

The MAD Program

(Methodical Accelerator Design)

Version 8.19

User's Reference Manual

Hans Grote
F. Christoph Iselin

Abstract

MAD is a tool for charged-particle optics in alternating-gradient accelerators and beam lines. It can handle very large and very small accelerators and solves various problems on such machines.

Features include: Linear lattice parameter calculation, linear lattice matching, transfer matrix matching, survey calculations, error definitions, closed orbit correction, particle tracking, chromatic effects and resonances, file services, subroutines, intra-beam scattering, Lie-algebraic analysis, debugging services. Most of the Lie-algebraic algorithms used have been developed at Maryland University by Alex Dragt and his collaborators.

The present revision of this manual is valid for Version 8.19 of MAD. A set of new commands has been added to the program MAD to enhance the flexibility of tracking studies, both for LHC and for LEP. The new features include an automatic search of dynamic aperture and the calculation of early indicators, such as Lyapunov exponent, smear and detuning à la Laskar. In conjunction with more conventional MAD variables, such as normal mode chromaticities and anharmonicities, these quantities can be saved in special tables to generate correlation plots or can be used as global matching constraints, for a numerical optimisation of machine parameters related to the dynamic aperture. Local variables such as closed orbit, betatron functions, dispersion or normal mode eigenvectors are accessible via MAD tables, generated by the commands EIGEN, ENVELOPE, OPTICS, TWISS, or TWISS3. The information contained in these tables can be plotted or manipulated through string variables, either preset or user-defined.

A new sequence editor command REPLACE, Relative field errors are now allowed on all element types, A new mechanism to select elements in the ring. This mechanism is incompatible with previous versions, but more flexible.

Geneva, Switzerland
April 29, 1996

The MAD program contains the following copyright note:

CERN

EUROPEAN ORGANISATION FOR NUCLEAR RESEARCH

Program name: MAD --- Methodical Accelerator Design

CERN program library entry: T5001

Authors or contacts: Hans Grote, F. Christoph Iselin
SL Division
CERN
CH-1211 GENEVA 23
SWITZERLAND
Tel. [0041] (022) 767 36 57
chris.iselin@cern.ch

Copyright CERN, Geneva 1990 - Copyright and any other appropriate legal protection of this computer program and associated documentation reserved in all countries of the world.

Organisations collaborating with CERN may receive this program and documentation freely and without charge.

CERN undertakes no obligation for the maintenance of this program, nor responsibility for its correctness, and accepts no liability whatsoever resulting from its use.

Program and documentation are provided solely for the use of the organisation to which they are distributed.

This program may not be copied or otherwise distributed without permission. This message must be retained on this and any other authorised copies.

The material cannot be sold. CERN should be given credit in all references.

Preface

The MAD framework should make it easy to add new features in the form of program modules. The authors of MAD hope that such modules will also be contributed and documented by others. Future updates to this manual will be published in the form of complete chapters which either replace existing ones, or document new features. As required, the table of contents and the indices will be revised. The contributions of other authors are acknowledged in the relevant chapters.

Misprints and obscurity are almost inevitable in a manual of this size. Comments from readers are therefore most welcome. They may also be sent to one of the following internet addresses:

`chris.iselin@cern.ch`
`hans.grote@cern.ch`
`eberhard.keil@cern.ch`

An up-to-date version of this manual is available at

`http://hpariel.cern.ch/fci/mad/mad.html`

Contents

1	Conventions	1
1.1	Reference System	1
1.2	Closed Orbit	2
1.3	Global Reference System	2
1.3.1	Straight Elements	3
1.3.2	Bending Magnets	4
1.3.3	Rotation of the Reference System	4
1.3.4	Elements which do not Change the Local Reference	4
1.4	Sign Conventions for Magnetic Fields	5
1.5	Variables Used in MAD	5
1.5.1	Canonical Variables Describing Orbits	5
1.5.2	Normalised Variables and other Derived Quantities	6
1.5.3	Linear Lattice Functions (Optical Functions)	7
1.5.4	Chromatic Functions	7
1.5.5	Variables in the TUNES Tables	8
1.5.6	Variables in the TRACK Table	9
1.6	Physical Units Used	9
1.7	Wild Card Patterns	9
1.8	Save Sequence as a Line, SAVELINE Command	10
1.9	Save Complete Machine Structure, including Imperfections	10
1.10	Build SEQUENCE from USE'd Beam Line	11
2	Commands and Statements	12
2.1	Statements, Attributes, Options, Comments	12
2.2	Getting Help	13
2.2.1	HELP Command	13
2.2.2	SHOW Command	14
2.3	Identifiers or Labels	14
2.4	Command Attributes	14
2.4.1	Name Attributes	14
2.4.2	String Attributes	14
2.4.3	Logical Attributes	15
2.4.4	Integer Attributes	15
2.4.5	Real Expressions	15
2.4.6	Operators in Arithmetic Expressions	15
2.4.7	Operands in Arithmetic Expressions	16
2.4.8	Deferred Expressions and Random Values	17
2.4.9	Constraints	17
2.4.10	Variable Names	18
2.5	TITLE Statement	18

2.6	STOP Statement	18
2.7	OPTION Statement	18
2.8	Parameter Statements	20
2.8.1	Relations between Variable Parameters	20
2.8.2	Assignment to Parameters	20
2.8.3	Output of Parameters	21
2.9	Commands to Manipulate Global MAD Variables	21
2.10	SYSTEM: Execute Operating System Command	23
2.10.1	SYSTEM Command under UNIX	23
2.10.2	SYSTEM Command under VM/CMS	23
2.11	COMMENT/ENDCOMMENT Statements	24
3	Physical Elements and Markers	25
3.1	Input Format	25
3.2	Marker Definitions	26
3.3	Drift Space	26
3.4	Bending Magnets	26
3.5	Quadrupole	29
3.6	Sextupole	29
3.7	Octupole	30
3.8	General Thin Multipole	30
3.9	Solenoid	31
3.10	Closed Orbit Correctors	31
3.11	RF Cavity	32
3.12	Electrostatic Separator	32
3.13	Beam Position Monitor	33
3.14	Collimators (Aperture Definitions)	33
3.15	Coordinate Transformations	34
3.15.1	Rotation About the Vertical Axis	34
3.15.2	Rotation Around the Longitudinal Axis	34
3.16	BEAMBEAM Element	35
3.17	Arbitrary MATRIX Element	36
3.18	Editing Element Definitions	36
3.19	Element Classes	36
3.20	LUMP Element	37
4	Beam Lines, Sequences, Lists, Ranges, Selections	39
4.1	Beam Lines	39
4.1.1	Simple Beam Lines	39
4.1.2	Sub-lines	40
4.1.3	Reflection and Repetition	40
4.1.4	Replaceable Arguments	41
4.2	Beam Line Sequences	42
4.3	Editing a Beam Line Sequence	43
4.3.1	Making a sequence known for editing	43
4.3.2	Cyclic interchange	43
4.3.3	Installing an Element	44
4.3.4	Moving an Element	44
4.3.5	Reflecting a Sequence	44
4.3.6	Removing Elements	44
4.3.7	Replacing Elements	45

