, no more than first and second order field derivatives are implemented in the code. One may also wish to fasten calculations by limiting the time-consuming calculation of lengthy (while possibly ineffective in terms of accuracy) Taylor expansions.

In that spirit, the purpose of *ORDRE*, option $IO = 2 - 5$, is to allow for expansions to the $\vec{u}_0^{(IO)}$ term in eq. 4.1.1. Default functionning is $IO = 4$.

Note the following:

As concerns the optical elements

QUADRUPO, SEXTUPOL, OCTUPOLE, DECAPOLE, DODECAPO, MULTIPOL, ELMULT, EBMULT

magnetic field derivatives (see eq. 4.1.8) have been installed in the code according to $\vec{u}_0^{(5)}$ developement order; it may not be as complete for some other optical elements, as well as for the possible electric field component whose field derivatives may not be provided to more than second order.

In electric optical elements field derivatives (eq. 4.1.12) are usually provided to no more than second order, which justifies saving computing time by not pushing Taylor expansions as high as $\vec{u}_0^{(5)}$.

NOTE: see also the option *IORDRE* in field map declarations (*DIPOLE-M, TOSCA*, etc.).

PARTICUL: Particle characteristics

PARTICUL allows the definition of several characteristics of the particles (mass, charge, gyromagnetic factor and life-time in the center of mass), that are needed in various procedures as,

MCDESINT	: mass, COM life-time
SPNTRK	: mass, gyromagnetic factor
SRLOSS	: mass, charge
SYNRAD	: mass, charge
Electric and Electro-Magnetic elements	: mass, charge

The declaration of *PARTICUL* must **precede** these keywords.
Note that, in the case of electric or electro-magnetic optical elements, the mass and charge are needed in order to compute the particle velocity v, as involved in eq. 4.1.2.

REBELOTE: 'Do it again'

As soon as *REBELOTE* is encountered in the input data file, the code execution jumps back to the beginning of the data file to start a new run, and so on up to *NPASS* times. When the following random procedures are used: *MCOBJET, OBJETA, MCDESINT, SPNTRK* (*KSO* = 5), their random seeds are not reset, and therefore independent statistics will add up. *REBELOTE* is dedicated either to Monte Carlo calculations when more than 10^4particles are to be tracked (due to *IMAX* $\leq 10^4$, see *MCOBJET*), or to the tracking in circular machines (e.g. Synchrotron accelerators). The option index K is then used to either generate new initial coordinates ($K = 0$ see section 7.6.7), when using *MCOBJET* or any other generator of random initial coordinates, or in order that the final coordinates at the last run be taken as the initial coordinates of the next ($K = 99$ — see section 7.6.4).

Monte Carlo simulations: normally $K = 0$. *NPASS* runs through the same structure will follow, resulting in the calculation of $(1 + NPASS) * IMAX$ trajectories.

Circular machines: normally $K = 99$. *NPASS* turns in the same structure will follow, resulting in the tracking of *IMAX* particles over $1 + NPASS$ turns (Note: for the simulation of accelerators and synchrotron motion, see *SCALING*).

Output prints over *NPASS* runs might result in a prohibitively big file. They may be inhibited by means of the option *KWRIT*$= 0$.

REBELOTE provides statistical calculations and related informations on particle decay *(MCDESINT)*, spin tracking *(SPNTRK)*, stopped particles (*CHAMBR, COLLIMA*).

RESET: Reset counters and flags

Piling up problems in **zgoubi** input data file is allowed, with normally no particular precaution, except that each new problem must begin with a new object definition (with *MCOBJET, OBJET*, etc.). Nevertheless, when calling upon certain keywords, flags, counters or integrating procedures are involved. It may therefore be necessary to reset them. This is the purpose of *RESET* which normally appears right after the object definition and causes each problem to be treated as a new and independent one.

The keywords or procedures of concern and the effect of *RESET* are the following

CHAMBR : *NOUT* = number of stopped particles = 0; *CHAMBR* option switched off

COLLIMA : *NOUT* = number of stopped particles = 0

HISTO : Histograms are emptied

INTEG : *NRJ* = number of particles out of range = 0 (*INTEG* is the numerical integration subroutine; *NRJ* is incremented when a particle goes out of a field map)

MCDESINT : Decay in flight option switched off

SCALING : Scaling options disabled

SPNTRK : Spin tracking option switched off

SCALING: Time scaling of power supplies and R.F.

SCALING acts as a function generator dedicated to varying fields in optical elements, or potentials in electrostatic devices, or frequency in *CAVITE*. It is normally intended to be declared right after the object definition, and used in conjunction with *REBELOTE*, for the simulation of multiturn tracking - possibly including acceleration cycles.

SCALING acts on families of elements, a family being designated by its name that coincides with the keyword of the corresponding element. For instance, declaring *MULTIPOL* as to be varied will result in the same timing law being applied to all *MULTIPOL*'s in the **zgoubi** optical structure data file. Subsets can be selected by labeling keywords in the data file (section 7.6.3, page 238) and adding the corresponding *LABEL*('s) in the *SCALING* declarations (two *LABEL*'s maximum). The family name of concern, as well as the field versus timing scaling law of that family (or frequency versus timing in the case of *CAVITE*) are given as input data to the keyword *SCALING*. Up to 9 families can be declared as subject to a scaling law; a scaling law can be made of up to 10 successive timings; between two successive timings, the variation law is linear.

An example of data formatting is given hereafter. This is the SATURNE case in fact, already visited when tracking spin.

SCALING		- Scaling
1 4		Active. 4 families of elements are concern
QUADRUPO QFA QFB		- Quadrupoles labeled 'QFA' and Quadrup
2		2 timings
18131.E-3	24176.E-3	The field increases (linearly) from 18131E
1	6379	from turn 1 to turn 6379
MULTIPOL QDA QDB		- Multipoles labeled 'QDA' and Multipole
2		
18131.E-3	24176.E-3	Fields increase from 18131E-3$*B_i$ to 241
1	6379	from turn 1 to turn 6379
BEND		- All *BEND*'s (regardless of any LABEL)
2		
18131.E-3	24176.E-3	Same scaling
1	6379	
CAVITE		- Accelerating cavity
2		
1 1.22	1.33352	The synchronous rigidity $(B\rho)_s$ increases,
1 1200	6379	from $(B\rho)_{s_o}$ to 1.22 $*(B\rho)_{s_o}$ from turn 1 to
		from 1.22 $*(B\rho)_{s_o}$ to 1.33352 $(B\rho)_{s_o}$ from

The timing is in unit of turns. In this example, TIMING = 1 to 6379 (turns). Therefore, at turn number N, B and B_i are updated in the following way. Let $SCALE(TIMING = N)$ be the updating scale factor

$$\text{SCALE}(N) = 18.131\frac{24.176 - 18.131}{1 + 6379 - 1}(N - 1)$$

and then

$$B(N) = SCALE(N)B_0$$
$$B_i(N) = SCALE(N)B_{i0}$$

The R.F. frequency is computed using

$$f_{RF} = \frac{hc}{\mathcal{L}} \frac{q(B\rho)_s}{(q^2(B\rho)_s^2 + (Mc^2)^2)^{1/2}}$$

where the rigidity is updated in the following way. Let $(B\rho)_{s_o}$ be the initial rigidity (namely, $(B\rho)_{s_o} = BORO$ as defined in the keyword *OBJET* for instance). Then, at turn number N,

$$\text{if } 1 \leq N \leq 1200 \text{ then, } SCALE(N) = 1 + \frac{1.22 - 1}{1 + 1200 - 1}(N - 1)$$

$$\text{if } 1200 \leq N \leq 6379 \text{ then, } SCALE(N) = 1.22 + \frac{1.33352 - 1.22}{1 + 6379 - 1200}(N - 1200)$$

and then,

$$(B\rho)_s(N) = SCALE(N) \cdot (B\rho)_{s_o}$$

from which value the calculations of $f_{RF}(N)$ follow.

Note : It may happen that some optical elements won't scale, for source code developement reasons. This should be paid attention to.

SPNTRK: Spin tracking

The keyword *SPNTRK* permits switching on the spin tracking option. It also permits the attribution of an initial spin component to each one of the *IMAX* particles of the beam, following a distribution that depends on the option index *KSO*. It must be preceded by *PARTICUL* for the definition of mass and gyromagnetic factor.

KSO = 1 (respectively 2, 3): the *IMAX* particles of the beam are given a longitudinal (1,0,0) spin component (respectively transverse horizontal (0,1,0), vertical (0,0,1)).

KSO = 4: initial spin components are entered explicitly for each one of the *IMAX* particles of the beam.

KSO = 5: random generation of *IMAX* initial spin conditions as described in Fig. 11. Given a mean polarization axis (S) defined by its angles T_0 and P_0, and a cone of angle A with respect to this axis, the *IMAX* spins are sorted randomly in a Gaussian distribution

$$p(a) = \exp\left[-\frac{(A-a)^2}{2\delta A^2}\right] / \delta A \sqrt{2\pi}$$

and within a cylindrical uniform distribution around the (S) axis. Examples of simple distributions available by this mean are given in Fig. 12.

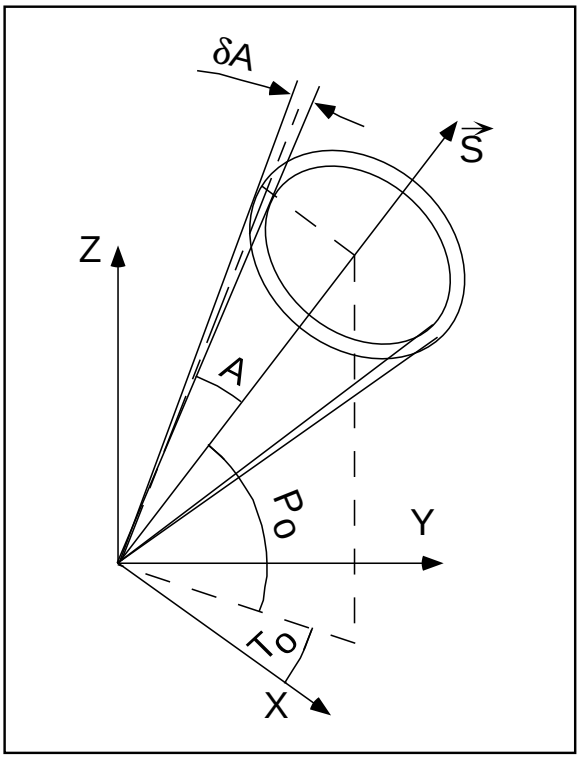


Figure 11: Spin distribution as obtained with option *KSO* = 5. The spins are distributed within an annular strip δA (standard deviation) at an angle A with respect to the axis of mean polarization (S) defined by T_0 and P_0.

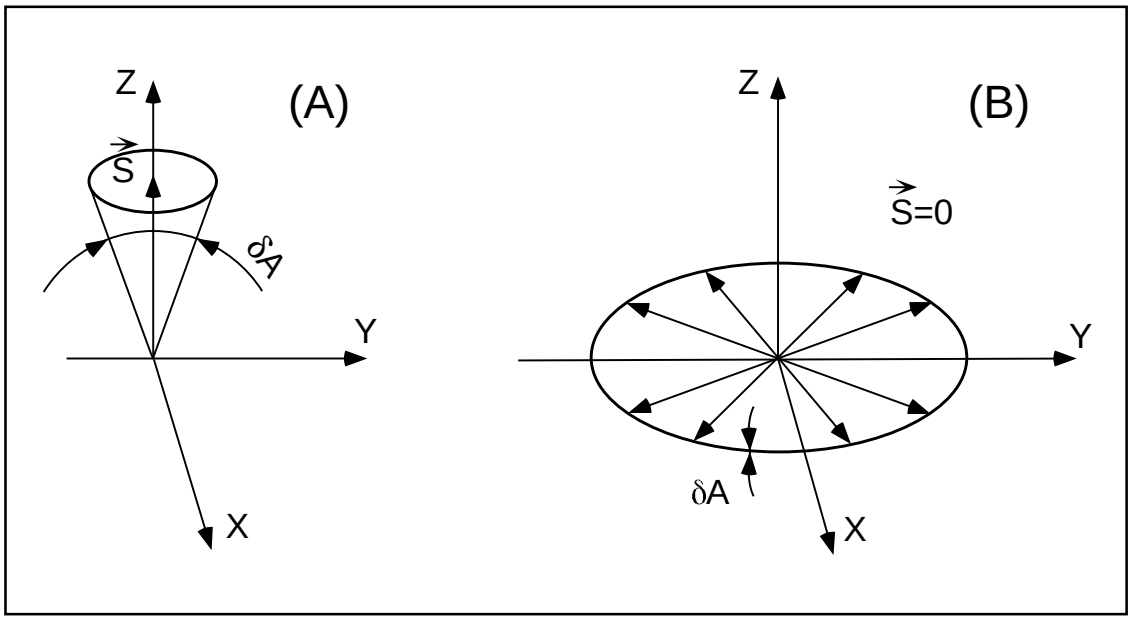


Figure 12: Examples of the use of *KSO* =5.
A: Gaussian distribution around a mean vertical polarization axis, obtained with T_0 = arbitrary, $P_0 = \pi/2$, $A = 0$ and $\delta A \neq 0$.
B: Isotropic distribution in the median plane, obtained with $P_0 = \pm\pi/2$, $A = \pi/2$, and $\delta A = 0$.

SRLOSS: Synchrotron radiation loss

[10]

The keyword *SRLOSS* allows activating or stopping (option $KSR = 1, 0$ respectively) stepwise tracking of energy loss by emission of photons in magnetic fields and the ensuing particle energy perturbation. It must be preceded by *PARTICUL* for defining mass and charge values as they enter in the definition of SR parameters.

Statistics on SR parameters are perform while tracking, results of which can be obtained by means of keyword SRPRNL.

SYNRAD: Synchrotron radiation spectral-angular densities

***** Not implemented in **zgoubi** . It is implemented in zpop. *****

The keyword *SYNRAD* enables (or disables) the calculation of synchrotron radiation (SR) electric field and spectral angular energy density. It must be preceded by *PARTICUL* for defining mass and charge values, as they enter in the definition of SR parameters.

SYNRAD is supposed to appear a first time at the location where SR calculations should start, with the first data *KSR* set to 1. It results in on-line storage of the electric field vector and other relevant quantities in zgoubi.sre, as step by step integration proceeds. The observer position (XO, YO, ZO) is specified next to *KSR*.

Data stored in zgoubi.sre:

(ELx, ELy, ELz): electric field vector $\vec{\mathcal{E}}$ (eq. 6.2.1)

$(btx, bty, btz) = \vec{\beta} = \frac{1}{c} \times$ particle velocity

$(gx, gy, gz) = \frac{d\vec{\beta}}{dt} =$ particle acceleration (eq. 6.2.3)

$\Delta\tau =$ observer time increment (eq. 6.2.2)

$t' = \tau - r(t')/c =$ retarded (particle) time

$(rtx, rty, rtz) : \vec{R}(t)$, particle to observer vector (eq. 6.2.4)

$(x, y, z) =$ particle coordinates

$\Delta s =$ step size in the magnet (fig. 2)

$NS =$ step number

$I =$ particle number

$LET(I) =$ tagging letter

$IEX(I) =$ stop flag (see section 7.6.8)

SYNRAD is supposed to appear a second time at the location where SR calculations should stop, with *KSR* set to 2. It results in the output of the angular

energy density $\int_{\nu_1}^{\nu_2} \partial^3 W/\partial\phi\,\partial\psi\,\partial\nu$ (eq. 6.2.11) as calculated from the Fourier transform of the electric field (eq. 6.2.11). The spectral range of interest and frequency sampling (ν_1, ν_2, N) are specified next to *KSR*.

Note that *KSR* = 0 followed by a dummy line of data allows temporary inhibition of SR procedures.

7.4 Optical Elements and related numerical procedures

AIMANT: Generation of dipole mid-plane 2-D map, polar frame

The keyword *AIMANT* provides an automatic generation of a dipole median plane field map in polar coordinates. A more recent and improved version will be found in *DIPOLE-M*. The extent of the map is defined by the following parameters, as shown in Figs. 13A and 13B.

AT : total angular aperture

RM : mean radius used for the positioning of field boundaries

RMIN, RMAX : minimum and maximum radial boundaries of the map

The 2 or 3 effective field boundaries (EFB) inside the map are defined from geometric boundaries, the shape and position of which are determined by the following parameters.

ACENT : arbitrary angle, used for the positioning of the EFB's.

ω : azimuth of an EFB with respect to *ACENT*

θ : angle of a boundary with respect to its azimuth (wedge angle)

R_1, R_2 : radius of curvature of an EFB

U_1, U_2 : extent of the linear part of the EFB.

At any node of the map mesh, the value of the Z component of the field is calculated as

$$B_Z = \mathcal{F} * B_0 * \left(1 + N * \left(\frac{R - RM}{RM} \right) + B * \left(\frac{R - RM}{RM} \right)^2 + G * \left(\frac{R - RM}{RM} \right)^3 \right) \quad (7.4.1)$$

where N, B and G are respectively the first, second and third order field indices and $\mathcal{F}$ is the fringe field coefficient.

Calculation of the Fringe Field Coefficient

With each EFB a realistic extent of the fringe field, λ, is associated (Figs. 13A and 13B), and a fringe field coefficient F is calculated. In the following λ stands for either λ_E (Entrance), λ_S (Exit) or λ_L (Lateral EFB).
If a node of the map mesh is at a distance of the EFB larger than λ, then $F = 0$ outside the field map and $\mathcal{F} = 1$ inside. If a node is inside the fringe field zone, then F is calculated as follows.
Two options are available, for the calculation of F, depending on the value of ξ.
If $\xi \geq 0$, F is a second order type fringe field (Fig. 14) given by

$$F = \frac{1}{2}\frac{(\lambda - s)^2}{\lambda^2 - \xi^2} \quad \text{if } \xi \leq s \leq \lambda \tag{7.4.2}$$

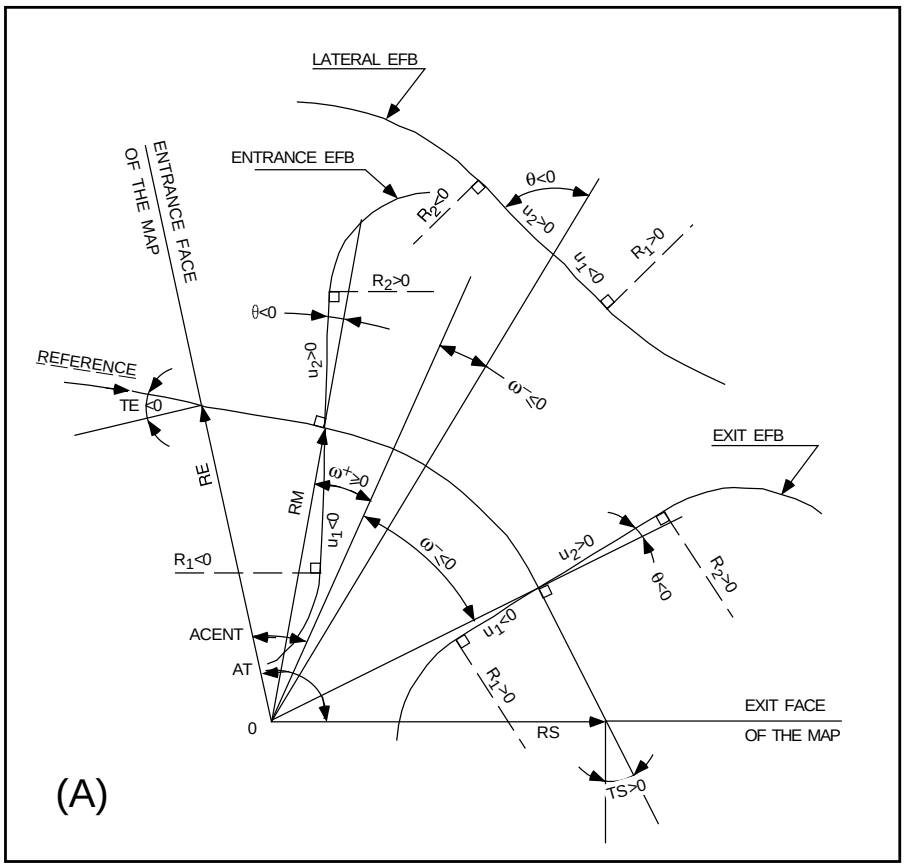
$$F = 1 - \frac{1}{2}\frac{(\lambda - s)^2}{\lambda^2 - \xi^2} \quad \text{if } -\lambda \leq s \leq -\xi \tag{7.4.3}$$

where s is the distance to the EFB, and

$$F = \frac{1}{2} + \frac{s}{\lambda + \xi} \quad \text{if } 0 \leq s \leq \xi \tag{7.4.4}$$

$$F = \frac{1}{2} - \frac{s}{\lambda + \xi} \quad \text{if } -\xi \leq s \leq 0 \tag{7.4.5}$$

This simple model allows a rapid calculation of the fringe field, but may lead to erratic behavior of the field when extrapolating out of the median plane, due to the discontinuity of d^2B/ds^2 at $s = \pm\xi$ and $s = \pm\lambda$. For more accuracy it is better to use the next option.



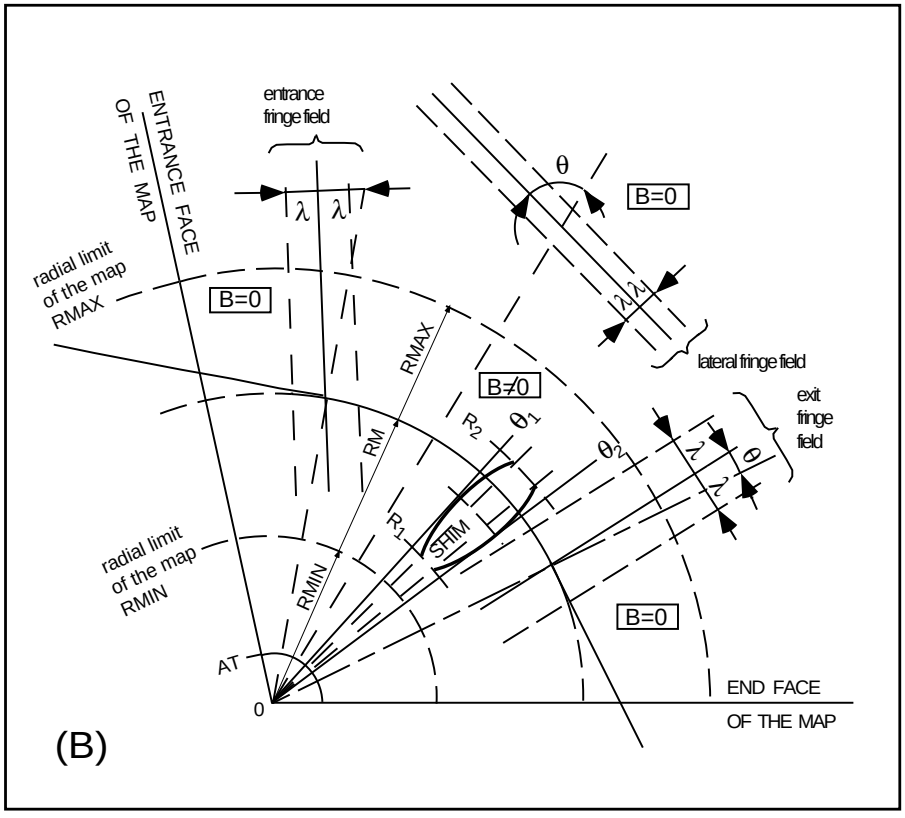


Figure 13: A: Parameters used to define the field map and geometric boundaries.
B: Parameters used to define the field map and fringe fields.

If $\xi = -1$, F is an exponential type fringe field (Fig. 14) given by [19]

$$F = \frac{1}{1 + \exp P(s)} \tag{7.4.6}$$

where s is the distance to the EFB, and

$$P(s) = C_0 + C_1 \left(\frac{s}{\lambda}\right) + C_2 \left(\frac{s}{\lambda}\right)^2 + C_3 \left(\frac{s}{\lambda}\right)^3 + C_4 \left(\frac{s}{\lambda}\right)^4 + C_5 \left(\frac{s}{\lambda}\right)^5 \tag{7.4.7}$$

The values of the coefficients C_0 to C_5 should be such that the derivatives of B_Z with respect to s be negligible at $s = \pm\lambda$, so as not to perturb the extrapolation of $\vec{B}$ out of the median plane (this restriction no longer holds in the improved version *DIPOLE-M*).

It is also possible to simulate a shift of the EFB, by giving a non zero value to the parameter *SHIFT*. s is then changed to $s-$ *SHIFT* in the previous equation. This allows small variations of the total magnetic length.

Let F_E (respectively F_S, F_L) be the fringe field coefficient attached to the entrance (respectively exit, lateral) EFB following eqs. above. At any node of the map mesh, the resulting value of the fringe field coefficient (eq. 7.4.1) is (Fig. 15)

$$\mathcal{F} = F_E * F_S * F_L$$

($F_L = 1$ if no lateral EFB is requested).

The Mesh of the Field Map

The magnetic field is calculated at the nodes of a mesh with polar coordinates, in the median plane. The radial step is given by

$$\delta R = \frac{\textit{RMAX - RMIN}}{\textit{IRMAX} - 1}$$

and the angular step by

$$\delta\theta = \frac{AT}{\textit{IAMAX} - 1}$$

where, *RMIN* and *RMAX* are the lower and upper radial limits of the field map, and AT is its total angular aperture (Fig. 13B). *IRMAX* and *IAMAX* are the total number of nodes in the radial and angular directions.

Simulating Field Defects and Shims

Once the initial map is calculated, it is possible to modify it by means of the parameter *NBS*, so as to simulate field defects or shims.

If $\mathbf{NBS = -2}$, the map is globally modified by a perturbation proportional to $R - R_0$, where R_0 is an arbitrary radius, with an amplitude $\Delta B_Z / B_0$, so that B_Z at the nodes of the mesh is replaced by

$$B_Z * \left(1 + \frac{\Delta B_Z}{B_0} \frac{R - R_0}{RMAX - RMIN}\right)$$

If $\mathbf{NBS = -1}$, the perturbation is proportional to $\theta - \theta_0$, and B_Z is replaced by

$$B_Z * \left(1 + \frac{\Delta B_Z}{B_0} \frac{\theta - \theta_0}{AT}\right)$$

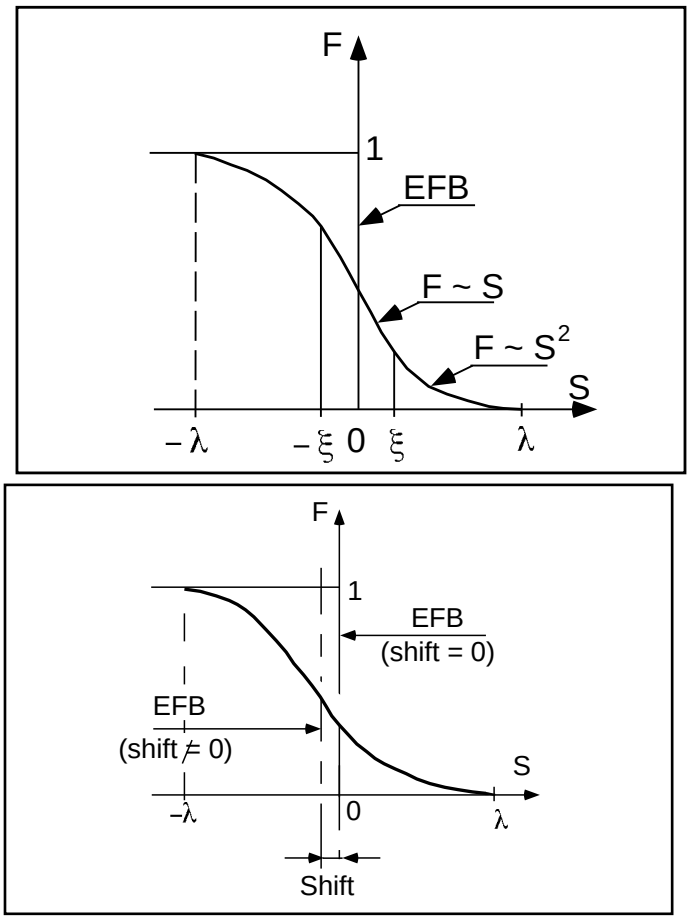


Figure 14: Second order type fringe field (upper plot) and exponential type fringe field (lower plot).

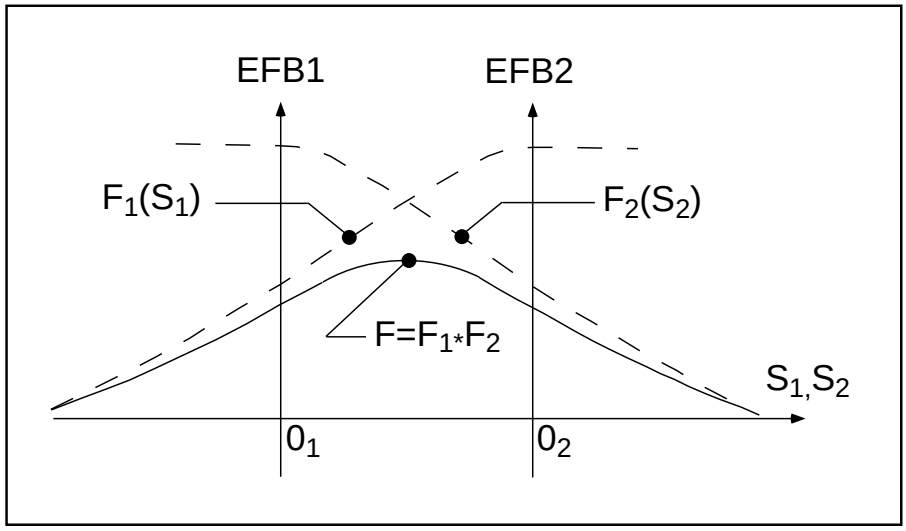


Figure 15: Effective value of $\mathcal{F}$ for overlapping fringe fields F_1 and F_2 centered at O_1 and O_2.

If NBS $\geq$ 1, then *NBS* shims are introduced at positions $\frac{R_1+R_2}{2}$, $\frac{\theta_1+\theta_2}{2}$ (Fig. 16) [20]

The initial field map is modified by shims with second order profiles given by

$$\theta = \left(\gamma + \frac{\alpha}{\mu}\right) \beta \frac{X^2}{\rho^2}$$

where X is shown in Fig. 16, $\rho = \frac{R_1+R_2}{2}$ is the central radius, α and γ are the angular limits of the shim, β and μ are parameters.

At each shim, the value of B_Z at any node of the initial map is replaced by

$$B_Z * \left(1 + F\theta * FR * \frac{\Delta B_Z}{B_0}\right)$$

where $F\theta = 0$ or $FR = 0$ outside the shim, and $F\theta = 1$ and $FR = 1$ inside.

Extrapolation Off Median Plane

The vector field $\vec{B}$ and its derivatives in the median plane are calculated by means of a second or fourth order polynomial interpolation, depending on the value of the parameter *IORDRE* (*IORDRE*=2, 25 or 4, see section 4.3.2). The transformation from polar to Cartesian coordinates is performed following eqs. (4.3.9 or 4.3.10). Extrapolation off median phase is then performed by means of Taylor expansions following the procedure described in section 4.2.2.

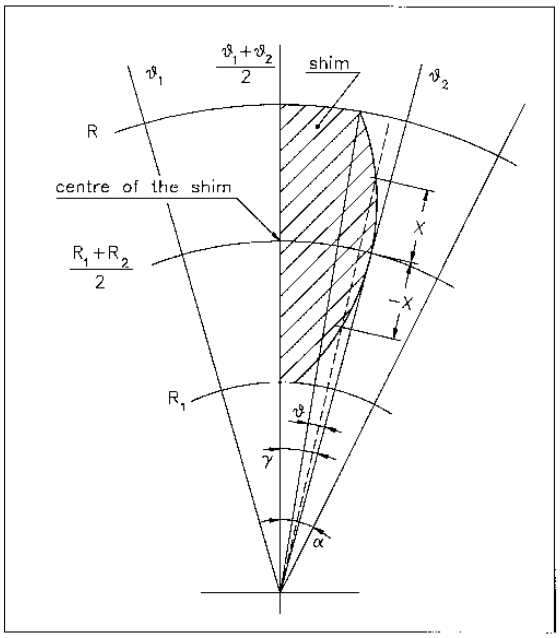


Figure 16: A second order profile shim. The shim is centered at $\frac{(R_1+R_2)}{2}$ and $\frac{(\theta_1+\theta_2)}{2}$.

AUTOREF: Automatic transformation to a new reference frame

AUTOREF positions the new reference frame following 3 options:

If I = 1, *AUTOREF* is equivalent to

$$CHANGREF[XCE = 0, YCE = Y(1), ALE = T(1)]$$

so that the new reference frame is at the exit of the last element, with particle 1 at the origin with its horizontal angle set to $T = 0$.

If I = 2, it is equivalent to

$$CHANGREF[XW, YW, T(1)]$$

so that the new reference frame is at the position (XW, YW) of the waist (calculated automatically in the same way as for *IMAGE*) of the three rays number 1, 4 and 5 (compatible for instance with *OBJET, KOBJ* = 5, 6 together with the use of *MATRIX*) while $T(1)$ is set to zero.

If I = 3, it is equivalent to

$$CHANGREF[XW, YW, T(I1)]$$

so that the new reference frame is at the position (XW, YW) of the waist (calculated automatically in the same way as for *IMAGE*) of the three rays number I1, I2 and I3 specified as data, while $T(1)$ is set to zero.

BEND: Bending magnet, Cartesian frame

BEND is one of the several keywords available for the simulation of dipole magnets. It presents the interest of easy handling, and is well adapted for the simulation of synchrotron dipoles and such other regular dipoles as sector magnets with wedge angles.

The dipole simulation is performed from the magnet geometrical length XL, from the skew angle (rotation *wrt.* the X axis, useful for obtaining vertical deviation magnet), and from the field $B1$ such that in absence of fringe field the deviation θ satisfies $XL = 2\frac{BORO}{B1}\sin(\frac{\theta}{2})$.

Then follows the description of the entrance and exit EFB's and fringe fields. The model is the same as for *DIPOLE*. The wedge angles W_E (entrance) and W_S (exit) are defined with respect to the sector magnet, with the signs described in Fig. 17. Within a distance $\pm X_E(\pm X_S)$ on both sides of the entrance (exit) EFB, the fringe field model is used; elsewhere, the field is supposed to be uniform.

If λ_E (resp. λ_S) is zero sharp edge field model is assumed at entrance (resp. exit) of the magnet and X_E (resp. X_S) is set to zero. In this case, the wedge angle vertical first order focusing effect (if $\vec{B}1$ is non zero) is simulated at magnet entrance and exit by a kick $P_2 = P_1 - Z_1\tan(\epsilon/\rho)$ applied to each particle (P_1, P_2 are the vertical angles upstream and downstream the EFB, Z_1 the vertical particle position at the EFB, ρ the local horizontal bending radius and ϵ the wedge angle experienced by the particle ; ϵ depends on the horizontal angle T).

Magnet (mis-)alignement is assured by *KPOS*. *KPOS* also allows some degrees of automatic alignement useful for periodic structures (section 7.6.5).

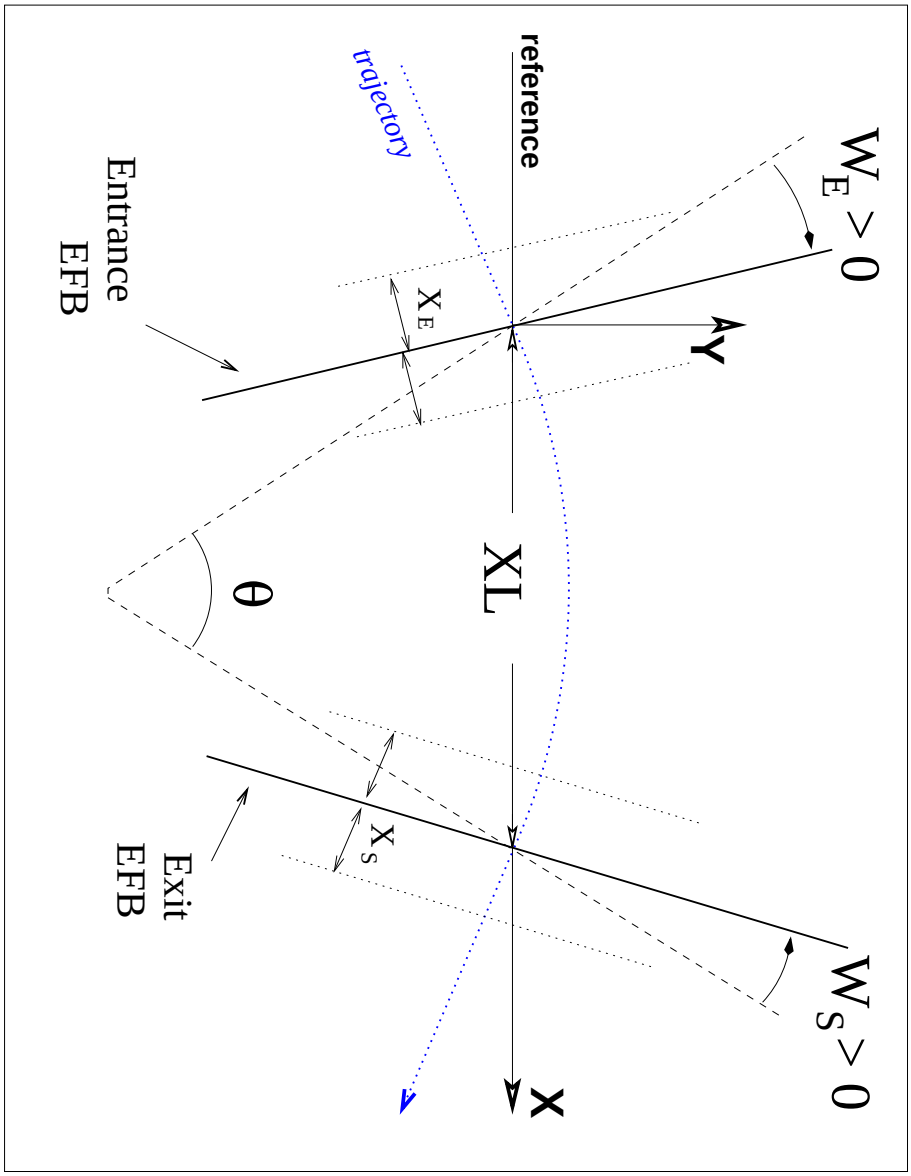


Figure 17: Geometry and parameters of *BEND* in its Cartesian frame : XL = length, θ = deviation, W_E, W_S are the entrance and exit wedge angles.

BREVOL: 1-D uniform mesh magnetic field map

BREVOL reads a 1-D axial field map from a storage data file, whose content must fit the following *FORTRAN* reading sequence

```
      OPEN (UNIT = NL, FILE = FNAME, STATUS = 'OLD' [,FORM='UNFORMATTED'])
      DO 1 I = 1, IX
        IF (BINARY) THEN
              READ(NL) X(I), BX(I)
        ELSE
              READ(NL,*) X(I), BX(I)
        ENDIF
1     CONTINUE
```

where IX is the number of nodes along the (symmetry) X-axis, $X(I)$ their coordinates, and $BX(I)$ the values of the X component of the field. BX is normalized with *BNORM* factor prior to ray-tracing, as well X is normalized with a *XNORM* coefficient (usefull to convert to centimeters, the working units in **zgoubi**). For binary files, *FNAME* must begin with 'B_ ' or 'b_ ', a flag 'BINARY' will thus be set to '.TRUE.'.

X-cylindrical symmetry is assumed, resulting in BY and BZ taken to be zero on axis. $\vec{B}(X,Y,Z)$ and its derivatives along a particle trajectory are calculated by means of a 5-point polynomial fit followed by second order off-axis Taylor series extrapolation (see sections 4.2.1, 4.3.1).

Entrance and/or exit integration boundaries may be defined in the same way as in *CARTEMES* by means of the flag ID and coefficients A, B, C, etc.

CARTEMES: 2-D Cartesian uniform mesh magnetic field map

CARTEMES was originally dedicated to the reading and processing of the measured median plane field maps of the QDD spectrometer SPES2 at Saclay. However, it can be used for the reading of any other 2-D median plane maps, provided that the format of the field data storage file fits the following *FORTRAN* sequence

```
OPEN (UNIT = NL, FILE = FNAME, STATUS = 'OLD' [,FORM='UNFORMATTED'])
        IF (BINARY) THEN
          READ(NL) (Y(J), J=1, JY)
        ELSE
          READ(NL,FMT='(10F8.2)') (Y(J), J=1, JY)
        ENDIF
        DO 1 I=1, IX
  IF (BINARY) THEN
    READ(NL) X(I), (BMES(I,J), J=1, JY)
  ELSE
    READ(NL,FMT='(10F8.1)') X(I), (BMES(I,J), J=1, JY)
          ENDIF
  1     CONTINUE
```

where, IX and JY are the number of longitudinal and transverse horizontal nodes of the uniform mesh, and $X(I)$, $Y(J)$ their coordinates. *FNAME* is the file containing the field data. For binary files, *FNAME* must begin with 'B_' or 'b_', a flag 'BINARY' will thus be set to '.TRUE.'.

The measured field *BMES* is normalized with *BNORM*,

$$B(I,J) = \mathit{BMES}(I,J) \times \mathit{BNORM}$$

As well the longitudinal coordinate X is normalized with a *XNORM* coefficient (usefull to convert to centimeters, the working units in **zgoubi** .
The vector field, $\vec{B}$, and its derivatives out of the median plane are calculated by means of a second or fourth order polynomial interpolation, depending on the value of the parameter *IORDRE* (*IORDRE* = 2, 25 or 4, see section 4.3.2).

In case a particle exits the mesh, its *IEX* flag is set to -1 (see section 7.6.8 on page 246), however it is still tracked with the field being *extrapolated* from the closest mesh nodes of the map. Note that such extrapolation process may induce eratic behavior if the distance from the mesh gets too large.

Entrance and/or exit integration boundaries can be defined with the flag ID, as follows (Fig. 18).

If ID $= 1$: the integration in the field is terminated on a boundary with equation $A'X + B'Y + C' = 0$, and then the trajectories are extrapolated linearly onto the exit end of the map.

If ID $= -1$: an entrance boundary is defined, with equation $A'X + B'Y + C' = 0$, up to which trajectories are first extrapolated linearly from the map entrance end, prior to being integrated in the field.

If ID ≥ 2: one entrance boundary, and $ID - 1$ exit boundaries are defined, as above. The integration in the field terminates on the last $(ID - 1)$ exit boundary. No extrapolation onto the map exit end is performed in this case.

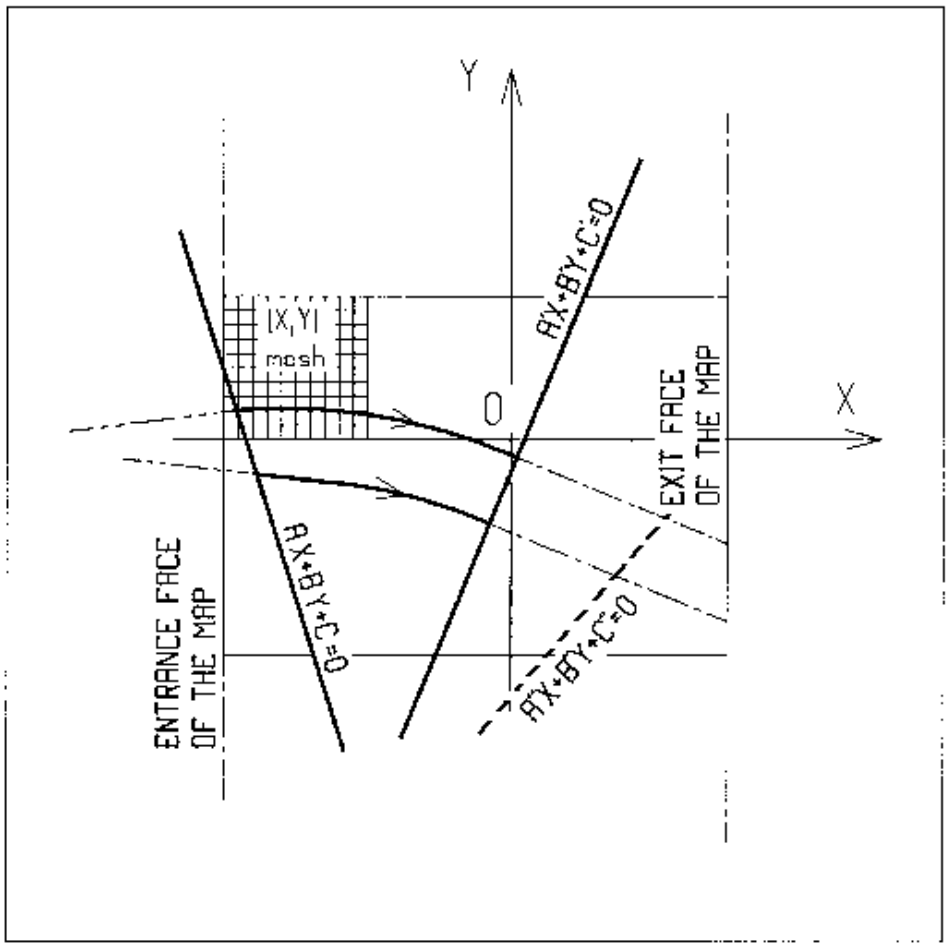


Figure 18: OXY is the coordinate system of the mesh. Integration boundaries may be defined, using $ID \neq 0$: particle coordinates are extrapolated linearly from the entrance face of the map, onto the boundary $A'X + B'Y + C' = 0$; after ray-tracing inside the map and terminating on the boundary $AX + BY + C = 0$, coordinates are extrapolated linearly onto the exit face of the map if $ID = 2$, or terminated on the last $(ID - 1)$ boundary if $ID > 2$.

CAVITE: Accelerating cavity

CAVITE provides an simulation of a (zero length) accelerating cavity; it can be used in conjunction with keywords *REBELOTE* and *SCALING* for the simulation of multiturn tracking with synchrotron acceleration (see section 7.6.7). It must be preceded by *PARTICUL* for the definition of mass M and charge q.

If IOPT = 0: *CAVITE* is switched off.

If IOPT = 1: *CAVITE* simulates the R.F. cavity of a synchrotron accelerator. Normally the keyword *CAVITE* appears **at the end** of the optical structure (the periodic motion over $IT = 1$, *NPASS* + 1 turns is simulated by means of the keyword *REBELOTE*, option K = 99 while R.F. and optical elements timings are simulated by means of *SCALING* — see section 7.6.7). The synchrotron motion of any of the *IMAX* particles of a beam is obtained by solving the following mapping

$$\begin{cases} \phi_2 - \phi_1 = 2\pi\, f_{RF} \left(\dfrac{\ell}{\beta c} - \dfrac{\mathcal{L}}{\beta_s c} \right) \\ W_2 - W_1 = q\hat{V}\, \sin\phi_1 \end{cases}$$

where

ϕ = R.F. phase; $\phi_2 - \phi_1$ = variation of ϕ between two traversals
W = kinetic energy; $W_2 - W_1$ = energy gain at a traversal of *CAVITE*
$\mathcal{L}$ = length of the synchronous closed orbit (to be calculated by prior ray-tracing, see the bottom NOTE)
ℓ = orbit length of the particle between two traversals
$\beta_s c$ = velocity of the (virtual) synchronous particle
βc = velocity of the particle
$\hat{V}$ = peak R.F. voltage
q = particle electric charge.

The R.F. frequency f_{RF} is a multiple of the synchronous revolution frequency, and is obtained from the input data, following

$$f_{RF} = \frac{hc}{\mathcal{L}} \frac{q(B\rho)_s}{\sqrt{q^2(B\rho)_s^2 + (Mc)^2}}$$

where

h = harmonic number of the R.F

M = mass of the particle

c = velocity of light.

The current rigidity $(B\rho)_s$ of the synchronous particle is obtained from the timing law specified by means of *SCALING* following $(B\rho)_s = BORO \cdot SCALE(TIMING)$ (see *SCALING* for the meaning and calculation of the scale factor *SCALE(TIMING)*). If *SCALING* is not used,

$(B\rho)_s$ is assumed to keep the constant value *BORO* given in the object description (see *OBJET* for instance).
The velocity βc of a particle is calculated from its current rigidity

$$\beta = \frac{q(B\rho)}{\sqrt{q^2(B\rho)^2 + (Mc)^2}}$$

The velocity $\beta_s c$ of the synchronous particle is obtained in the same way from

$$\beta_s = \frac{q(B\rho)_s}{\sqrt{q^2(B\rho)_s^2 + (Mc)^2}}$$

The kinetic energies and rigidities involved in these formulae are related by

$$q(B\rho) = \sqrt{W(W + 2Mc^2)}$$

Finally, the initial conditions for the mapping, at the first turn, are the following

- For the (virtual) synchronous particle

$$\phi_1 = \phi_s = \text{synchronous phase}$$
$$(B\rho)_{1s} = BORO$$

- For any of the $I = 1$, *IMAX* particles of the beam

$$\phi_{1I} = \phi_s = \text{synchronous phase}$$
$$(B\rho)_{1I} = BORO * D_I$$

where the quantities *BORO* and D_I are given in the object description.