4.3.8	Example for the Sequence Editor	45
4.4	More Examples for Beam Lines	46
4.4.1	CERN SPS Lattice	46
4.4.2	LEP Lattice	46
4.5	Beam Line References	47
4.6	Replacement Lists	47
4.7	Ranges and Sets of Observation Points in a Line	49
4.7.1	Examples for Ranges	50
5	Action Commands	51
5.1	USE Statement	51
5.2	COGUESS Command	53
5.3	BEAM Command	53
5.4	PRINT and SELECT Statements	55
5.5	SPLIT Command, Request Interpolation for OPTICS	56
5.6	SURVEY Statement	56
5.7	TWISS Statement	57
5.7.1	Twiss Parameters for a Period	58
5.7.2	Initial Values for Twiss Parameters taken from a Periodic Line	58
5.7.3	Twiss Parameters with Numerical Initial Values	59
5.7.4	SAVEBETA Command, Save Lattice Parameters for Later Use	59
5.8	IBS Command, Intra-Beam Scattering	60
5.9	Emittances, Eigenvectors, Normal Modes, Beam Envelope	60
5.9.1	Electron Beam Parameters	60
5.9.2	Representations of Beam with Full Coupling	61
5.10	DYNAMIC and STATIC, Lie-Algebraic Analysis	63
5.11	OPTICS Command	64
5.12	Examples for SPLIT and OPTICS Commands	66
5.13	SECTORMAP Command, Maps for Parts of the Ring	68
6	Calculation of Beam Parameters	69
6.1	Introduction	69
6.2	BMPM Command	70
6.3	Output of overall machine parameters	71
6.3.1	Synchrotron integrals	71
6.3.2	Machine parameters	71
6.3.3	Beam parameters and luminosities	71
6.3.4	RF related parameters	72
6.4	Evaluation of the beam size	73
6.5	Synchrotron radiation, Calculated for One Beam	73
6.6	Closed orbit calculations	74
7	File and Pool Handling	75
7.1	I/O Data Sets (Files)	75
7.2	ASSIGN Statement	76
7.3	SAVE, CALL, and RETURN Statements	76
7.4	Element Excitations	77
7.5	Archiving and Retrieving Dynamic Tables	78
7.5.1	ARCHIVE Command	78
7.5.2	RETRIEVE Command	78
7.6	PACKMEMORY Command, Garbage Removal	78

7.7	STATUS	78
7.8	Dumping and Reloading the Memory Pool	79
8	Error Definitions	80
8.1	Misalignment Definitions	80
8.2	Field Error Definitions	81
8.2.1	Field Errors in Bending Magnets	83
8.2.2	Field Errors in Quadrupoles	84
8.2.3	Field Errors in Sextupoles	84
8.2.4	Field Errors in Multipoles	84
8.2.5	Field Errors in Orbit Correctors	84
8.3	Error Option Command	84
8.4	EPRINT: Print Errors	84
8.5	ESAVE: Save Errors	85
9	Closed Orbit Commands	86
9.1	GETDISP and PUTDISP Statements	86
9.2	GETKICK and PUTKICK Statements	87
9.3	GETORBIT and PUTORBIT Statements	87
9.4	USEKICK and USEMONITOR, Activate/deactivate elements	88
9.5	MICADO Statement	89
9.6	CORRECT Statement	89
10	Plotting and Tabulating	91
10.1	SETPLOT and RESPLOT Commands	91
10.2	PLOT Command	92
10.3	Axis and frame selection	94
10.3.1	TWISS table	94
10.3.2	TRACK table	95
10.3.3	TUNES table	95
10.4	Tune Grid	95
10.5	TABLE and STRING Commands	96
10.5.1	Plotting Composite Values	97
11	HARMON Module	98
11.1	Activating and Deactivating HARMON	98
11.2	Chromaticity Calculation or Tuning	99
11.3	Resonance Calculations	99
11.4	Distortion Functions	100
11.5	Example for a HARMON Sequence	101
12	Matching Module	103
12.1	Activation and Deactivation of the Matching Module	104
12.1.1	Matching a Periodic Cell	104
12.1.2	Insertion Matching	104
12.1.3	End of Matching Run	105
12.2	Variable Parameters	105
12.3	Constraints	106
12.3.1	Simple Constraints	106
12.3.2	Sub-Period Constraint	107
12.3.3	Constraints on Transfer map	108