Calculation of the coordinates

Let $p_I = \left[p_{XI}^2 + p_{YI}^2 + p_{ZI}^2\right]^{1/2}$ be the momentum of particle I at the exit of the cavity, while $p_{I_0} = \left[p_{XI_0}^2 + p_{YI_0}^2 + p_{ZI_0}^2\right]^{1/2}$ is its momentum at the entrance. The kick in momentum is assumed to be fully longitudinal, resulting in the following relations between the coordinates at the entrance (denoted by the index zero) and at the exit

$$p_{XI} = \left[p_I^2 - (p_{I_0}^2 - p_{XI_0}^2)\right]^{1/2}$$
$$p_{YI} = p_{YI_0}, \quad \text{and} \quad p_{ZI} = p_{ZI_0} \quad \text{(longitudinal kick)}$$
$$X_I = X_{I_0}, \quad Y_I = Y_{I_0} \quad \text{and} \quad Z_I = Z_{I_0} \quad \text{(zero length cavity)}$$

and for the angles (see Fig. **??**)

$$\left.\begin{aligned} T_I &= \text{Atg}\left(\frac{p_{YI}}{p_{XI}}\right) \\ P_I &= \text{Atg}\left(\frac{P_{ZI}}{(p_{XI}^2 + p_{YI}^2)^{1/2}}\right) \end{aligned}\right\} \quad \text{(damping of the transverse motion)}$$

If IOPT = 2 : the same simulation of a synchrotron R.F. cavity, as for **IOPT = 1**, is performed, except that the keyword *SCALING* (family *CAVITE*) is not taken into account in this option : the increase in kinetic energy at each traversal, for the synchronous particle, is

$$\Delta W_s = q\hat{V}\ \sin\phi_s$$

where the synchronous phase ϕ_s is given in the input data. From this, the calculation of the law $(B\rho)_s$ and the R.F. frequency f_{RF} follows, according to the formulae given in *IOPT* = 1.

If IOPT = 3: acceleration without synchrotron motion. Any particle will be given a kick

$$\Delta W = q\hat{V}\ \sin\phi_s$$

where $\hat{V}$ and ϕ_s are input data.

NOTE: Calculation of the closed orbit.
Due to the fringe fields, the horizontal closed orbit may not coincide with the ideal axis of the optical elements. One way to calculate it at the beginning of the structure (*i.e.* where the initial particle coordinates have to be defined) is to ray-trace a single particle over a sufficiently large number of turns, starting with the initial condition $(Y_0 = T_0 = Z_0 = P_0 = 0)$, and so as to obtain a statistically well-defined phase-space ellipse. The initial conditions of the closed orbit then correspond to the coordinates Y_c and T_c of the center of this ellipse. Next, ray-tracing over one turn a particle starting with the initial condition (Y_c, T_c, $Z_0 = P_0 = 0$) will provide the length $\mathcal{L}$ (namely, the $F(6,1)$ coordinate) of the closed orbit.

EXERCISE #11 - Accelerating in EMMA

We will use the EMMA data file of Ex. 4 and the object generated in Ex. 6.

• We have seen that in EXERCISE # 9. We used acceleration to get beam path in tune diagram.

See full 2π cycle using gnuplot.

CHAMBR: Long transverse aperture limitation

CHAMBR causes the identification, counting and stopping of particles that reach the transverse limits of the vacuum chamber. The chamber can be either rectangular (*IFORM* = 1) or elliptic (*IFORM* = 2). The chamber is centered at YC, ZC and has transverse dimensions $\pm YL$ and $\pm ZL$ such that any particle will be stopped if its coordinates Y, Z satisfy

$$(Y - YC)^2 \geq YL^2 \text{ or } (Z - ZC)^2 \geq ZL^2 \quad \text{if} \quad \textit{IFORM} = 1$$
$$\frac{(Y - YC)^2}{YL^2} + \frac{(Z - ZC)^2}{ZL^2} \geq 1 \quad \text{if} \quad \textit{IFORM} = 2$$

The conditions introduced with *CHAMBR* are valid along the optical structure until the next occurrence of the keyword *CHAMBR*. Then, if $IL = 1$ the aperture is possibly modified by introducing new values of YC, ZC, YL and ZL, or, if $IL = 2$ the chamber ends and information is printed concerning those particles that have been stopped.

The testing is done in optical elements at each integration step, between the *EFB*'s. For instance, in *QUADRUPO* there will be no testing from $-X_E$ to 0 and from XL to $XL + X_S$, but only from 0 to XL; in *DIPOLE*, there is no testing as long as the *ENTRANCE EFB* is not reached, and testing is stopped as soon as the *EXIT* or *LATERAL EFB*'s are passed.

In polar coordinate optical elements Y stands for the radial coordinate (e.g. with *DIPOLE*, see Figs. 6C and 13). Therefore, centering *CHAMBR* at $YC = RM$ simulates a chamber curved with radius RM, and having a radial acceptance $RM \pm YL$. The testing is done in *ESL* (*DRIFT*) at the beginning and the end, and only for positive drifts. There is no testing in *CHANGREF*.

When a particle is stopped, its index *IEX* (see *OBJET* and section 7.6.8) is set to the value -4, and its actual path length is stored in the array *SORT* for possible further statistical purposes.

CHANGREF: Transformation to a new reference frame

CHANGREF it transports the particles to a new reference frame. It can be used anywhere in a structure. The new coordinates of the particles Y_2, T_2, Z_2 and P_2 and the path length S_2 are deduced from the old ones Y_1, T_1, Z_1, P_1 and S_1 by

$$T_2 = T_1 - ALE$$
$$Y_2 = \frac{(Y_1 - YCE)\cos T_1 + XCE \sin T_1}{\cos T_2}$$
$$DL^2 = (XCE - Y_2 \sin ALE)^2 + (YCE - Y_1 + Y_2 \cos ALE)^2$$
$$Z_2 = Z_1 + DL\mathrm{tg}P_1$$
$$S_2 = S_1 + \frac{DL}{\cos P_1}$$
$$P_2 = P_1$$

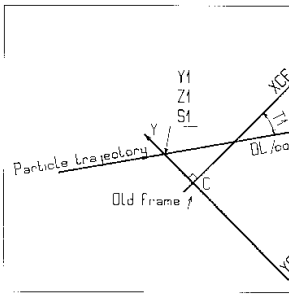


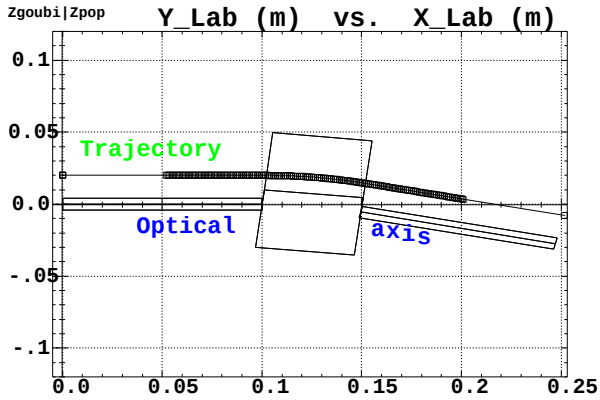
Figure
Scheme of the *CH*
dure.

where, XCE and YCE are shifts in the horizontal plane along,respectively, X- and Y-axis, and ALE is a rotation around the Z-axis. DL is given the sign of $XCE - Y_2 \sin(ALE)$.
This keyword may for instance be used for positioning optical elements, or for setting a reference frame at the entrance or exit of field maps, or to simulate misalignements (see also *KPOS*).
Effects of *CHANGREF* on spin tracking, particle decay and gas-scattering are taken into account (but not on synchrotron radiation).

The example below shows the use of *CHANGREF* for the symmetric positioning of a dipole+quadrupole magnet in a drift-bend-drift geometry with 12.691 degrees deviation (obtained upon combined effect of a dipole component and of quadrupole axis shifted 1 cm off optical axis).

Zgoubi data file :

```
Using CHANGREF
'OBJET'
51.71103865921708                          Electron, Ekin=15MeV.
2
1 1                                          One particle, with
2. 0.  0.0 0.0 0.0 1. 'R'      Y_0=2 cm, other coordinates zero.
1 1 1 1 1 1 1
'MARKER'    BEG    .plt               -> list  into zgoubi.plt.
'DRIFT'                                          10 cm drift.
10.
'CHANGREF'
0. 0. -6.34165                            First half z-rotate.
'CHANGREF'
0. 1. 0.                                              Next Y-shift.
'MULTIPOL'     Combined function multipole, dipole + quadrupole.
   2                                  -> list  into zgoubi.plt.
5  10. 2.064995867082342  2. 0. 0. 0. 0. 0. 0. 0.
 0 0  5. 1.1  1.00 1.00 1.00 1.00 1.00 1. 1. 1. 1.
4  .1455   2.2670  -.6395  1.1558  0. 0.  0.
 0 0  5. 1.1  1.00 1.00 1.00 1.00 1.00 1. 1. 1. 1.
4  .1455   2.2670  -.6395  1.1558  0. 0.  0.
0 0 0 0 0 0 0 0 0 0
.1   step size
1  0. 0.  0.
'CHANGREF'
0. -1. -6.34165          First Y-shift back, next half z-rotate.
'DRIFT'                                          10 cm drift.
10.
'MARKER'    END    .plt               -> list  into zgoubi.plt.
'FAISCEAU'
'END'
```



Note: The square markers scheme the stepwise integration in case of ± 5 cm additional fringe field extent upstream and downstream of the 5 cm long multipole.

CIBLE or TARGET: Generate a secondary beam from target interaction

The reaction is $1 + 2 \longrightarrow 3 + 4$ with the following parameters

Laboratory momentum	$p_1 \equiv 0$	p_2	p_3	p_4
Rest mass	M_1	M_2	M_3	M_4
Total energy in laboratory	M_1c^2	W_2	W_3	W_4

The geometry of the interaction is shown in Fig. 20.

The angular sampling at the exit of the target consists of the NT coordinates 0, $\pm TS$, $\pm 2 * TS$... $\pm(NT-1) * TS/2$ in the median plane, and the NP coordinates 0, $\pm PS$, $\pm 2 * PS$... $\pm(NP-1) * PS/2$ in the vertical plane.

The position of B downstream is deduced from that of A upstream by a transformation equivalent to two transformations using *CHANGREF*, namely

$$\textit{CHANGREF}(XCE = YCE = 0, \quad \textit{ALE} = \beta)$$

followed by

$$\textit{CHANGREF}(XCE = YCE = 0, \quad \textit{ALE} = \theta - \beta).$$

Particle 4 is discarded, while particle 3 continues. The energy loss Q is related to the variable mass M_4 by

$$Q = M_1 + M_2 - (M_3 + M_4) \qquad \text{and} \qquad dQ = -dM_4$$

The momentum sampling of particle 3 is derived from conservation of energy and momentum, according to

$$M_1c^2 + W_2 = W_3 + W_4$$
$$p_4^2 = p_2^2 + p_3^2 - 2p_2p_3\cos(\theta - T)$$

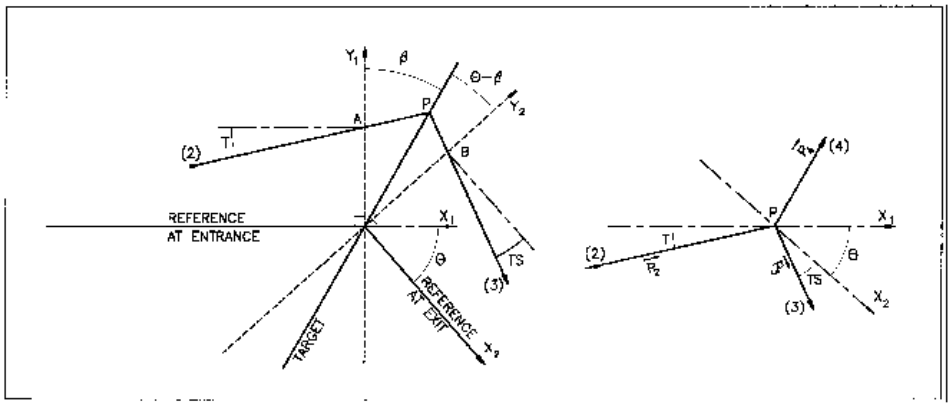


Figure 20: Scheme of the principles of *CIBLE (TARGET)*
A, T = position, angle of incoming particle 2 in the entrance reference frame
P = position of the interaction
B, T = position, angle of the secondary particle in the exit reference frame
θ = angle between entrance and exit frames
β = tilt angle of the target

COLLIMA: Collimator

COLLIMA acts as a mathematical aperture of zero length. It causes the identification, counting and stopping of particles that reach the aperture limits.

Physical aperture

A physical aperture can be either rectangular (*IFORM* = 1) or elliptic (*IFORM* = 2). The collimator is centered at YC, ZC and has transverse dimensions $\pm YL$ and $\pm ZL$ such that any particle will be stopped if its coordinates Y, Z satisfy

$$(Y - YC)^2 \geq YL^2 \text{ or } (Z - ZC)^2 \geq ZL^2 \quad \text{if} \quad \textit{IFORM} = 1$$
$$\frac{(Y - YC)^2}{YL^2} + \frac{(Z - ZC)^2}{ZL^2} \geq 1 \quad \text{if} \quad \textit{IFORM} = 2$$

Longitudinal collimation

COLLIMA can act as a longitudinal phase-space aperture, coordinates acted on are selected with *IFORM.J.* Any particle will be stopped if its horizontal (h) and vertical (v) coordinates satisfy

$$(h \leq h_{min} \text{ or } h \geq h_{max}) \text{ or } (v \leq v_{min} \text{ or } v \geq v_{max})$$

wherein, h is either path length S if *IFORM*=6 or time if *IFORM*=7, and v is either 1+DP/P if *J*=1 or kinetic energy if *J*=2 (provided mass and charge have been defined using the keyword *PARTICUL*).
If *IFORM*=11 (respectively 12) then ϵ_Y/π (respectively ϵ_Z/π) is to be specified by the user as well as $\alpha_{Y,Z}$, $\beta_{Y,Z}$. If *IFORM*=14 (respectively 15) then α_Y and β_Y (respectively α_Z, β_Z) are computed by **zgoubi** by prior matching of the particle population, only $\epsilon_{Y,Z}/\pi$ need be specified by the user.

Phase-space collimation

COLLIMA can act as a phase-space aperture. Any particle will be stopped if its coordinates satisfy

$$\gamma_Y Y^2 + 2\alpha_Y YT + \beta_Y T^2 \geq \epsilon_Y/\pi \quad \text{if} \quad \textit{IFORM} = 11 \text{ or } 14$$
$$\gamma_Z Z^2 + 2\alpha_Z ZP + \beta_Z P^2 \geq \epsilon_Z/\pi \quad \text{if} \quad \textit{IFORM} = 12 \text{ or } 15$$

If *IFORM*=11 (respectively 12) then ϵ_Y/π (respectively ϵ_Z/π) is to be specified by the user as well as $\alpha_{Y,Z}$, $\beta_{Y,Z}$. If *IFORM*=14 (respectively 15) then α_Y and β_Y (respectively α_Z, β_Z) are computed by **zgoubi** by prior matching of the particle population, only $\epsilon_{Y,Z}/\pi$ need be specified by the user.

When a particle is stopped, its index *IEX* (see *OBJET* and section 7.6.8) is set to the value -4, and its actual path length is stored in the array *SORT* for possible further statistical purposes (e.g. with *HISTO*).

DECAPOLE: Decapole magnet (Fig. 21)

The meaning of parameters for *DECAPOLE* is the same as for *QUADRUPO.*

In fringe field regions the magnetic field $\vec{B}(X,Y,Z)$ and its derivatives up to fourth order are derived from the scalar potential approximated to the 5th order in Y and Z

$$V(X,Y,Z) = G\left(Y^4Z - 2Y^2Z^3 + \frac{Z^5}{5}\right)$$
$$\text{with } G_0 = \frac{B_0}{R_0^4}$$

Outside fringe field regions, or everywhere in sharp edge decapole ($\lambda_E = \lambda_S = 0$) , $\vec{B}(X,Y,Z)$ in the magnet is given by

$$B_X = 0$$
$$B_Y = 4G_0(Y^2 - Z^2)YZ$$
$$B_Z = G_0(Y^4 - 6Y^2Z^2 + Z^4)$$

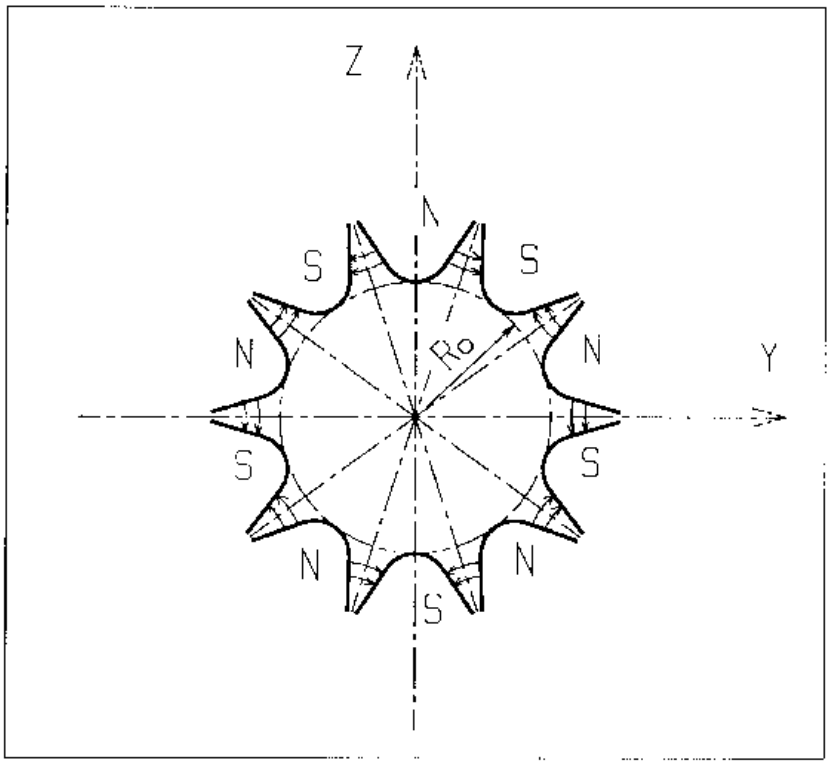


Figure 21: Decapole magnet

DIPOLE: Dipole magnet, polar frame

DIPOLE provides a model of a dipole field, possibly with transverse indices. The field along a particle trajectory is computed as the particle motion proceeds, straightforwardly from the dipole geometrical parametres.

These are handled as follows. The dimensionning of the magnet is defined by

AT : total angular aperture

RM : mean radius used for the positioning of field boundaries

The 2 or 3 effective field boundaries (EFB), from which the dipole field is drawn, are defined from geometric boundaries, the shape and position of which are determined by the following parameters.

ACENT : arbitrary inner angle, used for EFB's positioning

ω : azimuth of an EFB with respect to *ACENT*

θ : angle of an EFB with respect to its azimuth (wedge angle)

R_1, R_2 : radius of curvature of an EFB

U_1, U_2 : extent of the linear part of an EFB.

The magnetic field is calculated in polar coordinates. At any position (R, θ) along the particle trajectory the value of the vertical component of the mid-plane field is calculated by

$$B = \mathcal{F}(\mathcal{R}, \theta) * B_0 * \left(1 + N * \left(\frac{R - RM}{RM} \right) + B * \left(\frac{R - RM}{RM} \right)^2 + G * \left(\frac{R - RM}{RM} \right. \right. \tag{7.4.8}$$

where N, B and G are respectively the first, second and third order field indices and $\mathcal{F}(R, \theta)$ is the fringe field coefficient.

Calculation of the Fringe Field Coefficient

With each EFB a realistic extent of the fringe field, λ (normally equal to the gap size), is associated and a fringe field coefficient F is calculated. In the following λ stands for either λ_E (Entrance), λ_S (Exit) or λ_L (Lateral EFB).

F is an exponential type fringe field (Fig. 14) given by [19]

$$F = \frac{1}{1 + \exp P(s)}$$

wherein s is the distance to the EFB and depends on (R, θ), and

$$P(s) = C_0 + C_1 \left(\frac{s}{\lambda}\right) + C_2 \left(\frac{s}{\lambda}\right)^2 + C_3 \left(\frac{s}{\lambda}\right)^3 + C_4 \left(\frac{s}{\lambda}\right)^4 + C_5 \left(\frac{s}{\lambda}\right)^5$$

It is also possible to simulate a shift of the *EFB*, by giving a non zero value to the parameter *SHIFT*. s is then changed to $s - SHIFT$ in the previous equation. This allows small variations of the magnetic length.

Let F_E (respectively F_S, F_L) be the fringe field coefficient attached to the entrance (respectively exit, lateral) EFB. At any position on a trajectory the resulting value of the fringe field coefficient (eq. 7.4.9) is

$$\mathcal{F}(R, \theta) = F_E * F_S * F_L$$

($F_L = 1$ if no lateral EFB is requested).

Simulating Field Defects and Shims

Not provisionned in the present version.

Extrapolation Off Median Plane

The vector field $\vec{B}$ and its derivatives in the median plane are calculated by means of a second or fourth order polynomial interpolation, depending on the value of the parameter *IORDRE* (*IORDRE*=2, 25 or 4, see section 4.3.2). The transformation from polar to Cartesian coordinates is performed following eqs (4.3.9 or 4.3.10). Extrapolation off median plane is then performed by means of Taylor expansions, following the procedure described in section 4.2.2.

DIPOLE-M: Generation of dipole mid-plane 2-D map, polar frame

DIPOLE-M is a more recent, simpler and improved version of *AIMANT*.

The keyword *DIPOLE-M* provides an automatic generation of a dipole field map in polar coordinates. The extent of the map is defined by the following parameters, as shown in Figs. 13A and 13B.

AT : total angular aperture

RM : mean radius used for the positioning of field boundaries

RMIN, RMAX : minimum and maximum radii

The 2 or 3 effective field boundaries (EFB) inside the map are defined from geometric boundaries, the shape and position of which are determined by the following parameters.

ACENT : arbitrary inner angle, used for EFB's positioning

ω : azimuth of an EFB with respect to *ACENT*

θ : angle of an EFB with respect to its azimuth (wedge angle)

R_1, R_2 : radius of curvature of an EFB

U_1, U_2 : extent of the linear part of an EFB.

At any node of the map mesh, the value of the field is calculated as

$$B = \mathcal{F} * B_0 * \left(1 + N * \left(\frac{R - RM}{RM} \right) + B * \left(\frac{R - RM}{RM} \right)^2 + G * \left(\frac{R - RM}{RM} \right)^3 \right) \tag{7.4.9}$$

where N, B and G are respectively the first, second and third order field indices and $\mathcal{F}$ is the fringe field coefficient.

Calculation of the Fringe Field Coefficient

With each EFB a realistic extent of the fringe field, λ (normally equal to the gap size), is associated and a fringe field coefficient F is calculated. In the following λ stands for either λ_E (Entrance), λ_S (Exit) or λ_L (Lateral EFB).

F is an exponential type fringe field (Fig. 14) given by [19]

$$F = \frac{1}{1 + \exp P(s)}$$

where s is the distance to the EFB, and

$$P(s) = C_0 + C_1 \left(\frac{s}{\lambda}\right) + C_2 \left(\frac{s}{\lambda}\right)^2 + C_3 \left(\frac{s}{\lambda}\right)^3 + C_4 \left(\frac{s}{\lambda}\right)^4 + C_5 \left(\frac{s}{\lambda}\right)^5$$

It is also possible to simulate a shift of the *EFB*, by giving a non zero value to the parameter *SHIFT*. s is then changed to $s - SHIFT$ in the previous equation. This allows small variations of the total magnetic length.

Let F_E (respectively F_S, F_L) be the fringe field coefficient attached to the entrance (respectively exit, lateral) EFB. At any node of the map mesh, the resulting value of the fringe field coefficient (eq. 7.4.9) is

$$\mathcal{F} = F_E * F_S * F_L$$

($F_L = 1$ if no lateral EFB is requested).

The Mesh of the Field Map

The magnetic field is calculated at the nodes of a mesh with polar coordinates, in the median plane. The radial step is given by

$$\delta R = \frac{RMAX - RMIN}{IRMAX - 1}$$

and the angular step by

$$\delta\theta = \frac{AT}{IAMAX - 1}$$

where, *RMIN* and *RMAX* are the lower and upper radial limits of the field map, and AT is its total angular aperture (Fig. 13B). *IRMAX* and *IAMAX* are the total number of nodes in the radial and angular directions.

Simulating Field Defects and Shims

Once the initial map is calculated, it is possible to modify it by means of the parameter *NBS*, so as to simulate field defects or shims.

If NBS $= -2$, the map is globally modified by a perturbation proportional to $R - R_0$, where R_0 is an arbitrary radius, with an amplitude $\Delta B_Z/B_0$, so that B_Z at the nodes of the mesh is replaced by

$$B_Z * \left(1 + \frac{\Delta B_Z}{B_0} \frac{R - R_0}{RMAX - RMIN}\right)$$

If NBS $= -1$, the perturbation is proportional to $\theta - \theta_0$, and B_Z is replaced by

$$B_Z * \left(1 + \frac{\Delta B_Z}{B_0} \frac{\theta - \theta_0}{AT}\right)$$

If NBS ≥ 1, then *NBS* shims are introduced at positions $\frac{R_1 + R_2}{2}$, $\frac{\theta_1 + \theta_2}{2}$ (Fig. 16) [20]

The initial field map is modified by shims with second order profiles given by

$$\theta = \left(\gamma + \frac{\alpha}{\mu}\right) \beta \frac{X^2}{\rho^2}$$

where X is shown in Fig. 14, $\rho = \frac{R_1 + R_2}{2}$ is the central radius, α and γ are the angular limits of the shim, β and μ are parameters.

At each shim, the value of B_Z at any node of the initial map is replaced by

$$B_Z * \left(1 + F\theta * FR * \frac{\Delta B_Z}{B_0}\right)$$

where $F\theta = 0$ or $FR = 0$ outside the shim, and $F\theta = 1$ and $FR = 1$ inside.