12.3.4	Matching Weights	108
12.4	GLOBAL: Constraints on Global Quantities	109
12.5	Matching Output Level	110
12.6	Matching Methods	110
12.6.1	LMDIF, Gradient Minimisation	110
12.6.2	MIGRAD, Gradient Minimisation	111
12.6.3	SIMPLEX, Minimisation by Simplex Method	111
12.7	Matching Examples	111
12.7.1	Simple Periodic Beam Line	111
12.7.2	Insertion Matching	112
12.7.3	A Matching Example for LEP, Version 11	112
12.7.4	Examples for Complex Matching Constraints	113
13	Tracking Module	115
13.1	Activation and Deactivation of the Tracking Module	115
13.2	NOISE Statement	116
13.3	OBSERVE Statement	116
13.4	START Statement	117
13.5	TSAVE Statement	117
13.6	RUN Statement	118
13.7	DYNAPSTART and DYNAP: Dynamic Aperture search	118
13.8	A Tracking Example	121
14	Subroutines and Procedures	122
14.1	SUBROUTINE and ENDSUBROUTINE	122
14.2	CALLSUBROUTINE	123
14.3	DO and ENDDO	123
15	Known Defects of Version 8	124
15.1	Definitions	124
15.1.1	Treatment of $\partial^2 Q / \partial \delta^2$ in TWISS	124
A	Format of the SURVEY, STRUCTURE, and TWISS Files	125
A.1	Common Output	125
A.2	SURVEY Output	126
A.3	TWISS Output	127
A.4	CHROM Output	127
B	Format for File Names	128
B.1	IBM VM/CMS System	128
B.1.1	Standard Streams	128
B.1.2	User-Defined Files	128
B.2	UNIX and UNICOS Systems	129
B.2.1	Standard Stream Names	129
B.2.2	User-Defined Files	130
B.3	VAX VMS System	130
B.3.1	Standard Stream Names	130
B.3.2	User-Defined Files	131

C	Format of TFS Files	132
C.1	Descriptor Lines	132
C.2	Column Formats	133

List of Figures

1.1	Local Reference System	1
1.2	Global Reference System	2
2.1	Contour plot of the dynamic aperture in the $x - y$ plane (left) and corresponding linear betatron invariant versus the $x - y$ angle in degrees (right), both expressed in terms of the r.m.s. beam size, for LHC version 4.2 at injection, with systematic and random magnetic imperfections: tracking is performed over 1000 turns, including synchrotron oscillations with $\Delta p/p = 10^{-3}$ (see the sample file <code>dynapcont.mad</code>). Note that the conventional definition of dynamic aperture may be rather sensitive to the choice of the initial phases (usually set to zero). As shown in the picture on the right, the result is comprised between the minimum and maximum value of the betatron invariant over the tracking interval, related to the smear.	22
2.2	Tunes versus normalised amplitude at constant $\Delta p/p$ (left) and chromatic detunings (right) for LHC version 4.2 at injection, with systematic and random magnetic imperfections (see the sample file <code>detuning.mad</code>).	22
2.3	Tune footprint at constant $\Delta p/p$ (right) obtained by tracking with the <code>fastune</code> option and corresponding to a prescribed scan in the $x - y$ plane (left), for LHC version 4.2 at injection, with systematic and random magnetic imperfections (see the sample file <code>footprint.mad</code>).	23
3.1	Reference System for Straight Beam Elements	27
3.2	Reference System for a Rectangular Bending Magnet	27
3.3	Reference System for a Sector Bending Magnet	28
3.4	Reference System for a Rotation Around the y-Axis	34
3.5	Reference System for a Rotation Around the s-Axis	35
6.1	Variation of the beam-beam tune shifts	72
8.1	Example of Misplacement in the (x, s) -plane	81
8.2	Example of Misplacement in the (x, y) -plane	82
8.3	Example of Misplacement in the (y, s) -plane	82
8.4	Example of Read Errors in a monitor	83