Extrapolation Off Median Plane

The vector field $\vec{B}$ and its derivatives in the median plane are calculated by means of a second or fourth order polynomial interpolation, depending on the value of the parameter *IORDRE* (*IORDRE*=2, 25 or 4, see section 4.3.2). The transformation from polar to Cartesian coordinates is performed following eqs (4.3.9 or 4.3.10). Extrapolation off median plane is then performed by means of Taylor expansions, following the procedure described in section 4.2.2.

DIPOLES: Dipole magnet N-tuple, polar frame [25]

DIPOLES works much like *DIPOLE* as to the field modelling, yet with the particularity that it allows positioning up to 5 such dipoles within the angular sector with full aperture AT thus allowing accounting for overlapping fringe fields This is done in the following way[6]

The dimensionning of the magnet is defined by

AT : total angular aperture

RM : mean radius used for the positioning of field boundaries

For each one of the $N = 1$ to 5 dipoles of the N-tuple, the 2 effective field boundaries (entrance and exit EFBs) from which the dipole field is drawn (eq. 7.4.11) are defined from geometric boundaries, the shape and position of which are determined by the following parameters (in the same manner as in *DIPOLE, DIPOLE-M*) (see Fig. 13-A page141, and Fig. 22)

ACN_i : arbitrary inner angle, used for EFB's positioning

ω : azimuth of an EFB with respect to *ACENT*

θ : angle of an EFB with respect to its azimuth (wedge angle)

R_1, R_2 : radius of curvature of an EFB

U_1, U_2 : extent of the linear part of an EFB

Calculation of the Field Due to a Single Dipole

The magnetic field is calculated in polar coordinates. At all (R, θ) in the median plane ($z = 0$), the magnetic field due a single one (index i) of the dipoles of a N-tuple magnet is written

$$B_{zi}(R,\theta) = B_{z0,i}\,\mathcal{F}_i(R,\theta)\,\left(1 + b_{1_i}(R - RM_i)/RM_i + b_{2_i}(R - RM_i)^2/RM_i^2 + ...\right) \tag{7.4.10}$$

[6] *FFAG* can be referred to as another instance of a procedure based on such method.

wherein $B_{z0,i}$ is a reference field, at reference radius RM_i, and $\mathcal{F}(R,\theta)$ is the fringe field coefficient, see below. This field model is proper to simulate for instance chicane dipoles, isochronous or superconducting FFAG magnets, etc.

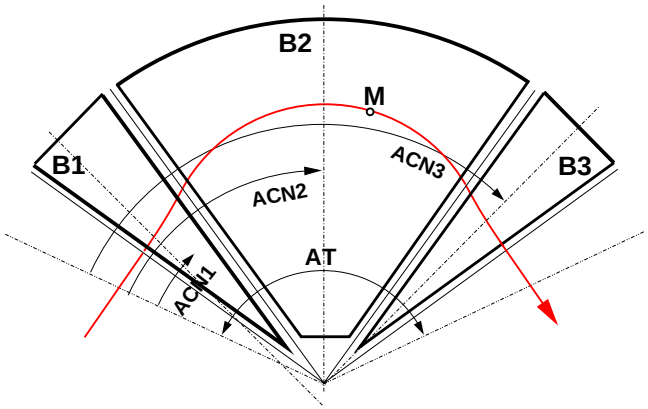


Figure 22: Definition of a dipole triplet using the *DIPOLES* or *FFAG* procedures.

Calculation of the Fringe Field Coefficient

In a dipole, with each EFB a realistic extent of the fringe field, g, is associated and a fringe field coefficient F is calculated.

F is an exponential type fringe field (Fig. 14, page 144) given by [19]

$$F = \frac{1}{1+\exp P(d)}$$

wherein d is the distance to the EFB and depends on (R,θ), and

$$P(d) = C_0 + C_1\left(\frac{d}{g}\right) + C_2\left(\frac{d}{g}\right)^2 + C_3\left(\frac{d}{g}\right)^3 + C_4\left(\frac{d}{g}\right)^4 + C_5\left(\frac{d}{g}\right)^5$$

In addition to that, g is made dependent of R (a way to simulate the effect of variable gap size on field fall-off), under the form

$$g(R) = g_0\,(RM/R)^{\kappa}$$

Let F_E (respectively F_S) be the fringe field coefficient attached to the entrance (respectively exit) EFB ; at any position on a trajectory the resulting value of the fringe field coefficient is taken to be

$$\mathcal{F}_i(R,\theta) = F_E * F_S \tag{7.4.11}$$

Full field at arbitrary position

Now, accounting for N neighboring dipoles in an N-tuple, the mid-plane field and field derivatives are obtained by addition of the contributions of the N dipoles taken separately, namely

$$\begin{aligned} B_z(R,\theta) &= \sum_{i=1,N} B_{zi}(R,\theta) = \sum_{i=1,N} B_{z0,i}\,\mathcal{F}_i(R,\theta)\,\mathcal{R}_i(r) \\ \frac{\partial^{k+l}\vec{B}_z(R,\theta)}{\partial\theta^k\partial r^l} &= \sum_{i=1,N} \frac{\partial^{k+l}\vec{B}_{zi}(R,\theta)}{\partial\theta^k\partial r^l} \end{aligned} \tag{7.4.12}$$

with $\mathcal{R}_i(R)$ as defined in Eq. 7.4.10. Note that, in doing so it is not meant that field superposition does apply in reality, it is just meant to provide the possibility of obtaining a realistic field shape, that would for instance closely match (using adequate $C_0 - C_5$ sets of coefficients) 3-D field simulations obtained from magnet codes.

Calculation of the mid-plane field derivatives

Two methods have been implemented to calculate the field derivatives in the median plane (Eq. 7.4.12), based on either analytical expressions derived from the magnet geometrical description, or classical numerical interpolation.
The first method has the merit of insuring best symplecticity in principle and fastest tracking. The interest of the second method is in its facilitating possible changes in the mid-plane magnetic field model $B_z(R,\theta)$, for instance if simulations of shims, defects, or special R,θ field dependence need to be introduced.

Analytical method :

The starting ingredients are, on the one hand distances to the EFBs,

$$d(R,\theta) = \sqrt{(x(R,\theta) - x_0(R,\theta))^2 + (y(R,\theta) - y_0(R,\theta))^2}$$

to be computed for the two cases d_{Entrance}, d_{Exit}, and on the other hand the expressions of the coordinates of particle position M and its projection P on the EFB in terms of the magnet geometrical parameters, namely

$$\begin{aligned} x(R,\theta) &= \cos(ACN-\theta) - RM \\ y(R,\theta) &= R\sin(ACN-\theta) \\ x_P(R,\theta) &= \sin(u)\,(y(R,\theta)-y_b)/2 + x_b\ \sin^2(u) + x(R,\theta)\ \cos^2(u) \\ y_P(R,\theta) &= \sin(u)\,(x(R,\theta)-x_b)/2 + y_b\ \cos^2(u) + y(R,\theta)\ \sin^2(u) \end{aligned}$$

with $x_b,\ y_b,\ u$ parameters drawn from the magnet geometry (sector angle, wedge angle, face curvatures, etc.).
These ingredients allow calculating the derivatives $\frac{\partial^{u+v}x(R,\theta)}{\partial\theta^u\partial r^v}$, $\frac{\partial^{u+v}y(R,\theta)}{\partial\theta^u\partial r^v}$, $\frac{\partial^{u+v}x_0(R,\theta)}{\partial\theta^u\partial r^v}$, $\frac{\partial^{u+v}y_0(R,\theta)}{\partial\theta^u\partial r^v}$, which, in turn, intervene in the derivatives of the compound functions $\frac{\partial^{u+v}F(R,\theta)}{\partial\theta^u\partial r^v}$, $\frac{\partial^{u+v}p(R,\theta)}{\partial\theta^u\partial r^v}$, $\frac{\partial^{u+v}d(R,\theta)}{\partial\theta^u\partial r^v}$.

Interpolation method :

The expression $B_z(R,\theta)$ in Eq. 7.4.12 is, in this case, computed at the $n\times n$ nodes ($n = 3$ or 5 in practice) of a "flying" interpolation grid in the median plane centered on the projection m_0 of the actual particle position M_0 as schemed in Fig. 23. A polynomial interpolation is involved, of the form

$$B_z(R,\theta) = A_{00} + A_{10}\theta + A_{01}r + A_{20}\theta^2 + A_{11}\theta r + A_{02}r^2$$

that yields the requested derivatives, using

$$A_{kl} = \frac{1}{k!l!}\frac{\partial^{k+l}B}{\partial\theta^k\partial r^l}$$

Note that, the source code contains the explicit analytical expressions of the coefficients A_{kl} solutions of the normal equations, so that the operation is not so CPU time consuming.

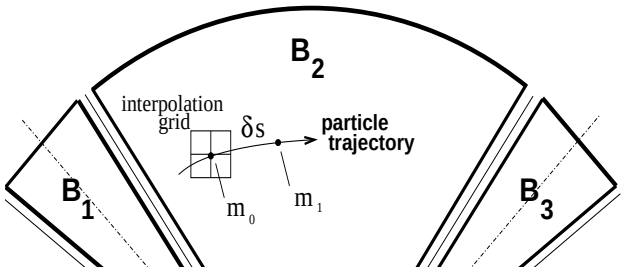


Figure 23: Interpolation method. m_0 and m_1 are the projections in the median plane of particle positions M_0 and M_1 separated by one integration step δs.

The vector field $\vec{B}$ and its derivatives in the median plane are calculated by means of a second or fourth order polynomial interpolation, depending on the value of the parameter *IORDRE* (*IORDRE*=2, 25 or 4, see section 4.3.2). The transformation from polar to Cartesian coordinates is performed following eqs (4.3.9 or 4.3.10). Extrapolation off median plane is then performed by means of Taylor expansions, following the procedure described in section 4.2.2.

DODECAPO: Dodecapole magnet (Fig. 24)

The meaning of parameters for *DODECAPO* is the same as for *QUADRUPO*.

In fringe field regions the magnetic field $\vec{B}(X,Y,Z)$ and its derivatives up to fourth order are derived from the scalar potential approximated to the 6th order in Y and Z

$$V(X,Y,Z) = G\left(Y^4 - \frac{10}{3}Y^2Z^2 + Z^4\right)YZ$$
$$\text{with } G_0 = \frac{B_0}{R_0^5}$$

Outside fringe field regions, or everywhere in sharp edge dodecapole ($\lambda_E = \lambda_S = 0$) , $\vec{B}(X,Y,Z)$ in the magnet is given by

$$B_X = 0$$
$$B_Y = G_0(5Y^4 - 10Y^2Z^2 + Z^4)Z$$
$$B_Z = G_0(Y^4 - 10Y^2Z^2 + 5Z^4)Y$$

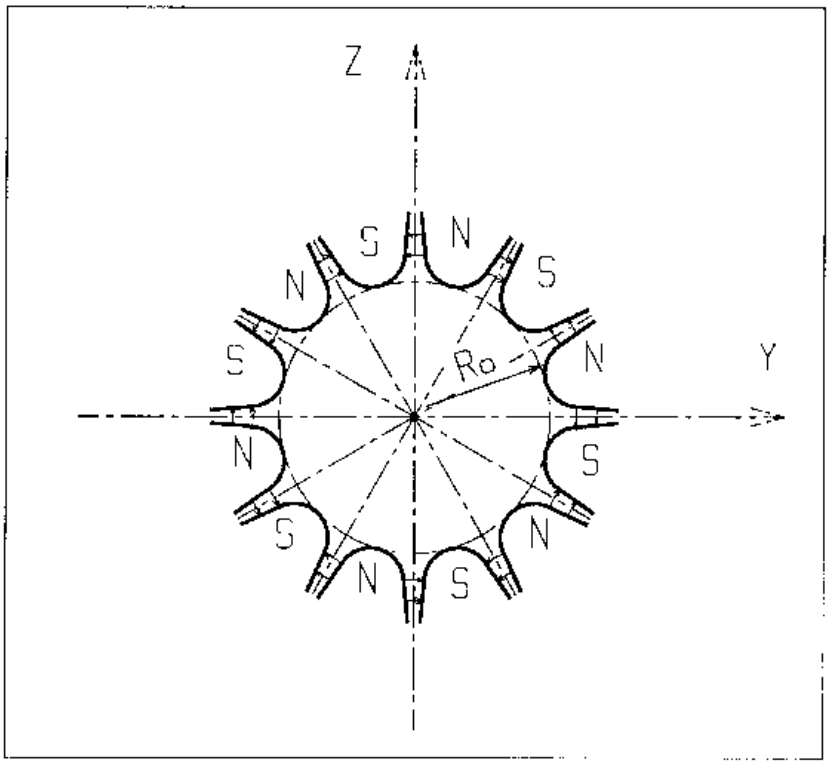


Figure 24: Dodecapole magnet

DRIFT or ESL: Field free drift space

DRIFT or *ESL* allow introduction of a drift space with length XL with positive or negative sign, anywhere in a structure. The associated equations of motion are (Fig. 25)

$$Y_2 = Y_1 + XL * \mathrm{tg}T$$
$$Z_2 = Z_1 + \frac{XL}{\cos T}\,\mathrm{tg}P$$
$$SAR_2 = SAR_1 + \frac{XL}{\cos T * \cos P}$$

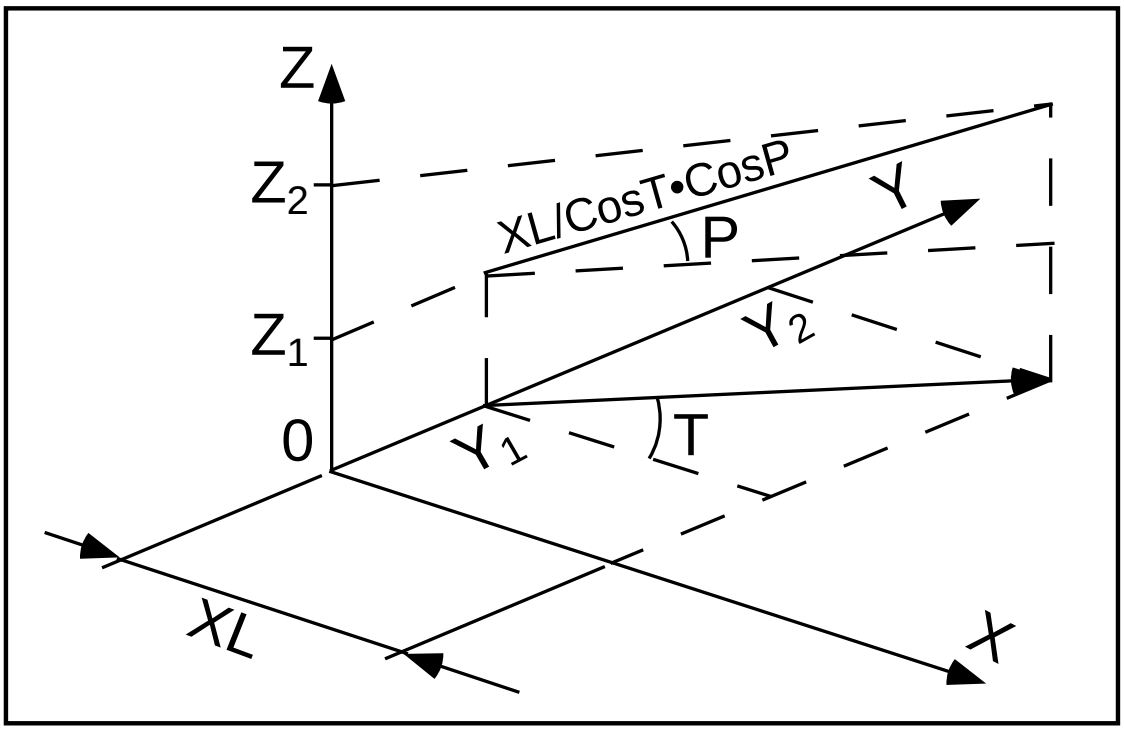


Figure 25: Transfer of particles in a drift space.

EBMULT: Electro-magnetic multipole

EBMULT simulates an electro-magnetic multipole, by addition of electric $(\vec{E})$ and magnetic $(\vec{B})$ multipole components (dipole to 20-pole). $\vec{E}$ and its derivatives $\frac{\partial^{i+j+k}\vec{E}}{\partial X^i \partial Y^j \partial Z^k}$ $(i+j+k \leq 4)$ are derived from the general expression of the multipole scalar potential (eq. 4.2.5), followed by a $\frac{\pi}{2n}$ rotation ($n =$ pole order), as described in section 4.4.4 (see also *ELMULT*). $\vec{B}$ and its derivatives are derived from the same general potential, as described in section 4.2.6 (see also *MULTIPOL*).

The entrance and exit fringe fields of the $\vec{E}$ and $\vec{B}$ components are treated separately, in the same way as described under *ELMULT* and *MULTIPOL*, for each one of these two fields. Wedge angle correction is applied in sharp edge field model if $\vec{B}1$ is non zero, as in *MULTIPOL*. Any of the $\vec{E}$ or $\vec{B}$ multipole field component can be rotated independently of the others.

Use *PARTICUL* prior to *EBMULT*, for the definition of particle mass and charge.

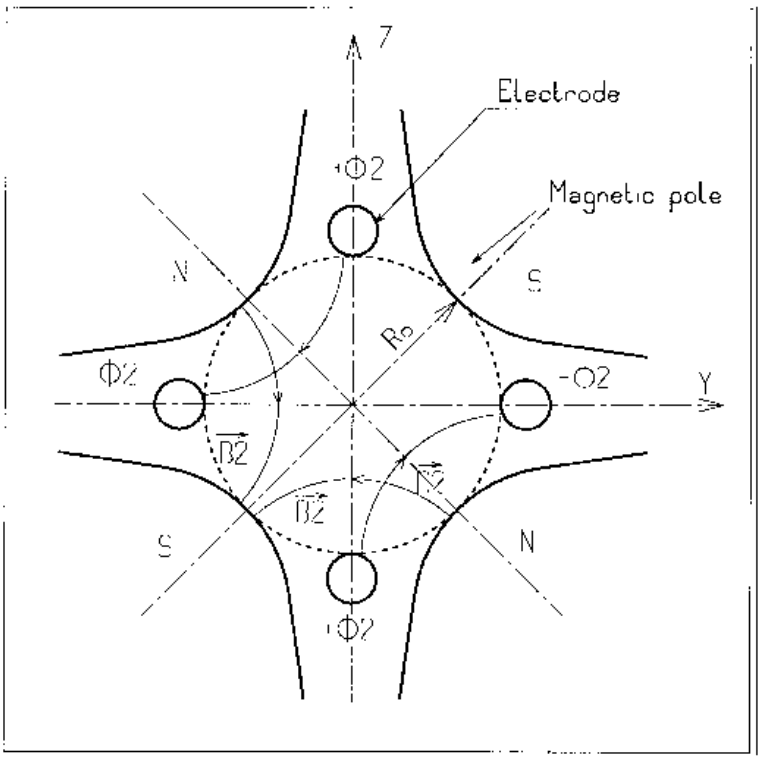


Figure 26: An example of $\vec{E}$, $\vec{B}$ multipole: the achromatic quadrupole
(known for its allowing null second order chromatic aberrations [21]).

EL2TUB: Two-tube electrostatic lens

The lens is cylindrically symmetric about the X-axis.
The length and potential of the first (resp. second) electrode are $X1$ and $V1$ ($X2$ and $V2$). The distance between the two electrodes is D, and their inner radius is R_0 (Fig. 27). X-axis cylindrical symmetry is assumed. The model for the electrostatic potential along the axis is [23]

$$V(X) = \frac{V_2 - V_1}{2} \,\mathrm{th}\, \frac{\omega x}{R_0} \left[+ \frac{V_1 + V_2}{2} \right] \qquad \text{if } D = 0$$

$$V(X) = \frac{V_2 - V_1}{2} \, \frac{1}{2\omega D} \, \ell n \, \frac{\mathrm{ch}\, \omega \dfrac{x + D}{R_0}}{\mathrm{ch}\, \omega \dfrac{x - D}{R_0}} \left[+ \frac{V_1 + V_2}{2} \right] \qquad \text{if } D \neq 0$$

(x = distance from half-way between the electrodes; ω = 1.318; th = hyperbolic tangent; ch = hyperbolic cosine) from which the field $\vec{E}(X, Y, Z)$ and its derivatives are derived following the procedure described in section 4.4.2 (note that they don't depend on the constant term $\left[\dfrac{V_1 + V_2}{2} \right]$ which disappears when differentiating).
Use *PARTICUL* prior to *EL2TUB*, for the definition of particle mass and charge.

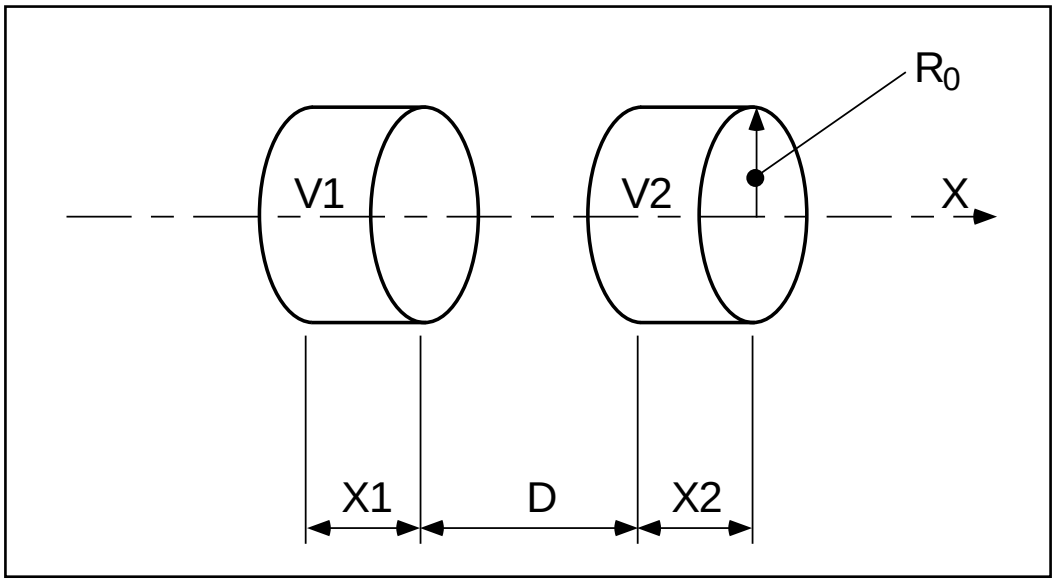


Figure 27: Two-electrode cylindrical electric lens.

ELMIR: Electrostatic N-electrode mirror/lens, straight slits

The device works as mirror or lens, horizontal or vertical. It is made of N 2-plate electrodes and has mid-plane symmetry.

Electrode lengths are $L1$, $L2$, ..., LN. D is the mirror/lens gap. The model for the Y-independent electrostatic potential is (after Ref. [24, p.412])

$$V(X,Z) = \sum_{i=2}^{N} \frac{Vi - Vi-_1}{\pi} \arctan \frac{\sinh(\pi(X - Xi-_1)/D)}{\cos(\pi Z/D)}$$

where Vi are the potential at the N electrodes (and normally $V1 = 0$ refers to the incident beam energy), Xi are the locations of the slits, X is the distance from the origin taken at the first slit (located at $X1 \equiv 0$ between the first and second electrodes). From $V(X,Z)$ the field $\vec{E}(X,Y,Z)$ and derivatives are deduced following the procedure described in section 4.4.4 (page 69).

The total X-extent of the mirror/lens is $L = \sum_{i=1}^{N} Li$.

In the mirror mode (*i.e.*, option flag $MT = 11$ for vertical mid-plane or 12 for horizontal mid-plane) stepwise integration starts at $X = -L1$ (entrance of the first electrode) and terminates either when back to $X = -L1$ or when reaching $X = L - L1$ (end of the $N - th$ electrode). In the latter case particles are stopped with their index IEX set to -8 (see section 7.6.8 on page 246). Normally $X1$ should exceed $3D$ (possibly sensibly, so that $V(X < X1)$ have negligible effect in terms of trajectory behavior).

In the lens mode (*i.e.*, option flag $MT = 21$ for vertical mid-plane or 22 for horizontal mid-plane) stepwise integration starts at $X = -L1$ (entrance of the first electrode) and terminates either when reaching $X = L - L1$ (end of the $N - th$ electrode) or when the particle deflection exceeds $\pi/2$. In the latter case the particle is stopped with their index IEX set to -3.

Use *PARTICUL* prior to *ELMIR*, for the definition of particle mass and charge.

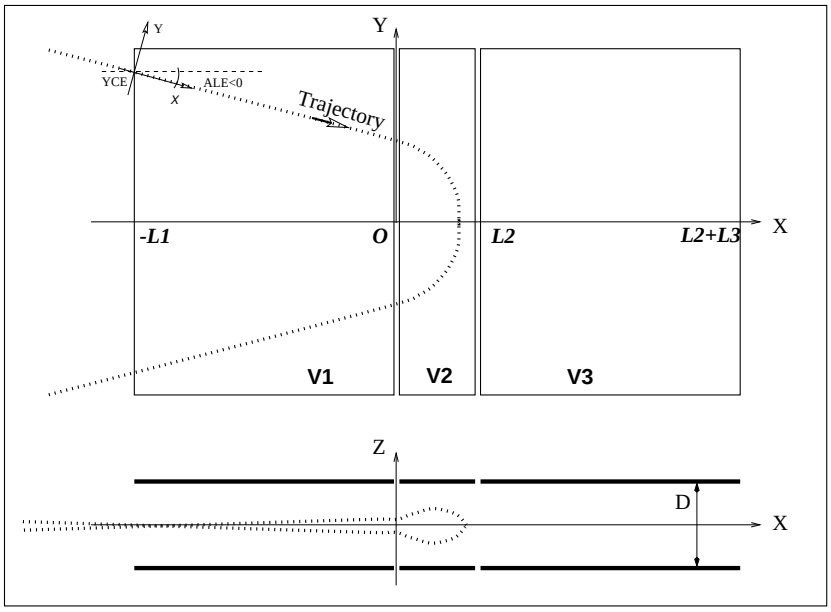


Figure 28: Electrostatic N-electrode mirror/lens, straight slits, in the case $N = 3$, in horizontal mirror mode ($MT = 11$).
Possible non-zero entrance quantities YCE, ALE should be specified using *CHANGREF*, or using *KPOS*=3 with *YCE*=pitch, *ALE*=half-deviation.

ELMIRC: Electrostatic N-electrode mirror/lens, circular slits [24]

The device works as mirror or lens, horizontal or vertical. It is made of N 2-plate electrodes and has mid-plane symmetry[7].

Electrode slits are circular, concentric with radii $R1$, $R2$, ..., R_{N-1}, D is the mirror/lens gap. The model for the mid-plane ($Z = 0$) radial electrostatic potential is (after Ref. [24, p.443])

$$V(r) = \sum_{i=2}^{N} \frac{Vi - Vi_{-1}}{\pi} \arctan\left(\sinh \frac{\pi(r - Ri_{-1})}{D}\right)$$

where Vi are the potential at the N electrodes (and normally $V1 = 0$ refers to the incident beam energy). r is the current radius.

The mid-plane field $\vec{E}(r)$ and its r-derivatives are first derived by differentiation, then $\vec{E}(r, Z)$ and derivatives are obtained from Taylor expansions and Maxwell relations. Eventually a transformation to the rotating frame provides $\vec{E}(X, Y, Z)$ and derivatives as involved in eq. 4.1.12.

Stepwise integration starts at entrance (defined by RE, TE) of the first electrode and terminates when rotation of the reference rotating frame (RM, X, Y) has reached the value AT. Normally, $R1 - RE$ and $R1 - RS$ should both exceed $3D$ (possibly sensibly, so that $V(r < RE)$ and $V(r < RS)$ have negligible effect in terms of trajectory tails).

Positioning of the element is performed by means of *KPOS* (see section 7.6.5).

Use *PARTICUL* prior to *ELMIRC*, for the definition of particle mass and charge.

[7] NOTE : in the present version of the code, the sole horizontal mirror mode is operational, and N is limited to 3.

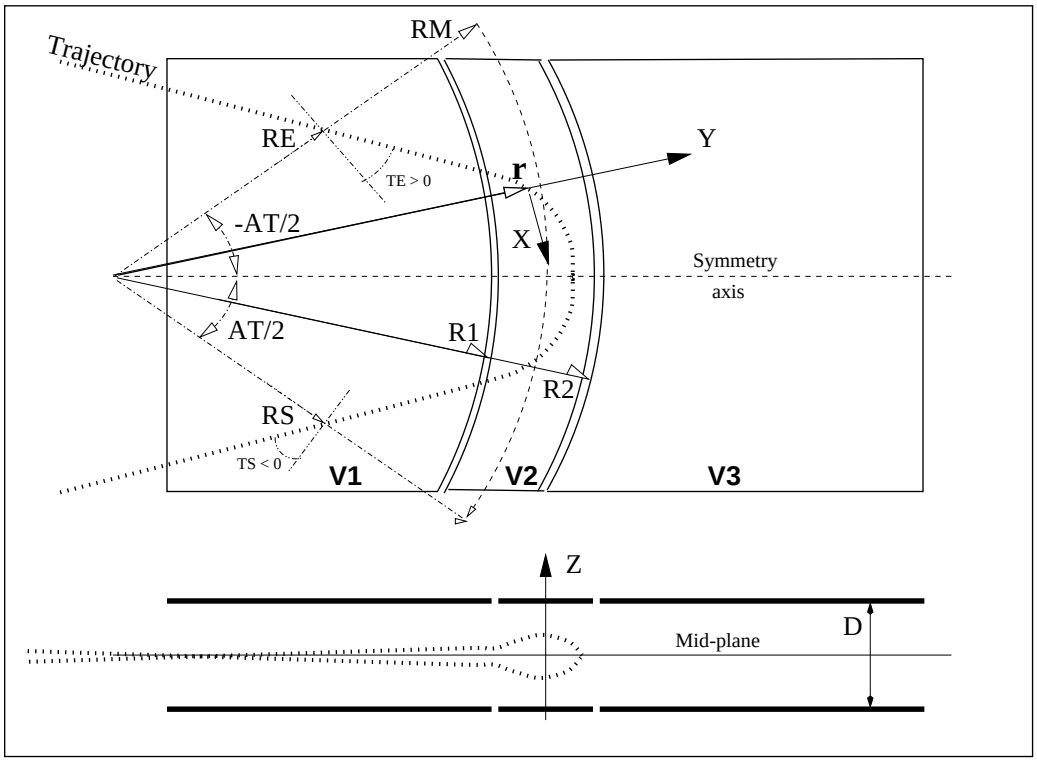


Figure 29: Electrostatic N-electrode mirror/lens, circular slits, in the case $N = 3$, in horizontal mirror mode.

ELMULT: Electric multipole

The simulation of multipolar electric field $\vec{M}_E$ proceeds by addition of the dipolar $(\vec{E}1)$, quadrupolar ($\vec{E}2$), sextupolar ($\vec{E}3$), etc., up to 20-polar $(\vec{E}10)$ components, and of their derivatives up to fourth order, following

$$\vec{M}_E = \vec{E}1 + \vec{E}2 + \vec{E}3 + \ ... \ + \vec{E}10$$
$$\frac{\partial \vec{M}_E}{\partial X} = \frac{\partial \vec{E}1}{\partial X} + \frac{\partial \vec{E}2}{\partial X} + \frac{\partial \vec{E}3}{\partial X} + \ ... \ + \frac{\partial \vec{E}10}{\partial X}$$
$$\frac{\partial^2 M_E}{\partial X \partial Z} = \frac{\partial^2 \vec{E}1}{\partial X \partial Z} + \frac{\partial^2 \vec{E}2}{\partial X \partial Z} + \frac{\partial^2 \vec{E}3}{\partial X \partial Z} + \ ... \ + \frac{\partial^2 \vec{E}10}{\partial X \partial Z}$$
etc.

The independent components $\vec{E}1$ to $\vec{E}10$ and their derivatives up to the second order are calculated by differentiating the general multipole potential given in eq. 4.2.5 (page 52), followed by a $\frac{\pi}{2n}$ rotation about the X-axis, so that the so defined right electric multipole of order n, and of strength [21, 22]

$$K_n = \frac{1}{2} \frac{\gamma}{\gamma^2 - 1} \frac{V_n}{R_0^n}$$

(V_n = potential at the electrode, R_0 = radius at pole tip, γ = relativistic Lorentz factor of the particle) has the same focusing effect than the right magnetic multipole of order n and strength $K_n = \frac{B_n}{R_0^{n-1} B\rho}$ (B_n = field at pole tip, $B\rho$ = particle rigidity, see *MULTIPOL*).
Such $\frac{\pi}{2n}$ rotation of the multipole components is obtained following the procedure described in section 4.4.4.

The entrance and exit fringe fields are treated separately. They are characterized by the integration zone X_E at entrance and X_S at exit, as for *QUADRUPO*, and by the extent λ_E at entrance, λ_S at exit. The fringe field extents for the dipole component are λ_E and λ_S. The fringe field for the quadrupolar (sextupolar, ..., 20-polar) component is given by a coefficient E_2 (E_3, ..., E_{10}) at entrance, and S_2 (S_3, ..., S_{10}) at exit, such that the fringe field extent is $\lambda_E * E_2$ ($\lambda_E * E_3$, ..., $\lambda_E * E_{10}$) at entrance and $\lambda_S * S_2$ ($\lambda_S * S_3$, ..., $\lambda_S * S_{10}$) at exit.

If $\lambda_E = 0$ ($\lambda_S = 0$) the multipole lens is considered to have a sharp edge field

at entrance (exit), and then, X_E (X_S) is forced to zero (for the mere purpose of saving computing time).

If $E_i = 0$ ($S_i = 0$) ($i = 2, 10$), the entrance (exit) fringe field for multipole component i is considered as a sharp edge field.

Overlapping of fringe fields inside the element is treated separately for each component, in the way described in *QUADRUPO*.

Moreover, any multipole component $\vec{E}i$ can be rotated independently by an angle RXi around the longitudinal X-axis, for the simulation of positioning defects, as well as skewed lenses.

Use *PARTICUL* prior to *ELMULT*, for the definition of particle mass and charge.

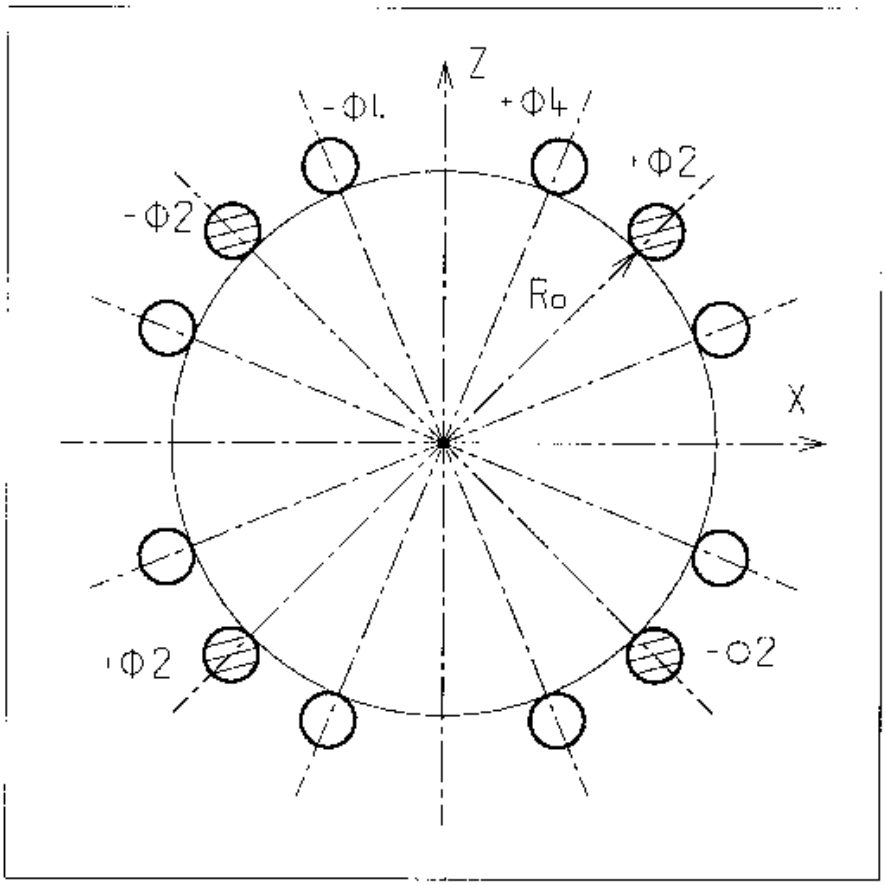


Figure 30: An electric multipole combining skew-quadrupole ($\vec{E}2 \neq \vec{0},\ \vec{R}2 = \pi/4$) and skew-octupole ($\vec{E}4 \neq \vec{0},\ \vec{R}4 = \pi/8$) components ($\vec{E}1 = \vec{E}3 = \vec{E}5 = \ ... \ = \vec{E}10 = \vec{0}$) [22].

ELREVOL: 1-D uniform mesh electric field map

ELREVOL reads a 1-D axial field map from a storage data file, whose content must fit the following *FORTRAN* reading sequence

```
OPEN (UNIT = NL, FILE = FNAME, STATUS = 'OLD' [,FORM='UNFORMATTED'])
DO 1 I=1, IX
   IF (BINARY) THEN
      READ(NL) X(I), EX(I)
   ELSE
      READ(NL,*) X(I), EX(I)
   ENDIF
  1     CONTINUE
```

where IX is the number of nodes along the (symmetry) X-axis, $X(I)$ their coordinates, and $EX(I)$ the values of the X component of the field. EX is normalized with *ENORM* prior to ray-tracing. As well the longitudinal coordinate X is normalized with a *XNORM* coefficient (usefull to convert to centimeters, the working units in **zgoubi** .

X-cylindrical symmetry is assumed, resulting in EY and EZ taken to be zero on axis. $\vec{E}(X,Y,Z)$ and its derivatives along a particle trajectory are calculated by means of a 5-points polynomial fit followed by second order off-axis Taylor series extrapolation (see sections 4.4.1 and 4.5).

Entrance and/or exit integration boundaries may be defined in the same way as in *CARTEMES* by means of the flag ID and coefficients A, B, C, A', B', C'.

Use *PARTICUL* prior to *ELREVOL*, for the definition of particle mass and charge.

FFAG: FFAG magnet, N-tuple [25]

FFAG works much like *DIPOLES* as to the field modelling, apart from the (so-called "scaling") radial dependence of the field.

The *FFAG* procedure allows overlapping of fringe fields of neighboring dipoles, thus simulating in some sort the field in a dipole N-tuple - as for instance in an FFAG doublet or triplet. This is done in the way described below.

The dimensionning of the magnet is defined by

AT : total angular aperture

RM : mean radius used for the positioning of field boundaries

For each one of the $N = 1$ to (maximum) 5 dipoles of the N-tuple, the 2 effective field boundaries (entrance and exit EFBs) from which the dipole field is drawn are defined from geometric boundaries, the shape and position of which are determined by the following parameters (in the same manner as in *DIPOLE*, *DIPOLE-M*) (see Fig. 13-A page 141, and Fig. 31)

ACN_i : arbitrary inner angle, used for EFB's positioning

ω : azimuth of an EFB with respect to *ACENT*

θ : angle of an EFB with respect to its azimuth (wedge angle)

R_1, R_2 : radius of curvature of an EFB

U_1, U_2 : extent of the linear part of an EFB

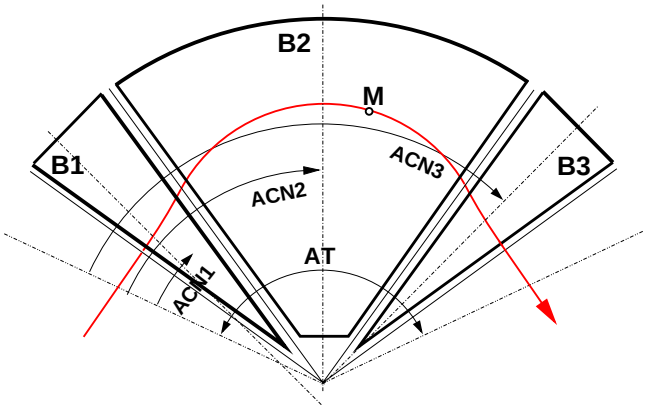


Figure 31: Definition of a dipole triplet using the *DIPOLES* or *FFAG* procedures.

Calculation of the Field Due to a Single Dipole

The magnetic field is calculated in polar coordinates. At all (R, θ) in the median plane ($z = 0$), the magnetic field due a single one (index i) of the dipoles of a N-tuple FFAG magnet is written

$$B_{zi}(R, \theta) = B_{z0,i}\, \mathcal{F}_i(R, \theta)\ (R/R_M)^{K_i}$$

wherein $B_{z0,i}$ is a reference field, at reference radius RM_i, whereas $\mathcal{F}(R, \theta)$ is calculated as described below.

Calculation of $\mathcal{F}_i(R, \theta)$

The fringe field coefficient $\mathcal{F}_i(R, \theta)$ associated with a dipole is computed as in the procedure *DIPOLES* (eq. 7.4.11), including radial dependence of the gap size

$$g(R) = g_0\, (RM/R)^{\kappa} \tag{7.4.13}$$

so to simulate the effect of gap shaping of $B_{zi}(R, \theta)$ on field fall-off, over the all radial extent of a scaling FFAG dipole (with normally - but not in practice - $\kappa = K_i$).

Full field at arbitrary position

For the rest, namely, calculation of the full field at arbitrary position, analytical calculation or numerical interpolation of the mid-plane field derivatives,

extrapolation off median plane, etc., things are performed exactly as in the case of the *DIPOLES* procedure (see page 173).

FFAG-SPI: Spiral FFAG magnet, N-tuple

FFAG-SPI works much like *FFAG* as to the field modelling, apart from the axial dependence of the field.

The *FFAG* procedure allows overlapping of fringe fields of neighboring dipoles, thus simulating in some sort the field in a dipole N-tuple - as for instance in an FFAG doublet or triplet. This is done in the way described below.

The dimensionning of the magnet is defined by

AT : total angular aperture

RM : mean radius used for the positioning of field boundaries

For each one of the $N = 1$ to (maximum) 5 dipoles of the N-tuple, the 2 effective field boundaries (entrance and exit EFBs) from which the dipole field is drawn are defined from geometric boundaries, the shape and position of which are determined by the following parameters (in the same manner as in *DIPOLE*, *DIPOLE-M*) (see Fig. 13-A page 141, and Fig. 32)

ACN_i : arbitrary inner angle, used for EFB's positioning

ω : azimuth of an EFB with respect to *ACENT*

ξ : spiral angle

Figure 32: Definition of a dipole triplet using the *DIPOLES* or *FFAG* procedures.

Calculation of the Field Due to a Single Dipole

The magnetic field is calculated in polar coordinates. At all (R, θ) in the median plane ($z = 0$), the magnetic field due a single one (index i) of the dipoles of a N-tuple FFAG magnet is written

$$B_{zi}(R, \theta) = B_{z0,i}\, \mathcal{F}_i(R, \theta)\, (R/R_M)^{K_i}$$

wherein $B_{z0,i}$ is a reference field, at reference radius RM_i, whereas $\mathcal{F}(R,\theta)$ is calculated as described below.

Calculation of $\mathcal{F}_i(R,\theta)$

The fringe field coefficient $\mathcal{F}_i(R,\theta)$ associated with a dipole is computed as in the procedure *DIPOLES* (eq. 7.4.11), including radial dependence of the gap size

$$g(R) = g_0 \left(RM/R\right)^{\kappa} \tag{7.4.14}$$

so to simulate the effect of gap shaping of $B_{zi}(R,\theta)$ on field fall-off, over the all radial extent of a scaling FFAG dipole (with normally - but not in practice - $\kappa = K_i$).

Full field at arbitrary position

For the rest, namely, calculation of the full field at arbitrary position, analytical calculation or numerical interpolation of the mid-plane field derivatives, extrapolation off median plane, etc., things are performed exactly as in the case of the *DIPOLES* procedure (see page 173).

MAP2D: 2-D Cartesian uniform mesh field map - arbitrary magnetic field [26]

MAP2D reads a 2-D field map that provides the three components B_X, B_Y, B_Z of the magnetic field at all nodes of a 2-D Cartesian uniform mesh in an (X, Y) plane. No particular symmetry is assumed, which allows the treatment of any type of field (e.g., dipole field with arbitrary Z elevation - the map needs not be a mid-plane map, solenoidal field, etc.). The data file should be filed with a format that fits the following *FORTRAN* reading sequence (presumably compatible with *TOSCA* code outputs)

```
      OPEN (UNIT = NL, FILE = FNAME, STATUS = 'OLD' [,FORM='UNFORMATTED'])
       DO 1 J=1,JY
          DO 1 I=1,IX
             IF (BINARY) THEN
                READ(NL) Y(J), Z(1), X(I), BY(I,J), BZ(I,J), BX(I,J)
             ELSE
                READ(NL,100) Y(J), Z(1), X(I), BY(I,J), BZ(I,J), BX(I,J)
100             FORMAT (1X, 6E11.4)
             ENDIF
      1        CONTINUE
```

where IX (JY) is the number of longitudinal (transverse horizontal) nodes of the 2-D uniform mesh, $Z(1)$ is the considered Z-elevation of the map. For binary files, FNAME must begin with 'B_' or 'b_', a flag 'BINARY' will thus be set to '.TRUE.'. The field $\vec{B} = (B_X, B_Y, B_Z)$ is next normalized with BNORM, prior to ray-tracing. As well the coordinates X, Y are normalized with *X-,Y-NORM* coefficients (usefull to convert to centimeters, the working units in **zgoubi** .

At each step of the trajectory of a particle, the field and its derivatives are calculated by a polynomial interpolation followed by a Z extrapolation (see sections 4.2.3, 4.3.3). Entrance and/or exit integration boundaries may be defined, in the same way as for *CARTEMES*.

MAP2D-E: 2-D Cartesian uniform mesh field map - arbitrary electric field

MAP2D-E reads a 2-D field map that provides the three components E_X, E_Y, E_Z of the electric field at all nodes of a 2-D Cartesian uniform mesh in an (X, Y) plane. No particular symmetry is assumed, which allows the treatment of any type of field (e.g., field of a parallel-plate mirror with arbitrary Z elevation - the map needs not be a mid-plane map). The data file should be filed with a format that fits the following *FORTRAN* reading sequence

```
      OPEN (UNIT = NL, FILE = FNAME, STATUS = 'OLD' [,FORM='UNFORMATTED'])
       DO 1 J=1,JY
          DO 1 I=1,IX
             IF (BINARY) THEN
                READ(NL) Y(J), Z(1), X(I), EY(I,J), EZ(I,J), EX(I,J)
             ELSE
                READ(NL,100) Y(J), Z(1), X(I), EY(I,J), EZ(I,J), EX(I,J)
100             FORMAT (1X, 6E11.4)
             ENDIF
      1        CONTINUE
```

where IX (JY) is the number of longitudinal (transverse horizontal) nodes of the 2-D uniform mesh, $Z(1)$ is the considered Z-elevation of the map. For binary files, FNAME must begin with 'E_' or 'b_', a flag 'BINARY' will thus be set to '.TRUE.'. The field $\vec{E} = (E_X, E_Y, E_Z)$ is next normalized with ENORM, prior to ray-tracing. As well the coordinates X, Y re normalized with *X-,Y-NORM* coefficients (usefull to convert to centimeters, the working units in **zgoubi** .

At each step of the trajectory of a particle, the field and its derivatives are calculated by a polynomial interpolation followed by a Z extrapolation (see sections 4.2.3, 4.3.3). Entrance and/or exit integration boundaries may be defined, in the same way as for *CARTEMES*.

MARKER: Marker

MARKER does nothing. Just a marker. No data.

As any other keyword, *MARKER* is allowed two *LABEL*s. Using '.plt' as a second *LABEL* will cause storage of current coordinates into zgoubi.plt

MATPROD: Matrix transfer

MATPROD performs a matrix transfer of the particle coordinates in the following way

$$X_i = \sum_j R_{ij} X_j^0 + \sum_{j,k} T_{ijk} X_j^0 X_k^0$$

where, X_i stands for any of the current coordinates Y, T, Z, P, path length and dispersion, and X_i^0 stands for any of the initial coordinates. $[R_{ij}]$ ($[T_{ijk}]$) is the first order (second order) transfer matrix as usually involved in second order beam optics [17]. Second order transfer is optional. The length of the element represented by the matrix may be introduced for the purpose of path length updating. Note : *MATRIX* delivers $[R_{ij}]$ and $[T_{ijk}]$ matrices in a format suitable for straightforward use with *MATPROD*.

MULTIPOL: Magnetic multipole

The simulation of multipolar magnetic field $\vec{M}$ by *MULTIPOL* proceeds by addition of the dipolar ($\vec{B}1$), quadrupolar ($\vec{B}2$), sextupolar ($\vec{B}3$), etc., up to 20-polar ($\vec{B}10$) components, and of their derivatives up to fourth order, following

$$\vec{M} = \vec{B}1 + \vec{B}2 + \vec{B}3 + \ ... \ + \vec{B}10$$
$$\frac{\partial \vec{M}}{\partial X} = \frac{\partial \vec{B}1}{\partial X} + \frac{\partial \vec{B}2}{\partial X} + \frac{\partial \vec{B}3}{\partial X} + \ ... \ + \frac{\partial \vec{B}10}{\partial X}$$
$$\frac{\partial^2 \vec{M}}{\partial X \partial Z} = \frac{\partial^2 \vec{B}1}{\partial X \partial Z} + \frac{\partial^2 \vec{B}2}{\partial X \partial Z} + \frac{\partial^2 \vec{B}3}{\partial X \partial Z} + \ ... \ + \frac{\partial^2 \vec{B}10}{\partial X \partial Z}$$
etc.

The independent components $\vec{B}1$, $\vec{B}2$, $\vec{B}3$, ..., $\vec{B}10$ and their derivatives up to the fourth order are calculated as described in section 4.2.6.

The entrance and exit fringe fields are treated separately. They are characterized by the integration zone X_E at entrance and X_S at exit, as for *QUADRUPO*, and by the extent λ_E at entrance, λ_S at exit. The fringe field extents for the dipole component are λ_E and λ_S. The fringe field for the quadrupolar (sextupolar, ..., 20-polar) component is given by a coefficient E_2 (E_3, ..., E_{10}) at entrance, and S_2 (S_3, ..., S_{10}) at exit, such that the extent is $\lambda_E * E_2$ ($\lambda_E * E_3$, ..., $\lambda_E * E_{10}$) at entrance and $\lambda_S * S_2$ ($\lambda_S * S_3$, ..., $\lambda_S * S_{10}$) at exit.

If $\lambda_E = 0$ ($\lambda_S = 0$) the multipole lens is considered to have a sharp edge field at entrance (exit), and then, X_E (X_S) is forced to zero (for the mere purpose of saving computing time). If $E_i = 0$ ($S_i = 0$) ($i = 2, 10$), the entrance (exit) fringe field for the multipole component i is considered as a sharp edge field. In sharp edge field model, the wedge angle vertical first order focusing effect (if $\vec{B}1$ is non zero) is simulated at magnet entrance and exit by a kick $P_2 = P_1 - Z_1 \tan(\epsilon/\rho)$ applied to each particle (P_1, P_2 are the vertical angles upstream and downstream the EFB, Z_1 the vertical particle position at the EFB, ρ the local horizontal bending radius and ϵ the wedge angle experienced by the particle ; ϵ depends on the horizontal angle T).

Overlapping of fringe fields inside the optical element is treated separately for each component, in the way described in *QUADRUPO*.

Any multipole component $\vec{B}i$ can be rotated independently by an angle RXi around the longitudinal X-axis, for the simulation of positioning defects, as well as skewed lenses.

Magnet (mis-)alignement is assured by *KPOS*. *KPOS* also allows some degrees of automatic alignement useful for periodic structures (section 7.6.5).

OCTUPOLE: Octupole magnet (Fig. 33)

The meaning of parameters for *OCTUPOLE* is the same as for *QUADRUPO*. In fringe field regions the magnetic field $\vec{B}(X,Y,Z)$ and its derivatives up to fourth order are derived from the scalar potential approximated to the 8-th order in Y and Z

$$V(X,Y,Z) = \left(G - \frac{G''}{20}(Y^2+Z^2) + \frac{G''''}{960}(Y^2+Z^2)^2\right)(Y^3Z - YZ^3)$$

$$\text{with } G_0 = \frac{B_0}{R_0^3}$$

Outside fringe field regions, or everywhere in sharp edge dodecapole ($\lambda_E = \lambda_S = 0$) , $\vec{B}(X,Y,Z)$ in the magnet is given by

$$B_X = 0$$
$$B_Y = G_0(3Y^2Z - Z^3)$$
$$B_Z = G_0(Y^3 - 3YZ^2)$$

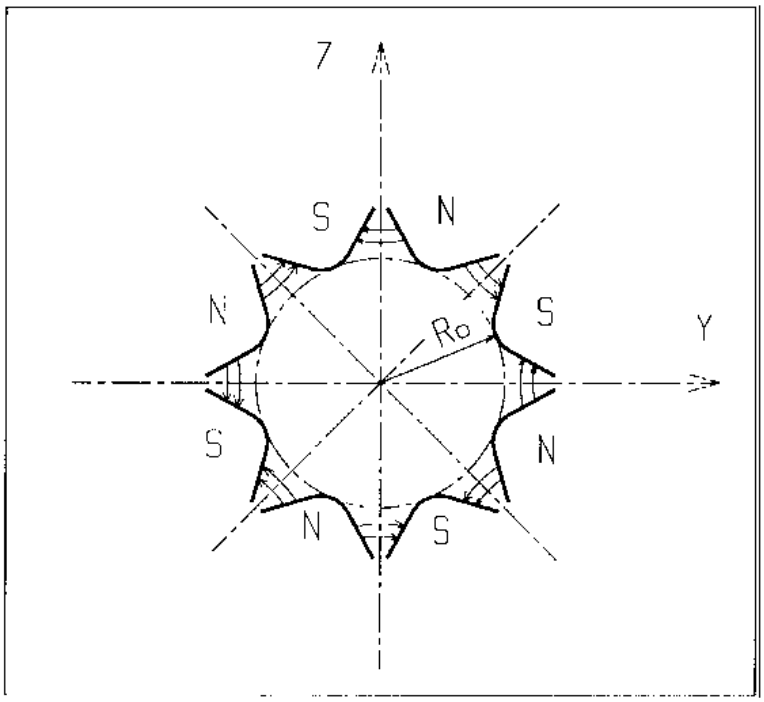


Figure 33: Octupole magnet

POISSON:Read magnetic field data from *POISSON* output

This keyword allows reading a field profile $B(X)$ from *POISSON* output. Let *FNAME* be the name of this output file (normally, *FNAME* = outpoi.lis); the data are read following the *FORTRAN* statements hereunder

```
I = 0
  11    CONTINUE
  I = I + 1
          READ(LUN,101,ERR=10,END=10) K, K, K, R, X(I), R, R, B(I)
  101     FORMAT(I1, I3, I4, E15.6, 2F11.5, 2F12.3)
        GOTO 11
  10    CONTINUE
        ...
```

where X(I) is the longitudinal coordinate, and B(I) is the Z component of the field at a node (I) of the mesh. K's and R's are dummy variables appearing in the *POISSON* output file outpoi.lis but not used here.

From this field profile, a 2-D median plane map is built up, with a rectangular and uniform mesh; mid-plane symmetry is assumed. The field at each node (X_i, Y_j) of the map is $B(X_i)$, independent of Y_j (*i.e.*, the distribution is uniform in the Y direction).

For the rest, *POISSON* works in a way similar to *CARTEMES*.

POLARMES: 2-D polar mesh magnetic field map

Similar to *CARTEMES*, apart from the polar mesh frame: IX is the number of angular nodes, JY the number of radial nodes; $X(I)$ and $Y(J)$ are respectively the angle and radius of a node (these parameters are similar to those entering in the definition of the map in *DIPOLE-M*).

PS170: Simulation of a round shape dipole magnet

PS170 is dedicated to a 'rough' simulation of CERN's *PS170* dipole.

The field B_0 is constant inside the magnet, and zero outside. The pole is a circle of radius R_0, centered on X axis. The output coordinates are generated at the distance XL from the entrance (Fig. 25).

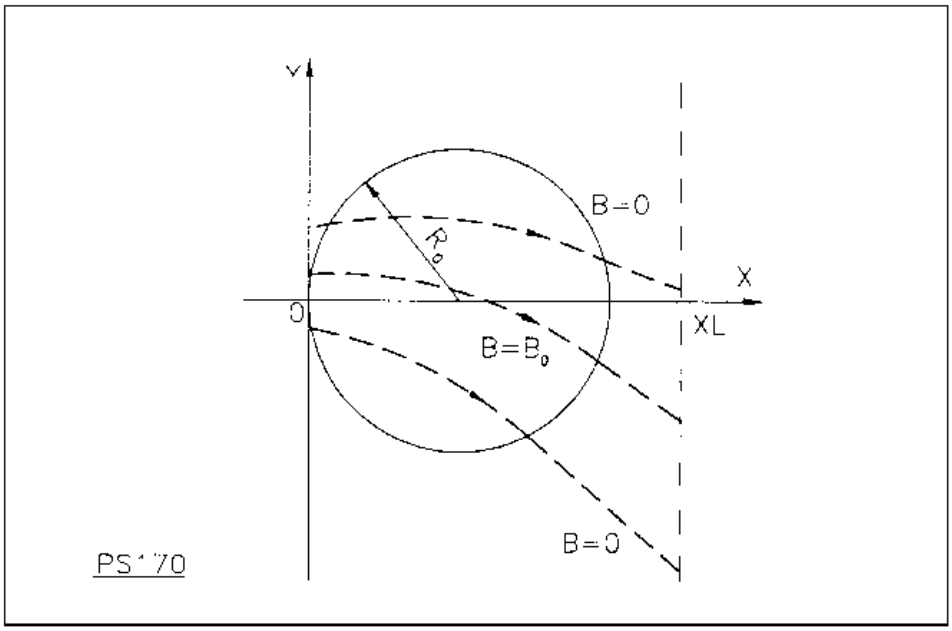


Figure 34: Scheme of the PS170 magnet simulation.

QUADISEX, SEXQUAD: Sharp edge magnetic multipoles

SEXQUAD defines in a simple way a sharp edge field with quadrupolar, sextupolar and octupolar components. *QUADISEX* adds a dipole component. The length of the element is XL. The vertical component $B \equiv B_Z(X, Y, Z = 0)$ of the field and its derivatives in median plane are calculated at each step from the following expressions

$$
\begin{aligned}
B &= B_0 \left(U + \frac{N}{R_0} Y + \frac{B}{R_0^2} Y^2 + \frac{G}{R_0^3} Y^3 \right) \\
\frac{\partial B}{\partial Y} &= B_0 \left(\frac{N}{R_0} + 2\frac{B}{R_0^2} Y + 3\frac{G}{R_0^3} Y^2 \right) \\
\frac{\partial^2 B}{\partial Y^2} &= B_0 \left(2\frac{B}{R_0^2} + 6\frac{G}{R_0^3} Y \right) \\
\frac{\partial^3 B}{\partial Y^3} &= 6 B_0 \frac{G}{R_0^3}
\end{aligned}
$$

and then extrapolated out of the median plane by Taylor expansion in Z (see section 4.2.2).

With option *SEXQUAD*, $U = 0$, while with *QUADISEX*, $U = 1$.

QUADRUPO: Quadrupole magnet (Fig. 35)

The length of the magnet XL is the distance between the effective field boundaries (EFB). The field at the pole tip R_0 is B_0.
The extent of the entrance (exit) fringe field is characterized by $\lambda_E(\lambda_S)$. The distance of ray-tracing on both sides of the EFB's, in the field fall off regions, will be $\pm\ X_E$ at the entrance, and $\pm\ X_S$ at the exit (Fig. 36), by prior and further automatic changes of frame.
In the fringe field regions $[-X_E, X_E]$ and $[-X_S, X_S]$ on both sides of the EFB's, $\vec{B}(X,Y,Z)$ and its derivatives up to fourth order are calculated at each step of the trajectory from the analytical expressions of the three components B_X, B_Y, B_Z obtained by differentiation of the scalar potential (see section 4.2.6) approximated to the 8th order in Y and Z.

$$V(X,Y,Z) = \left(G - \frac{G''}{12}(Y^2+Z^2) + \frac{G''''}{384}(Y^2+Z^2)^2 - \frac{G''''''}{23040}(Y^2+Z^2)^3 \right) YZ$$
$$(G' = dG/dX, \quad G'' = d^2G/dX^2, ...)$$

where G is the gradient on axis [19]:

$$G(s) = \frac{G_0}{1+\exp P(s)} \qquad \text{with} \quad G_0 = \frac{B_0}{R_0}$$

and,

$$P(s) = C_0 + C_1\left(\frac{s}{\lambda}\right) + C_2\left(\frac{s}{\lambda}\right)^2 + C_3\left(\frac{s}{\lambda}\right)^3 + C_4\left(\frac{s}{\lambda}\right)^4 + C_5\left(\frac{s}{\lambda}\right)^5 P(s) = C_0 + C$$

where, s is the distance to the field boundary and λ stands for λ_E or λ_S (normally, $\lambda \simeq 2 * R_0$).
When fringe fields overlap inside the magnet $(XL \leq X_E + X_S)$, the gradient G is expressed as

$$G = G_E + G_S - 1$$

where, G_E is the entrance gradient and G_S is the exit gradient.
If $\lambda_E = 0$ ($\lambda_S = 0$), the field at entrance (exit) is considered as sharp edged, and then $X_E(X_S)$ is forced to zero (for the mere purpose of saving computing time).

Outside of the fringe field regions (or everywhere when $\lambda_E = \lambda_S = 0$) $\vec{B}(X, Y, Z)$ in the magnet is given by

$$B_X = 0$$
$$B_Y = G_0 Z$$
$$B_Z = G_0 Y$$

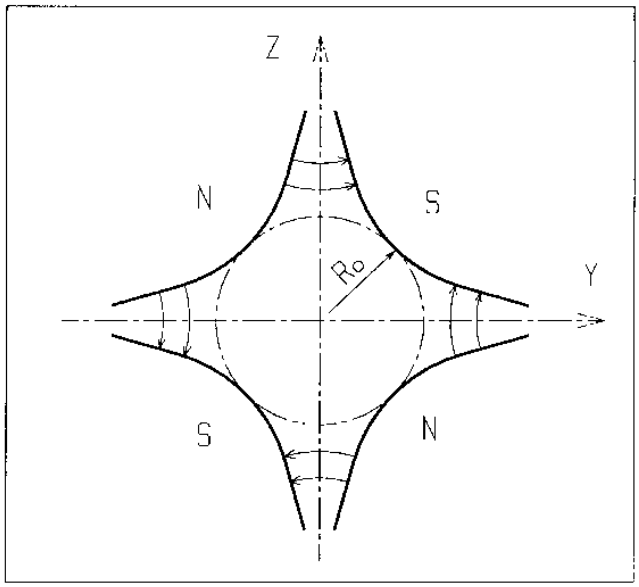


Figure 35: Quadrupole magnet

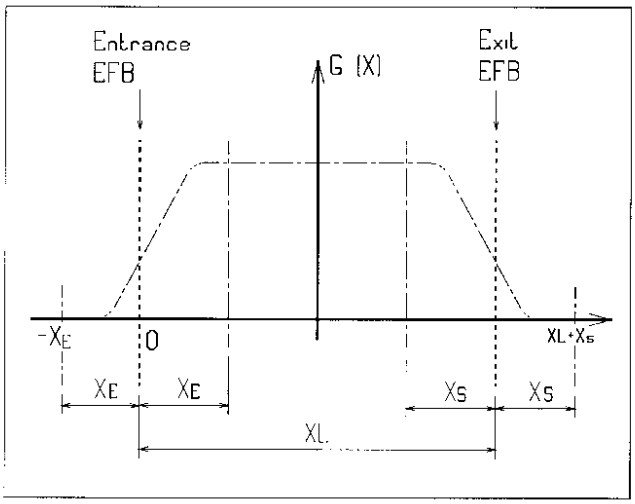


Figure 36: Scheme of the longitudinal field gradient $G(X)$.
(OX) is the longitudinal axis of the reference frame $(0, X, Y, Z)$ of **zgoubi** . The length of the element is XL, but trajectories are ray-traced from $-X_E$ to $XL + X_S$, by means of prior and further automatic changes of frame.

SEPARA: Wien Filter - analytical simulation

SEPARA provides an analytic simulation of an electrostatic separator. Input data are the length L of the element, the electric field E and the magnetic field B. The mass m and charge q of the particles are entered by means of the keyword *PARTICUL*.

The subroutines involved in this simulation solve the following system of three equations with three unknown variables S, Y, Z (while $X \equiv L$), that describe the cycloidal motion of a particle in $\vec{E}$, $\vec{B}$ static fields (Fig. 37).

$$X = -R\cos\left(\frac{\omega S}{\beta c} + \epsilon\right) - \frac{\alpha S}{\omega \beta c} + \frac{C_1}{\omega}$$
$$Y = R\sin\left(\frac{\omega S}{\beta c} + \epsilon\right) - \frac{\alpha}{\omega^2} - \frac{C_2}{\omega} + Y_0$$
$$Z = S\sin(P_0) + Z_0$$

where, S is the path length in the separator, $\alpha = -\frac{Ec^2}{\gamma}$, $\omega = -\frac{Bc^2}{m\gamma}$, $C_1 = \beta\sin(T_0)\cos(P_0)$ and $C_2 = \beta c\cos(T_0)\cos(P_0)$ are initial conditions. c = velocity of light, βc = velocity of the particle, $\gamma = (1-\beta^2)^{-\frac{1}{2}}$ and $\tan\epsilon = (C_2 + \frac{\alpha}{\omega})/C_1$. Y_0, T_0, Z_0, P_0 are the initial coordinates of the particle in the **zgoubi** reference frame. Here βc and γ are assumed constant, which is true as long as the change of momentum due to the electric field remains negligible all along the separator.

The index IA in the input data allows switching to inactive element (thus equivalent to *ESL*), horizontal or vertical separator. Normally, E, B and the value of β_W for wanted particles are related by

$$B(T) = -\frac{E\left(\frac{V}{m}\right)}{\beta_W \cdot c\left(\frac{m}{s}\right)}$$

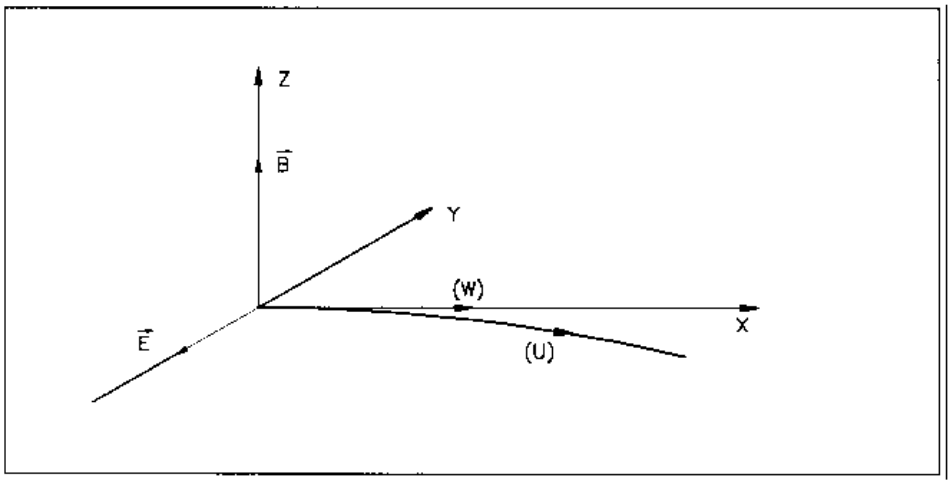


Figure 37: Horizontal separation between a wanted particle, (W), and an unwanted particle, (U). (W) undergoes a linear motion while (U) undergoes a cycloidal motion.

SEXTUPOL: Sextupole magnet (Fig. 38)

The meaning of parameters for *SEXTUPOL* is the same as for *QUADRUPO*.

In fringe field regions the magnetic field $\vec{B}(X,Y,Z)$ and its derivatives up to fourth order are derived from the scalar potential approximated to 7th order in Y and Z

$$V(X,Y,Z) = \left(G - \frac{G''}{16}(Y^2+Z^2) + \frac{G''''}{640}(Y^2+Z^2)^2\right)\left(Y^2Z - \frac{Z^3}{3}\right)$$

$$\text{with } G_0 = \frac{B_0}{R_0^2}$$

Outside fringe field regions, or everywhere in sharp edge sextupole ($\lambda_E = \lambda_S = 0$), $\vec{B}(X,Y,Z)$ in the magnet is given by

$$B_X = 0$$
$$B_Y = 2G_0YZ$$
$$B_Z = G_0(Y^2 - Z^2)$$

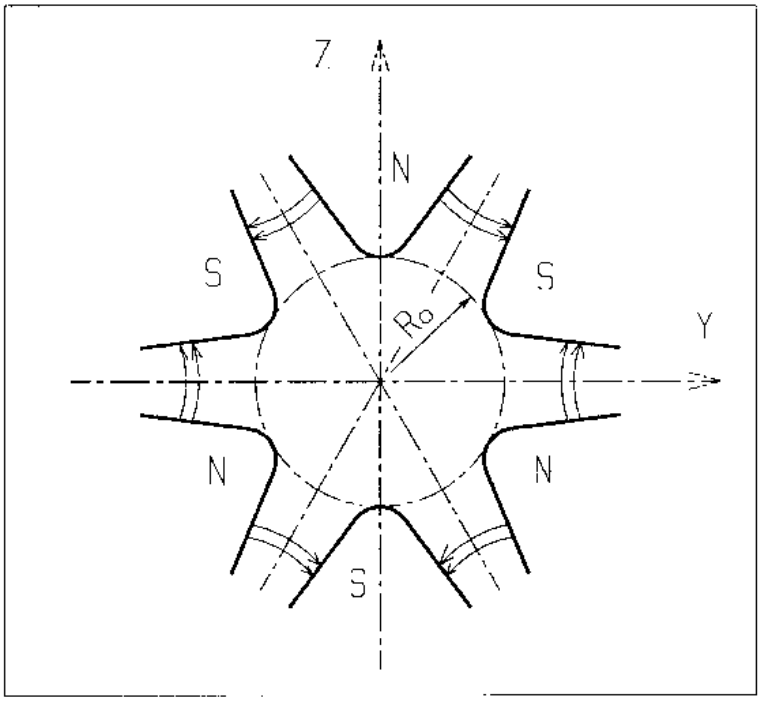


Figure 38: Sextupole magnet

SOLENOID: Solenoid (Fig. 39)

The solenoidal magnet has an effective length XL, a mean radius R_0 and an asymptotic field $B_0 = \mu_0 NI/XL$ (i.e., $\int_{-\infty}^{\infty} B_X(X,r)dX = \mu_0 NI,\ \forall r < R_0$), wherein B_X=longitudinal field component, NI = number of Ampere-Turns, $\mu_0 = 4\pi 10^{-7}$.

The distance of ray-tracing beyond the effective length XL, is X_E at the entrance, and X_S at the exit (Fig. 39).

The field $\vec{B}(X,r)$, $r = (Y^2 + Z^2)^{1/2}$, and its derivatives up to the second order with respect to X, Y or Z are obtained after the method proposed in ref. [27], that involves the three complete elliptic integrals K, E and Π. These are calculated with the algorithm proposed in the same reference. Their derivatives are calculated by means of recursive relations [28].

This analytical model for the solenoidal field allows simulating an extended range of coil geometries (legnth and radius) provided that the coil thickness is small enough compared to the mean radius R_0.

In particular the field on-axis writes (taking $x = r = 0$ as solenoid center)

$$B_X(x, r=0) = \frac{\mu_0 NI}{2XL}\left[\frac{XL/2 - x}{\sqrt{(XL/2-x)^2 + R_0^2}} + \frac{XL/2 + x}{\sqrt{(XL/2+x)^2 + R_0^2}}\right]$$

and yields the magnetic length

$$L_{mag} \equiv \frac{\int_{-\infty}^{\infty} B_X(x, r < R_0)dx}{B_X(x = r = 0)} = XL\sqrt{1 + \frac{4R_0^2}{XL^2}} > XL$$

with in addition

$$B_X(\text{center}) \equiv B_X(x = r = 0) = \frac{\mu_0 NI}{XL\sqrt{1 + \frac{4R_0^2}{XL^2}}}.$$

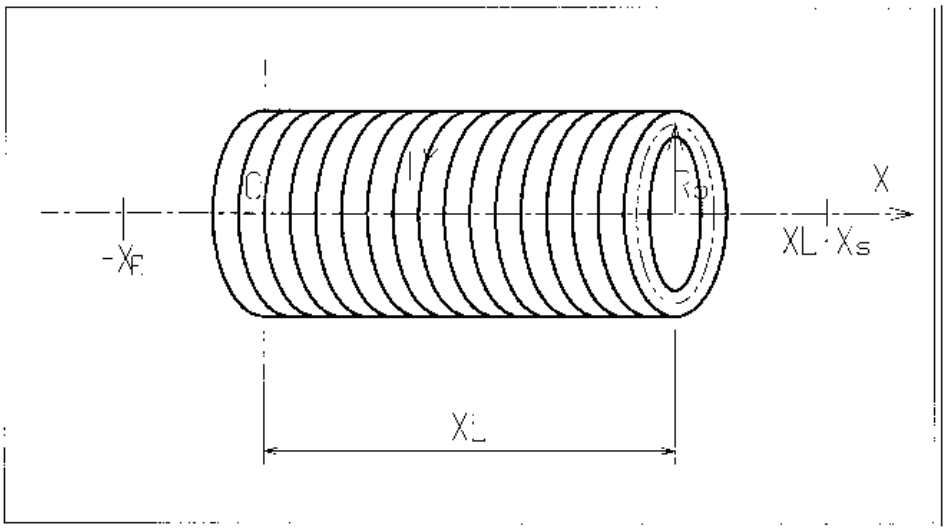


Figure 39: Solenoidal magnet.

TOSCA: 2-D and 3-D Cartesian or cylindrical mesh field map

TOSCA is dedicated to the reading and treatment of 2-D or 3-D Cartesian or cylindrical mesh field maps as delivered by the *TOSCA* magnet computer code standard output.

The total number of field data files to be read is given by the parameter IZ that appears in the data list following the keyword. Each file contains the field components B_X, B_Y, B_Z on an (X, Y) mesh at a given Z coordinate. $IZ = 1$ for 2-D maps, and in this case B_X and B_Y are assumed zero all over the map[8]. For 3-D maps with mid-plane symmetry, $IZ \geq 2$, and thus, the first data file whose name follows in the data list is supposed to contain the median plane field (assuming $Z = 0$ and $B_X = B_Y = 0$), while the next file(s) contain the next maps in increasing Z order. For arbitrary 3-D maps (and in particular, contrary to what precedes without mid-plane symmetry assumption), following *MOD* value, see below, the total number of maps (whose names follow in the data list) is IZ, and map number $[IZ/2] + 1$ is the $Z = 0$ one.

The field map data files should be formatted following the *FORTRAN* reading sequence below.

```
DO  1  K = 1, KZ
  OPEN (UNIT = NL, FILE = FNAME, STATUS = 'OLD' [,FORM='UNFORMATTED'])
  DO  1  J = 1,  JY
    DO  1 I = 1, IX
              IF (BINARY) THEN
                READ(NL) Y(J), Z(K), X(I), BY(J,K,I), BZ(J,K,I), BX(J,K,I)
              ELSE
       READ(NL,100) Y(J), Z(K), X(I), BY(J,K,I), BZ(J,K,I), BX(J,K,I)
 100            FORMAT(1X,6E11.2)
              ENDIF
  1     CONTINUE
```

where, IX (JY, KZ) is the number of longitudinal (transverse horizontal, vertical) nodes of the 3-D uniform mesh. For binary files, FNAME must begin with 'B_' or 'b_', a flag 'BINARY' will thus be set to '.TRUE.'.

A flag *MOD* determines wether Cartesian or Z-axis cylindrical mesh is used. *MOD* can take various values depending also on the map data file formatting. (To be documented - see FORTRAN subroutine FMAPW and its entries FMAPR, FMAPR2.)

The field $\vec{B} = (B_X, B_Y, B_Z)$ is normalized by means of *BNORM* in a similar way as in *CARTEMES*. As well the coordinates X (and Y, Z with 3-D field

[8]Use *MAP2D* in case non-zero B_X, B_Y are to be taken into account in a 2-D map.

maps) is normalized with a *X-[Y-,Z-]NORM* coefficient (usefull to convert to centimeters, the working units in **zgoubi** .

At each step of the trajectory of a particle inside the map, the field and its derivatives are calculated

- in the case of 2-D map, by means of a second or fourth order polynomial interpolation, depending on *IORDRE* (*IORDRE* = 2, 25 or 4), as for *CARTEMES*,

- in the case of 3-D map, by means of a second order polynomial interpolation with a $3 \times 3 \times 3$-point parallelipipedic grid, as described in section 4.3.4.

Entrance and/or exit integration boundaries between which the trajectories are integrated in the field may be defined, in the same way as in *CARTEMES*.

TRAROT: Translation-Rotation of the reference frame

This procedure transports particles into a new frame by translation and rotation. Effect on spin tracking, particle decay and gas-scattering are taken into account (but not on synchrotron radiation).

UNDULATOR: Undulator magnet

UNDULATOR

To be documented

Figure 40: Undulator magnet.

UNIPOT: Unipotential cylindrical electrostatic lens

The lens is cylindrically symmetric about the X-axis.

The length of the first (resp. second, third) electrode is $X1$ (resp. $X2$, $X3$). The distance between the electrodes is D. The potentials are $V1$ and $V2$. The inner radius is R_0 (Fig. 41). The model for the electrostatic potential along the axis is [29]

$$V(x) = \frac{V2 - V1}{2\omega D}\left[\ln \frac{\cosh \frac{\omega\left(x + \frac{X2}{2} + D\right)}{R_0}}{\cosh \frac{\omega\left(x + \frac{X2}{2}\right)}{R_0}} + \ln \frac{\cosh \frac{\omega\left(x - \frac{X2}{2} - D\right)}{R_0}}{\cosh \frac{\omega\left(x - \frac{X2}{2}\right)}{R_0}}\right]$$

($x =$ distance from the center of the central electrode; ω = 1,318; cosh = hyperbolic cosine), from which the field $\vec{E}(X, Y, Z)$ and its derivatives are deduced following the procedure described in section 4.4.2.
Use *PARTICUL* prior to *UNIPOT*, for the definition of particle mass and charge.

The total length of the lens is $X1 + X2 + X3 + 2\,D$; stepwise integration starts at entrance of the first electrode and terminates at exit of the third one.

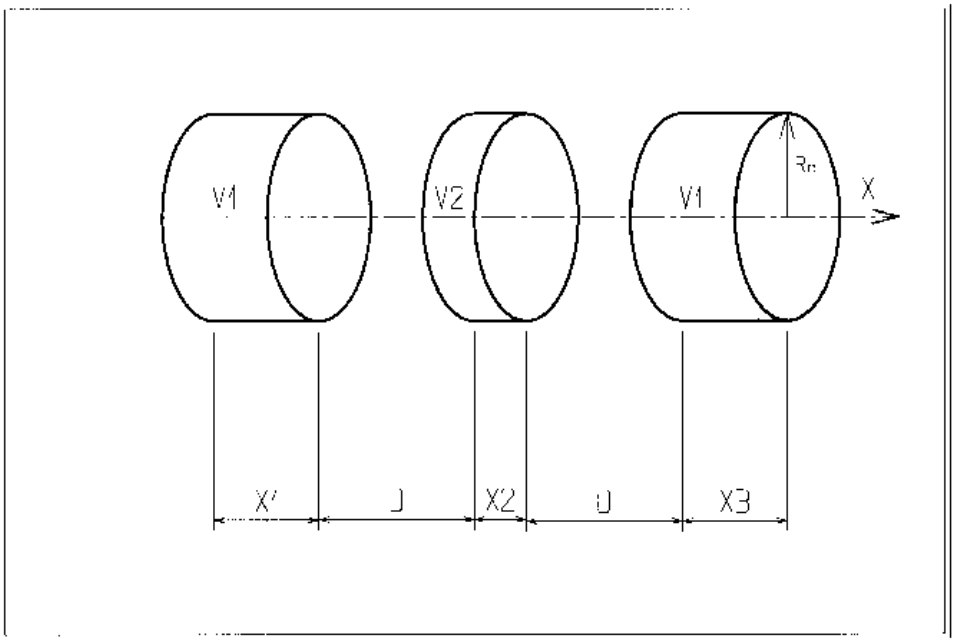


Figure 41: Three-electrode cylindrical unipotential lens.

VENUS: Simulation of a rectangular shape dipole magnet

VENUS is dedicated to a 'rough' simulation of Saturne Laboratory's *VENUS* dipole. The field B_0 is constant inside the magnet, with longitudinal extent XL and transverse extent $\pm YL$; outside these limits, $B_0 = 0$ (Fig. 42).

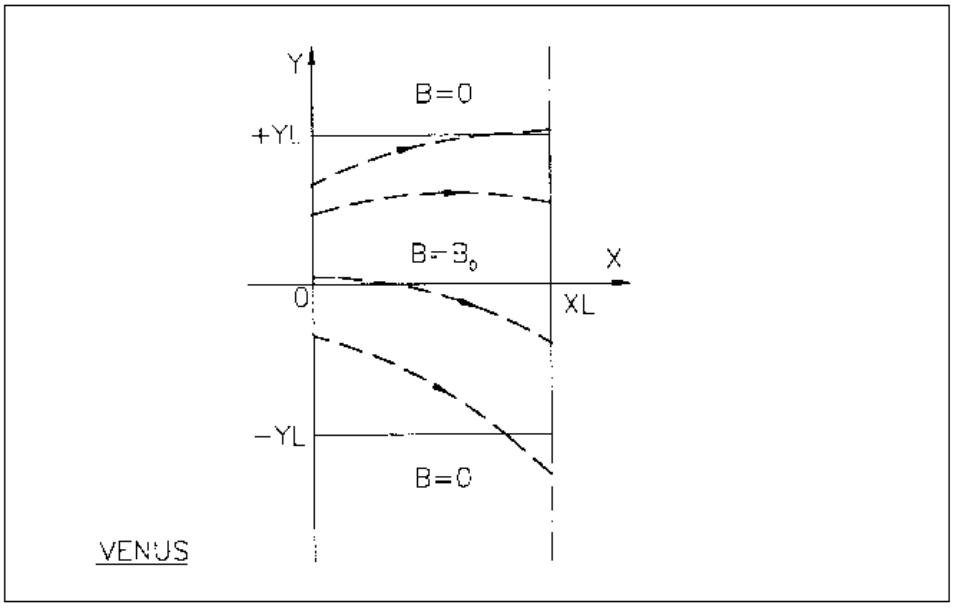


Figure 42: Scheme of *VENUS* rectangular dipole.

WIENFILT: Wien filter

WIENFILT simulates a Wien Filter, with transverse and orthogonal electric and magnetic fields $\vec{E}_Y$, $\vec{B}_Z$ or $\vec{E}_Z$, $\vec{B}_Y$ (Fig. 37). It must be preceded by *PARTICUL* for the definition of particle mass and charge.

The length XL of the element is the distance between its entrance and exit EFB's. The electric and magnetic field intensities E_0 and B_0 in the central, uniform field region, normally satisfy the relation

$$B_0 = -\frac{E_0}{\beta_W c}$$

for the selection of " wanted" particles of velocity $\beta_W c$. Ray-tracing in field fall-off regions extends over a distance X_E (X_S) beyond the entrance (exit) EFB by means of prior and further automatic changes of frame. Four sets of coefficients λ, $C_0 - C_5$ allow the description of the entrance and exit fringe fields outside the uniform field region, following the model [19]

$$F = \frac{1}{1 + \exp(P(s))}$$

where $P(s)$ is of the term

$$P(s) = C_0 + C_1\left(\frac{s}{\lambda}\right) + C_2\left(\frac{s}{\lambda}\right)^2 + C_3\left(\frac{s}{\lambda}\right)^3 + C_4\left(\frac{s}{\lambda}\right)^4 + C_5\left(\frac{s}{\lambda}\right)^5$$

and s is the distance to the EFB. When fringe fields overlap inside the element (*i.e.* $XL \leq X_E + X_S$), the field fall-off is expressed as

$$F = F_E + F_S - 1$$

where $F_E(F_S)$ is the value of the coefficient respective to the entrance (exit) EFB.

If $\lambda_E = 0$ ($\lambda_S = 0$) for either the electric or magnetic component, then both are considered as sharp edge fields and $X_E(X_S)$ is forced to zero (for the purpose of saving computing time). In this case, the magnetic wedge angle vertical first order focusing effect is simulated at entrance and exit by a kick $P_2 = P_1 - Z_1 \tan(\epsilon/\rho)$ applied to each particle (P_1, P_2 are the vertical angles upstream and downstream the EFB, Z_1 the vertical particle position at the EFB, ρ the local horizontal bending radius and ϵ the wedge angle experienced by the particle

; ϵ depends on the horizontal angle T). This is not done for the electric field, however it is advised not to use a sharp edge electric dipole model since this entails non symplectic mapping, and in particular precludes focusing effects of the non zero longitudinal electric field component.

YMY: Reverse signs of Y and Z reference axes

YMY performs a 180° rotation of particle coordinates with respect to the X-axis, as shown in Fig. 43. This is done by means of a change of sign of Y and Z axes, and therefore coordinates, as follows

$$Y2 = -Y1, \quad T2 = -T1, \quad Z2 = -Z1 \quad \text{and} \quad P2 = -P1$$

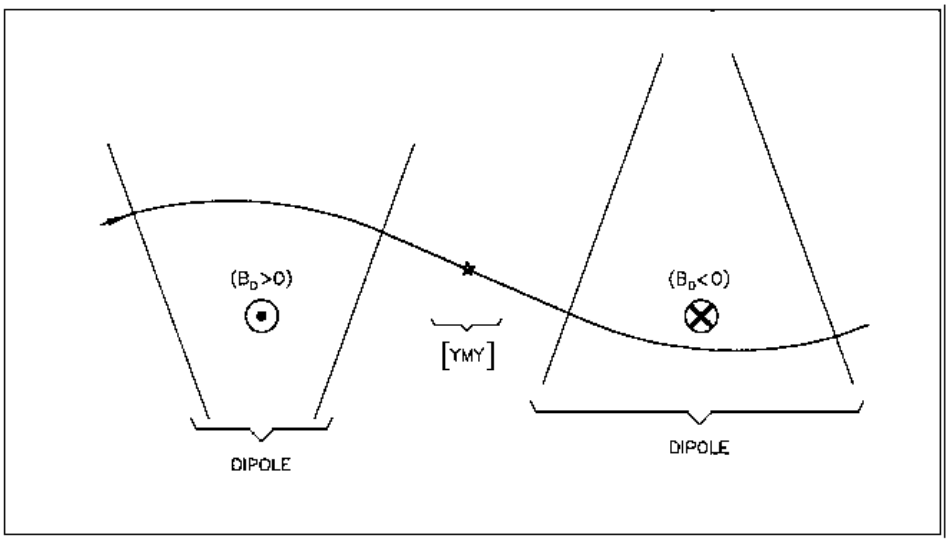


Figure 43: The use of YMY in a sequence of two identical dipoles o

7.5 Output Procedures

These procedures are dedicated to the printing of particle coordinates, histograms, spin coordinates, etc. They may be called for at any spot in the data pile.

FAISCEAU, FAISCNL, FAISTORE: Print/Store particle coordinates

FAISCEAU can be introduced anywhere in a structure. It produces a print of initial and actual coordinates of the particles at the location where it stands, together with their tagging indices and letters, following the same format as for *FAISCNL* (except for *SORT(I)* which is not printed) .

FAISCNL has a similar effect, except that the information is stored in a dedicated file *FNAME* (standard name is *FNAME* = 'zgoubi.fai' for post-processing with **zpop**). This file may further on be read by means of *OBJET*, option *KOBJ* = 3, or used for other purposes such as graphics (see Part D of the Guide). The data written to that file are formatted and ordered according to the *FORTRAN* sequence below

```
   OPEN (UNIT = NL, FILE = FNAME, STATUS = 'NEW')
   DO  1  I=1, IMAX
     WRITE(NL,110) LET(I),IEX(I),(FO(J,I),J=1,7),(F(J,I),J=1,7),KineticE,
  >  I(I),IREP(I),SORT(I),Mass,Charge,G-Factor,com-Life-time,unused,RET(I), DPR(I),
  >                                    BORO,IPASS,KLEY,LBL1,LBL2,NOEL
110     FORMAT(1X,A1,I2,1P,7E16.8,
    2         /,3E24.16,
    3         /,4E24.16,E16.8,
    4         /,2I6,8E16.8,
    5         /,E16.8,   I6,  A8,  2A10, I5)
```

The meaning of main data is the following (see the keyword *OBJET*)

$LET(I)$	: one-character string, for tagging particle number I
$IEX, I, IREP(I)$	: flag, particle number, index
$FO(1-6, I)$	: coordinates D, Y, T, Z, P and path length at the origin
$F(1-6, I)$	: idem, at the current position
$SORT(I)$	: path length at which the particle has possibly been stopp (see *CHAMBR* or *COLLIMA*)
$RET(I), DPR(I)$	: synchrotron phase space coordinates; *RET* =phase (radi *DPR* = momentum dispersion (MeV/c) (see *CAVITE*)
IPASS	: turn number (see *REBELOTE*)
etc.	:

FAISTORE has an effect similar to *FAISCNL*, with two more features. On the first data line, *FNAME* may be followed by a series of up to 10 *LABEL*'s proper to the elements of the data file at the exit of which the print should occur; if there is no label, the print occurs by default at the location of *FAISTORE*; if there are labels the print occurs right downstream of all optical elements

wearing those labels (and no longer at the *FAISTORE* location). The next data line gives a parameter IP: printing will occur every IP other pass, if using *REBELOTE* with *NPASS* $\geq IP - 1$. For instance the data list

FAISTORE
zgoubi.fai *HPCKUP* *VPCKUP*
12

will result in output prints into zgoubi.fai, every 12 other pass, each time elements of the zgoubi.dat data list labeled either *HPCKUP* or *VPCKUP* are encountered.

Note

Binary storage can be obtained from *FAISCNL* and *FAISTORE*. This for the sake of compactness and access speed, for instance in case voluminous amounts of data would have to be manipulated.

This is achieved by giving the storage file a name of the form *b_FNAME* or *B_FNAME* (e.g., 'b_zgoubi.fai'). The FORTRAN *WRITE* list is the same as in the *FORMATTED* case above.

This is compatible with the *READ* statements in **zpop** that will recognize binary storage from that very radical 'b_' or 'B_'.

FOCALE, IMAGE[S]: Particle coordinates and beam size; localization and size of horizontal waist

FOCALE calculates the dimensions of the beam and its mean transverse position, at a longitudinal distance XL from the position corresponding to the keyword *FOCALE*.

IMAGE computes the location and size of the closest horizontal waist.

IMAGES has the same effect as *IMAGE*, but, in addition, for a non-monochromatic beam it calculates as many waists as there are distinct momenta in the beam, provided that the object has been defined with a classification of momenta (see *OBJET*, *KOBJ* = 1, 2 for instance).

Optionally, for each of these three procedures, **zgoubi** can list a trace of the coordinates in the X, Y and in the Y, Z planes.

The following quantities are calculated for the N particles of the beam (*IMAGE, FOCALE*) or of each group of momenta (*IMAGES*)

- Longitudinal position:

$$\textit{FOCALE}: \quad X = XL$$

$$\textit{IMAGE[S]}: \quad X = -\frac{\sum_{i=1}^{N} Y_i * tgT_i - \left(\sum_{i=1}^{N} Y_i * \sum_{i=1}^{N} tgT_i\right) /N}{\sum_{i=1}^{N} tg^2T_i - \left(\sum_{i=1}^{N} tgT_i\right)^2 /N}$$

$$Y = Y_1 + X * \mathrm{tg}T_1$$

  where Y_1 and T_1 are the coordinates of the first particle of the beam (*IMAGE, FOCALE*) or the first particle of each group of momenta (*IMAGES*).

- Transverse position of the center of mass of the waist (*IMAGE[S]*) or of the beam (*FOCALE*), with respect to the reference trajectory

$$YM = \frac{1}{N}\sum_{i=1}^{N}(Y_i + X\mathrm{tg}T_i) - Y = \frac{1}{N}\sum_{i=1}^{N} Y\, M_i$$

- FWHM of the image (*IMAGE[S]*) or of the beam (*FOCALE*), and total width, respectively, W and WT

$$W = 2.35 \left(\frac{1}{N} \sum_{i=1}^{N} Y\ M_i^2 - Y\ M^2 \right)^{\frac{1}{2}}$$

$$WT = \max(YM_i) - \min(YM_i)$$

FOCALEZ, IMAGE[S]Z: Particle coordinates and beam size; localization and size of vertical waist

Similar to *FOCALE* and *IMAGE[S]*, but the calculations are performed with respect to the vertical coordinates Z_i and P_i, in place of Y_i and T_i.

HISTO: 1-D histogram

Any of the coordinates used in **zgoubi** may be histogrammed, namely initial Y_0, T_0, Z_0, P_0, S_0, D_0 or actual Y, T, Z, P, S, D particle coordinates (S = path length ; D may change in decay process simulation with *MCDESINT*, or when ray-tracing in $\vec{E}$ fields), and also spin coordinates and modulus S_X, S_Y, S_Z and $\left\|\vec{S}\right\|$.

HISTO can be used in conjunction with *MCDESINT*, for statistics on the decay process, by means of *TYP*. *TYP* is a one-character variable. If it is set equal to 'S', only secondary particles will be histogrammed. If it is set equal to 'P', then only primary particles will be histogrammed. For no discrimination between S-econdary and P-rimary particles, *TYP* = 'Q' must be used.

The dimensions of the histogram (number of lines and columns) may be modified. It can be normalized with *NORM* = 1, to avoid saturation.

Histograms are indexed with the parameter NH. This allows making independent histograms of the same coordinate at several spots in a structure. This is also useful when piling up problems in an input data file (see also *RESET*). NH is in the range 1-5.

If *REBELOTE* is used, the statistics on the $1+$*NPASS* runs in the structure will add up.

IMAGE[S][Z]: Localization and size of vertical waists

See FOCALE[Z].

MATRIX: Calculation of transfer coefficients, periodic parameters

MATRIX causes the calculation of the transfer coefficients of the structure, at the spot where it is introduced in the structure, or at the closest horizontal focus. In this last case the position of the focus is calculated automatically in the same way as the position of the waist in *IMAGE*. Depending on option *IFOC*, *MATRIX* also delivers the Twiss functions, tune numbers, chromaticities and other perturbation parameters in the hypothesis of a periodic structure.

Depending on the value of option *IORD*, different procedures follow

- If *IORD* = 0, *MATRIX* is inhibited (equivalent to *FAISCEAU*, whatever *IFOC*).
- If *IORD* = 1, the first order transfer matrix $[R_{ij}]$ is calculated, from a third order expansion of the coordinates. For instance

  $$Y^+ = \left(\frac{Y}{T_0}\right) T_0 + \left(\frac{Y}{T_0^2}\right) T_0^2 + \left(\frac{Y}{T_0^3}\right) T_0^3, \quad Y^- = -\left(\frac{Y}{T_0}\right) T_0 + \left(\frac{Y}{T_0^2}\right) T_0^2 - \left($$

  will yield, neglecting third order terms,

  $$R_{12} = \left(\frac{Y}{T_0}\right) = \frac{Y^+ - Y^-}{2T_0}$$

- If *IORD* = 2, fifth order Taylor expansions are used for the calculation of the first order transfer matrix $[R_{ij}]$ and of the second order matrix $[T_{ijk}]$. Other higher order coefficients are also calculated.

An automatic generation of an appropriate object for the use of *MATRIX* can be obtained by means
- if *IORD* = 1, of the procedure *OBJET*(KOBJ = 5[.I, I=1,9]) (pages 93, **??**), that generates sets of up to 9*11 trajectories. In this case, up to nine matrices may be calculated, each one wrt. to the reference trajectory of concern as indicated using I in KOBJ = 5[.I, I=1,9] ;
- if *IORD* = 2, of the procedure *OBJET*(KOBJ = 6) that generates a set of 61 trajectories.

The next option, *IFOC*, acts as follows

- If *IFOC* = 0, the transfer coefficients are calculated at the location of *MATRIX*, and with respect to the reference trajectory. For instance, Y^+ and

T^+ above are defined for particle number i as $Y^+ = Y^+(i) - Y(Ref)$, and $T^+ = T^+(i) - T(\text{ref.})$.

- If *IFOC* = 1, the transfer coefficients are calculated at the horizontal focus closest to *MATRIX* (determined automatically), while the reference direction is that of the reference particle. For instance, Y^+ is defined for particle number i as $Y^+ = Y^+(i) - Y_{\text{focus}}$, while T^+ is defined as $T^+ = T^+(i) - T(\text{ref.})$).
- If *IFOC* = 2, no change of reference frame is performed: the coordinates refer to the current frame. Namely, $Y^+ = Y^+(i)$, $T^+ = T^+(i)$, etc.

Periodic structures

- If *IFOC* = 10 + *NPeriod*, *MATRIX* calculates periodic parameters characteristic of the structure such as Twiss functions and tune numbers, assuming that it is *NPeriod*-periodic; no change of reference is performed for these calculations. If *IFOC* = 2 additional periodic parameters are computed such as chromaticities, beta-function momentum dependence, etc.

  These quantities are derived from the first order perturbed and unperturbed transfer matrices as obtained in the way described above, and by identification with the Twiss form $[R_{ij}] = I\cos(\mu) + J\sin(\mu)$.

PICKUPS: Beam centroid path; closed orbit

PICKUPS computes the beam centroid path, from average value of particle coordinates as observed at *LABEL'*ed keywords.

In conjunction with *REBELOTE*, this procedure computes by the same method the closed orbit in the periodic structure.

The *LABEL* list of concern follows the keyword *PICKUPS*.

PLOTDATA: Intermediate output for the PLOTDATA graphic software [30]

To be documented

SPNPRNL, SPNPRNLA, SPNPRT: Print/Store spin coordinates

SPNPRNL has the same effect as *SPNPRT* (see below), except that the information is stored in a dedicated file *FNAME* (standard is *FNAME* = 'zgoubi.spn' for post-processing with **zpop**). The data are formatted and ordered according to the *FORTRAN* sequence below

```
OPEN (UNIT = NL, FILE = FNAME, STATUS = 'NEW')
DO  1  I=1, IMAX
    WRITE (NL,100) LET(I), IEX(I), (SI(J,I)J=1,4), (SF(J,I),J=1,4), GAMMA, I
 100   FORMAT(1X, A1, I2, 1P, 8E15.7, /, E15.7, 2I3, I6)
  1     CONTINUE
```

The meaning of these parameters is the following

LET(I),IEX(I) : tagging character and flag (see *OBJET*)

SI(1-4,I) : spin components SX, SY, SZ and modulus, at the origin

SF(1-4,I) : idem, at the current position

GAMMA : Lorentz relativistic factor

I : particle number

IMAX : total number of particles ray-traced (see *OBJET*)

IPASS : turn number (see *REBELOTE*)

SPNPRNLA has an effect similar to *SPNPRNL*, with one more feature. The line next to *FNAME* gives a parameter IP printing will occur every IP other pass, when using *REBELOTE* with *NPASS* $\geq IP - 1$.

SPNPRT can be introduced anywhere in a structure. It produces a listing (into zgoubi.res) of the initial and actual coordinates and modulus of the spin of the *IMAX* particles, at the location where it stands, together with their Lorentz factor γ, following the format detailed above. The mean values of the spin components are also printed.

SRPRNT: Print SR loss statistics

SRPRNT may be introduced anywhere in a structure. It produces a listing (into zgoubi.res) of current state of statistics on several parameters related to SR loss presumably activated beforehand with keyword *SRLOSS*.

TWISS: Calculation of optical parameters ; periodic parameters

TWISS causes the calculation of transfer coefficients and various other parameters, in particular periodical quantities such as tunes, chromaticies, etc.
If *KTWISS* = 1, the object necessary for these calculations can be generated automatically by means of *OBJET* with option *KOBJ* = 5. When using *KTWISS* = 2, the object can be generated automatically with *OBJET* and *KOBJ* = 6.

7.6 Complementary Features

7.6.1 Backward Ray-tracing

For the purpose of parameterization for instance, it may be interesting to ray-trace backward from the image toward the object. This can be performed by first reversing the position of optical elements in the structure, and then reversing the integration step sign in all the optical elements.

An illustration of this feature is given in the following Figure 44.

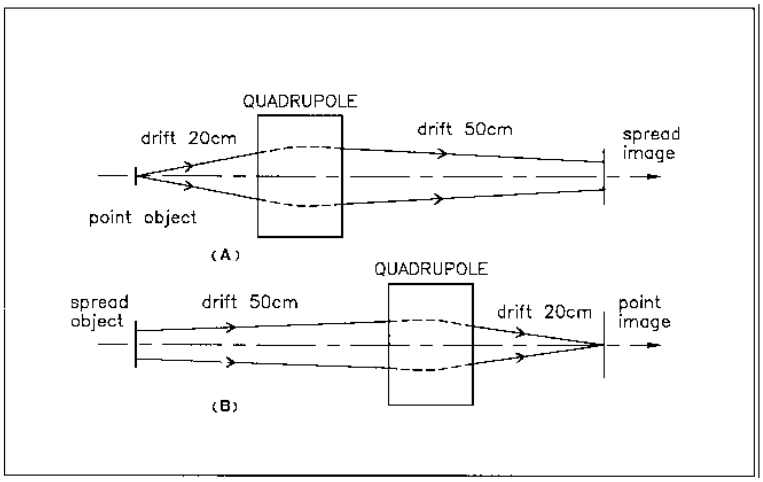


Figure 44: A. Regular forward ray-tracing, from object to image.
B. Same structure, with backward ray-tracing from image to object: negative integration step XPAS is used in the quadrupole.

7.6.2 Checking Fields and Trajectories inside Optical Elements

In all optical elements, an option index IL is available. It is normally set to 0 and in this case has no effect.
$IL = 1$ causes a print in zgoubi.res of particle coordinates and field along trajectories in the optical element. In the meantime, a calculation and summation of the values of $\vec{\nabla} \cdot \vec{B}$, $\vec{\nabla} \times \vec{B}$ and $\nabla^2 \vec{B}$ (same for $\vec{E}$) at all integration steps is performed, which allows a check of the behavior of $\vec{B}$ (or $\vec{E}$) in field maps (all these derivatives should normally be zero).
$IL = 2$ causes a print of particle coordinates and other informations in zgoubi.plt at each integration step ; this information can further be processed with **zpop**[9]. In order to limit the volume of that storage file (when dealing with small step size, large number of particles, etc.) it is possible to print out every other 10^n integration step by taking $IL = 2 \times 10^n$ (for instance, $IL = 200$ would cause output into zgoubi.plt every 100 other step).

When dealing with maps (e.g., *CARTEMES*, *ELREVOL*), another option index IC is available. It is normally set to 0 and in this case has no effect.
$IC = 1$ causes a print of the field map in zgoubi.res.
$IC = 2$ will cause a print of field maps in zgoubi.map which can further be processed with **zpop**.

[9] See Part D of the Guide.

7.6.3 Labeling keywords

Keywords in **zgoubi** data file zgoubi.dat can be *LABEL*'ed, for the purpose of the execution of such procedures as *PICKUPS*, *FAISCNL*, *FAISTORE* , *SCALING*, and also for the purpose of particle coordinate storage into zgoubi.plt (see Sections 7.6.2 and **??**).

Each keyword accepts two *LABEL*'s, of which the first one is used for the above mentioned purposes. The keyword and related *LABEL*['s] should fit within a 80-character long string on a single line.

7.6.4 Multiturn tracking in circular machines

Multiturn tracking in circular machines can be performed by means of the keyword *REBELOTE*, put at the end of the optical structure with its argument *NPASS*$+1$ being the number of turns to be performed. In order that the *IMAX* particles of the beam start a new turn with the coordinates they have reached at the end of the previous one, the option $K = 99$ has to be specified in *REBELOTE*.

Synchrotron acceleration can be simulated, following the procedure below

- *CAVITE* appears at the end of the structure (before *REBELOTE*), with option *IOPT*$= 1$
- the R.F. frequency of the cavity is given a timing law by means of *SCALING*, family *CAVITE*
- the magnets are given the same timing law $B\rho(T)$, (where $T = 1$ to *NPASS*$+1$ is the turn number) by means of *SCALING*.

Eventually some families of magnets may be given a law which does not follow $B\rho(T)$, for the simulation of special processes (e.g. fast crossing of spin resonances with independent families of quadrupoles).

7.6.5 Positioning, (mis-)alignement, of optical elements and field maps

The last record in most optical elements and field maps is the positioning flag *KPOS*, followed by the parameterss *XCE, YCE* for translation and *ALE* for rotation. The positioning works in two different ways, depending whether they are defined in Cartesian (X, Y, Z) coordinates (e.g., *QUADRUPO, TOSCA*), or polar (R, θ, Z) coordinates (*DIPOLE*).

Cartesian Coordinates:

If *KPOS* $= 1$, the optical element is moved (shifted by *XCE, YCE* and Z-rotated by *ALE*) with respect to the incoming reference frame. Trajectory coordinates after traversal of the element refer the element frame.
If *KPOS* $= 2$, the shifts *XCE* and *YCE*, and the tilt angle *ALE* are taken into account, for mis-aligning the element with respect to the incoming reference, as shown in Fig. 45. The effect is equivalent to a *CHANGREF(XCE,YCE,ALE)* upstream of the optical element, followed by *CHANGREF(XCS,YCS,ALS=-ALE)* downstream of it, with computed *XCS, YCS* values as schemed in Fig. 45.
KPOS $= 3$ option is available for some magnets (e.g., *BEND*, *MULTIPOL*); it is effective only if a non zero dipole component $B1$ is present, or if *ALE* is non-zero. It positions automatically the device in the following way, convenient for periodic structures (Fig 46).

Both incoming and outgoing refernce frames are tilted w.r.t. the magnet

- either, by an angle *ALE* if ALE$\neq$0,
- or, if ALE=0 by half the deviation $\theta/2$ (such that $L = 2\frac{BORO}{B1}\sin(\theta/2)$

wherein L = geometrical length, *BORO*= reference rigidity as defined in *OBJET*).
Next, the optical element is shifted by *Y-shift=YCE* (*XCE* is not used) in a direction orthogonal to the new magnet axis (i.e., at an angle $ALE + \pi/2$ wrt. the incoming reference frame X-axis).

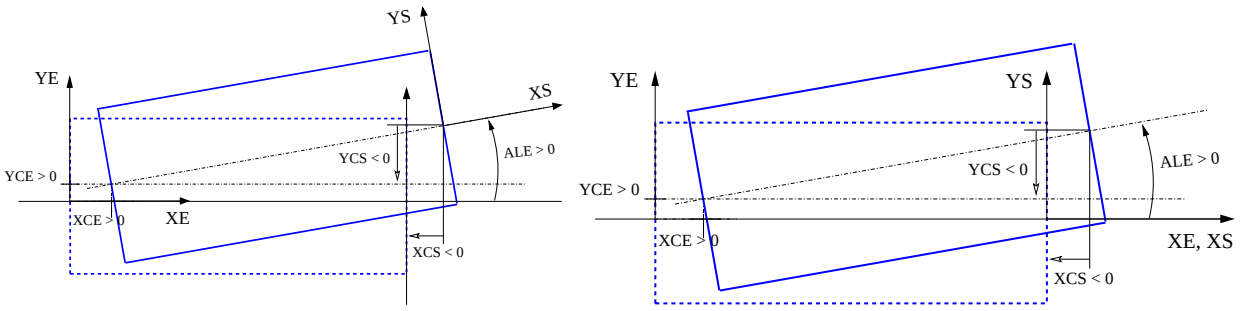


Figure 45: Left: moving an optical element using *KPOS*$= 1$. Right: Mis-aligning an optical element using *KPOS*$= 2$. (X_E, Y_E) and (X_S, Y_S) are respectively the incoming and outgoing reference frames.

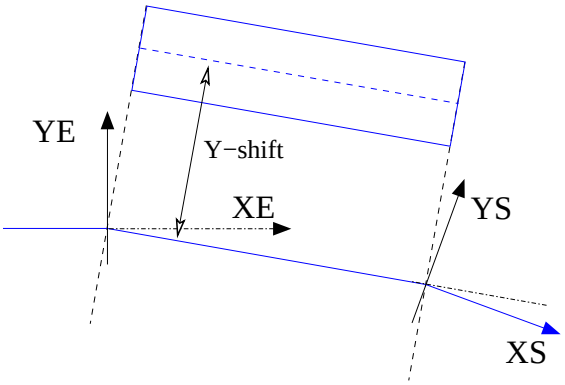


Figure 46: Half-deviation alignement of a Cartesian coordinate bending element, using *KPOS*$= 3$. (X_E, Y_E) and (X_S, Y_S) are respectively the incoming and outgoing reference frames.

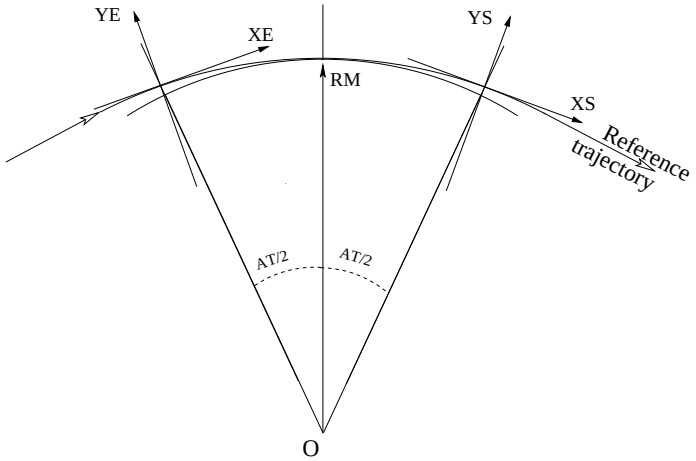


Figure 47: Positioning of a polar field map, using *KPOS*$= 1$.

Polar Coordinates

If *KPOS* $= 1$, the element is positioned automatically in such a way that a particle entering with zero initial coordinates and $1+DP = B\rho/BORO$ relative momentum will reach position (RM, $\frac{AT}{2}$) in the element with $T = 0$ angle with respect to the moving frame in the polar coordinates system of the element (Fig. 47; see *DIPOLE-M* and *POLARMES*).
If *KPOS* $= 2$, the map is positioned in such a way that the incoming reference frame is presented at radius RE with angle TE. The exit reference frame of **zgoubi** is positioned in a similar way with respect to the map, by means of the two parameters RS (radius) and TS (angle) (see Fig. 13A.).

7.6.6 Coded integration step

In several optical elements (e.g., all multipoles, *BEND*) the integration step (in general noted *XPAS*) can be coded under the form *XPAS* = b.fffE10 in order to allow two different step sizes in the uniform part of the field (the optical element body) and in the field fall-off regions. b is an arbitrary integer and fff is a 3-digit integer; they give the number of steps respectively in the body and fringe field regions. For instance 120.012E10 requests 120 steps in the body and 12 in the fringe field regions. The maximum allowed value for fff is 999 steps.

7.6.7 Ray-tracing of an arbitrarily large number of particles

Monte Carlo multiparticle simulations involving an arbitrary number of particles can be performed by means of *REBELOTE*, put at the end of the optical structure, with its argument *NPASS* being the number of passes through *REBELOTE*, and (*NPASS*+1) * *IMAX* the number of particles to be ray-traced. In order that new initial conditions (D, Y, T, Z, P, X) be generated at each pass, $K = 0$ has to be specified in *REBELOTE*.
Statistics on coordinates, spins, and other histograms can be performed by means of such procedures as *HISTO, SPNTRK*, etc. that stack the information from pass to pass.

7.6.8 Stopped particles: the IEX flag

As described in *OBJET*, each particle $I = 1$, IMAX is attached a value $IEX(I)$ of the IEX flag. Normally, $IEX(I) = 1$. Under certain circumstances, IEX may take negative values, as follows

-1 : the trajectory happened to wander outside the limits of a field map

-2 : too many integration steps in an optical element

-3 : deviation happened to exceed $\frac{\pi}{2}$ in an optical element

-4 : stopped by walls (procedures *CHAMBR, COLLIMA*)

-5 : too many iterations in subroutine *DEPLA*

-6 : energy loss exceeds particle energy

-7 : field discontinuities larger than 50% wthin a field map

-8 : reached field limit in an optical element

Only in the case $IEX = -1$ will the integration not be stopped since in this case the field outside the map is extrapolated from the map data, and the particle may possibly get back into the map (see section 4.3.2 on page 59). In all other cases the particle of concern will be stopped.

7.6.9 Negative rigidity

zgoubi can handle negative rigidities $B\rho = p/q$. This is equivalent to considering either particles of negative charges ($q < 0$), or counter going particles ($p < 0$), or virtually reversed fields (w.r.t. the field sign that shows in the optical element data list).
Negative rigidities may be specified in terms of $BORO < 0$ or $D = B\rho/BORO < 0$ when defining the initial coordinates with *OBJET* and *MCOBJET*.

References

[1] F. Méot et S. Valéro, *Manuel d'utilisation de Zgoubi*, report IRF/LNS/88-13, CEA Saclay, 1988.

[2] F. Méot and S. Valéro, *Zgoubi users' guide*, report DSM/LNS/GT/90-05, CEA Saclay (1990) and TRIUMF report TRI/CD/90-02 (1990).

[3] F. Méot and S. Valéro, *Zgoubi users' guide*, report DSM/LNS/GT/93-12, CEA Saclay (1993).

[4] F. Méot, *The electrification of Zgoubi*, Saturne report DSM/LNS/GT/93-09, CEA Saclay (1993) ; F. Méot, *Generalization of the Zgoubi method for ray-tracing to include electric fields*, NIM A 340 (1994) 594-604.

[5] D. Carvounas, *Suivi numérique de particules chargées dans un solénoïde*, report de stage, CEA/LNS/GT-1991.

[6] F. Méot, *Raytracing in 3-D field maps with Zgoubi*, report DSM/LNS/GT/90-01, CEA Saclay, 1990.

[7] G. Leleux, *Compléments sur la physique des accélérateurs*, cours de DEA Univ. Paris-VI, report IRF/LNS/86-101, CEA Saclay, March 1986.

[8] F. Méot, *A numerical method for combined spin tracking and raytracing of charged particles*, NIM **A313** (1992) 492, and proc. EPAC (1992) p.747.

[9] V. Bargmann, L. Michel, V.L. Telegdi, Precession of the polarization of particles moving in a homogeneous electromagnetic field, Phys. Rev. Lett. 2 (1959) 435.

[10] F. Méot and J. Payet, *Numerical tools for the simulation of synchrotron radiation loss and induced dynamical effects in high energy transport lines*, Report DSM/DAPNIA/SEA-00-01, CEA Saclay (2000).

[11] F. Méot, *Synchrotron radiation interferences at the LEP miniwiggler*, Report CERN SL/94-22 (AP), 1994.

[12] L. Ponce, R. Jung, F. Méot, LHC proton beam diagnostics using synchrotron radiation, Yellow Report CERN-2004-007.

[13] J.D. Jackson, *Classical electrodynamics*, 2nd Ed., J. Wiley and Sons, New York, 1975.

[14] F. Méot, A theory of low frequency far-field synchrotron radiation, Particle Accelerators Vol 62, pp. 215-239 (1999).

[15] B. Mayer, personal communication, CEA Saclay, 1990.

[16] Installed by J. S. Berg, BNL (2007). Cf. Detection and remediation of stagnation in the Nelder-Mead algorithm using a sufficient decrease condition, C. T. Kelley, Siam J. Optim., Vol. 10, No. 1, pp. 43-55.

[17] L. Farvacque *et al.*, *Beta user's guide*, Note ESRF-COMP-87-01, 1987; J. Payet, IRF/LNS, CEA Saclay, private communication; see also J.M. Lagniel, *Recherche d'un optimum*, Note IRF/LNS/SM 87/48, CEA Saclay 1987.

[18] F. Méot and N. Willis, *Raytrace computation with Monte Carlo simulation of particle decay*, internal report CEA/LNS/88-18 CEA Saclay, 1988.

[19] H.A. Enge, *Deflecting magnets*, in **Focusing of Charged Particles**, ed. A. Septier, **Vol. II**, pp 203-264, Academic Press Inc., 1967.

[20] P. Birien et S. Valéro, *Projet de spectromètre magnétique à haute résolution pour ions lourds*, **Section IV** p.62, Note CEA-N-2215, CEA Saclay, mai 1981.

[21] V. M. Kel'man and S. Ya. Yavor, Achromatic quadrupole electron lenses, Soviet Physics - Technical Physics, vol. 6, No 12, June 1962;
S. Ya. Yavor *et als.*, *Achromatic quadrupole lenses*, NIM **26** (1964) 13-17.

[22] A. Septier et J. van Acker, *Les lentilles quadrupolaires électriques*, NIM **13** (1961) 335-355; Y. Fujita and H. Matsuda, *Third order transfer matrices for an electrostatic quadrupole lens,* NIM **123** (1975) 495-504.

[23] A. Septier, Cours du DEA de physique des particules, optique corpusculaire, Université d'Orsay, 1966-67, pp. 38-39.

[24] S. P. Karetskaya et als., Mirror-bank energy analyzers, in Advances in electronics and electron physics, Vol. 89, Acad. Press (1994) 391-491.

[25] F. Lemuet, F. Méot, *Developements in the ray-tracing code Zgoubi for 6-D multiturn tracking in FFAG rings*, NIM A **547** (2005) 638-651.

[26] P. Akishin, JINR, Dubna, 1992.

[27] M.W. Garrett, *Calculation of fields [...] by elliptic integrals*, J. Appl. Phys., **34**, 9, sept. 1963.

[28] P.F. Byrd and M.D. Friedman, *Handbook of elliptic integrals for engineers and scientists,* pp. 282-283, Springer-Verlag, Berlin, 1954.

[29] A. Tkatchenko, *Computer program UNIPOT*, SATURNE, CEA Saclay, 1982.

[30] J.L. Chuma, *PLOTDATA*, TRIUMF Design Note TRI-CO-87-03a.

[31] J.L. Hamel, *mini graphic library LIBMINIGRAF*, CEA-DSM, Saclay, 1996.

[32] E. Grorud, J.L. Laclare, G. Leleux, Résonances de dépolarisation dans SATURNE 2, Int. report GOC-GERMA 75-48/TP-28, CEA Saclay (1975), and Home Computer Codes POLAR and POPOL, IRF/LNS/GT, CEA Saclay (1975).

[33] M. Froissart et R. Stora, Dépolarisation d'un faisceau de protons polarisés dans un synchrotron, NIM 7 (1960) 297-305.

[34] J. Thirion et P. Birien, Le spectromètre II, Internal Report DPh-N/ME, CEA Saclay, 23 Déc. 1975;
H. Catz, Le spectromètre II, Internal Report DPh-N/ME, CEA Saclay, 1980.

[35] P. Pile, I-H. Chiang, K. K. Li, C. J. Kost, J. Doornbos, F. Méot et als., A two-stage separated 800-MeV/c Kaon beamline, TRIUMF and BNL Preprint (1997).

[36] F. Méot, The raytracing code Zgoubi, CERN SL/94-82 (AP) (1994), 3rd Intern. Workshop on Optimization and Inverse Problems in Electromagnetism, CERN, Geneva, Switzerland, 19-21 Sept. 1994.

[37] W. H. Press et *als.*, Numerical recipes, Cambridge Univ. Press (1987).

[38] V. O. Kostroun, Simple numerical evaluation of modified Bessel functions and integrals [...], NIM 172 (1980) 371-374.

acceleration, 113, 115, 136
AIMANT, 92, **123**, 150
ALE, 220
AUTOREF, **131**

backward ray-tracing, 217
BEND, 46, **132**
BINARY, **89**
BORO, 74, 79, 86, 136
BREVOL, **133**

CARTEMES, 43, 133, **134**, 170, 176, 177, 183, 184, 196, 218
CAVITE, 115, **136**, 137, 220
CHAMBR, 113, 140, **140**
CHANGREF, 106, 107, 131, 140, **141**, 143
checking field, 217
checking trajectories, 217
CIBLE, 104, **143**
COLLIMA, 113, **145**
constraint (FIT), 91, **94**
Cromaticity, **210**, 211, 216

DECAPOLE, 110, **147**
DIPOLE, 92, 132, 140, **148**, 153, 220
DIPOLE-M, 123, 126, **150**, 184, 223
DIPOLES, **153**
DODECAPO, 47, 110, **158**
DRIFT, **159**

EBMULT, 47, 56, 92, 110, **160**
EL2TUB, 56, **162**
ELMIR, **163**
ELMIRC, **165**
ELMULT, 56, 92, 110, 160, **167**
ELREVOL, 56, **170**, 218
END, 88, **90**
ESL, 106, **159**

FAISCEAU, **205**
FAISCNL, **205**, 219
FAISTORE, **205**, 219
FFAG magnet, radial, 154, **171**
FFAG magnet, spiral, **174**
FIN, **90**
FIT, 84, 88, 91, **91**, 92, 94
FOCALE, **207**
FOCALEZ, **208**
fringe fields
  overlapping, 153

GASCAT, **103**

HISTO, 75, 106, 146, 209, **209**, 224

IC, 218
ID, **134**
IDMAX, 80, **80**
IEX, 52, **80**, 82, 122, 134, 140, 146, 205, 214, **224**
IL, 217
IMAGE, 131, 207, **207**
IMAGES, 79, 80, 207, **207**
IMAGESZ, **208**
IMAGEZ, **208**
IMAX, **74**, **80**, 82, 86, 106, 113, 118, 136, 137, 214, 219, 223
integration step size, 224
  coded, 223

negative, 107, 217
IORDRE, 43, 49, 111, 129, 134, 149, 152, 157, 196

KPOS, **220**

LABEL, 115, 178, 206, 212, **219**

MAP2D, 46, **176**, 195
MAP2D-E, 46, **177**
MARKER, **178**
MATPROD, **179**
MATRIX, 84, 91, 94, 131, 179, **210**, 211
MCDESINT, 75, 79, 88, 104, 107, 113, 209
MCOBJET, 74, 75, 95, 113, 114, 224
Misalignement, **220**
Monte Carlo, 74, 86, 104, 113, 223
multiparticle, 113, 223
MULTIPOL, 47, 92, 110, 115, 160, 167, 180
multiturn, 73, 115, 136, 219, 233
multiturn tracking, 113

negative charge, 74, 79, 224
negative momentum, 74, 79, 224
negative rigidity, 224
NPASS, 113, 136, 206, 209, 214, 219, 223

OBJET, 79, 94, 104, 105, 114, 131, 136, 140, 207, 216, 221, 224
OBJETA, 86, 113
OCTUPOLE, 110, 182
ORDRE, 110
outpoi.lis, 183

PARTICUL, 104, 112, 118, 121, 122, 145, 160, 162, 163, 165, 168, 170, 190, 199, 201
PICKUPS, **212**, 219
PLOTDATA, 213
POISSON, 183
POLARMES, 184, 223
Positioning, **220**
PS170, 185

QUADISEX, 43, 46, 186
QUADRUPO, 47, 92, 93, 110, 140, 147, 167, 168, 180–182, 187, 192, 220

REBELOTE, 73, 74, 82, 88, 113, 115, 136, 205, 206, 209, 212, 214, 219, 220, 223
RESET, 114

SCALING, 93, 115, 136, 137, 219, 220
SEPARA, 190
SEXQUAD, 43, 186
SEXTUPOL, 47, 110, 192
SOLENOID, 193
spin tracking, 57, 113, 118, 141, 197, 204, 209, 214, 224
SPNPRNL, 214
SPNPRNLA, 214
SPNPRT, 214
SPNTRK, 88, 113, **118**, 224
SRLOSS, **121**
SRPRNT, **215**
stopped particles, 113, 114, 140, 145, 146, 205, 224
synchrotron motion, 113, 115, 136
synchrotron radiation, 64
synchrotron radiation loss, 121, 215

synchrotron radiation spectra, 122
SYNRAD, **122**

TARGET, **143**
TOSCA, 43, 89, 93, 111, 195, 220
TRAROT, 197
TWISS, 84, 91, 94, 210, 211, **216**

UNDULATOR, 198
UNIPOT, 56, 199

variable (FIT), 91, **91**
VENUS, 43, 46, 200

WIENFILT, 46, 56, 201

XCE, 220
XPAS
    negative, 107
XPAS, coded, 223
XPAS, negative, 217

YCE, 220
YMY, 203

zgoubi.dat, 206
zgoubi.fai, 205, 206
zgoubi.map, 218
zgoubi.plt, 178, 218, 219
zgoubi.res, 218
zgoubi.spn, 214
zgoubi.sre, 122
zpop, 68, 205, 214, 218