List of Tables

1.1	Physical Units used by MAD	10
2.1	Utility Commands	12
2.2	Symbolic Constants	17
2.3	Initial Defaults for command options	20
3.1	Element Definition Commands	25
4.1	Line Definition Commands	39
5.1	Environment Setting and Action Commands	51
5.2	Default Beam Data	54
5.3	An OPTICS Output Table Example	66
6.1	Electron Beam Parameter Command	69
7.1	File Handling Commands	75
7.2	Standard Files Used by MAD	75
8.1	Error Definition Commands	80
9.1	Commands Related to the Closed Orbit	86
10.1	Plotting Commands	91
11.1	HARMON Commands	98
12.1	Matching Commands	103
12.2	Default Matching Weights	108
13.1	Tracking Commands	115
14.1	Subroutine Commands	122
A.1	Element Data in Position Records	126
B.1	Standard Files Used by MAD, IBM-VM/CMS Version	128
B.2	Standard Files Used by MAD, UNIX Version	130
B.3	Standard Files Used by MAD, VAX-VMS Version	130
C.1	Column Formats used in TFS Tables	133

Chapter 1

Conventions

1.1 Reference System

The accelerator and/or beam line to be studied is described as a sequence of beam elements placed sequentially along a reference orbit. The reference orbit is the path of a charged particle having the central design momentum of the accelerator through idealised magnets with no fringe fields (see Figure 1.1).

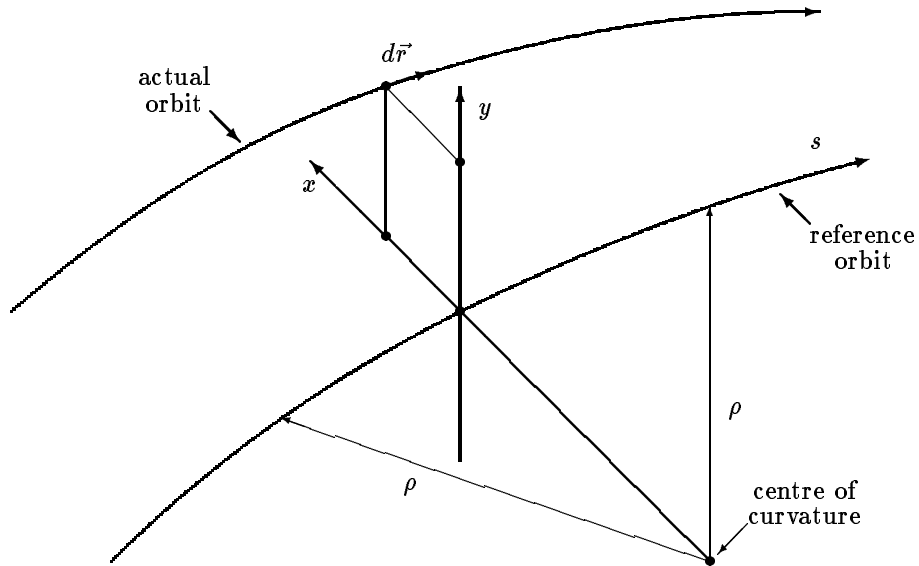


Figure 1.1: Local Reference System

The reference orbit consists of a series of straight line segments and circular arcs. It is defined under the assumption that all elements are perfectly aligned. The accompanying tripod of the reference orbit spans a local curvilinear right handed coordinate system (x, y, s) . The local s -axis is the tangent to the reference orbit. The two other axes are perpendicular to the reference orbit and are labelled x (in the bend plane) and y (perpendicular to the bend plane).

1.2 Closed Orbit

Due to various errors like misalignment errors, field errors, fringe fields etc., the closed orbit does not coincide with the reference orbit. It also changes with the momentum error. The closed orbit is described with respect to the reference orbit, using the local reference system (x, y, s) . It is evaluated including any nonlinear effects.

MAD also computes the betatron and synchrotron oscillations with respect to the closed orbit. Results are given in the local (x, y, s) -system defined by the reference orbit.

1.3 Global Reference System

The reference orbit of the accelerator is uniquely defined by the sequence of physical elements. The local reference system (x, y, s) may thus be referred to a global Cartesian coordinate system (X, Y, Z) (see Figure 1.2). The positions between beam elements are numbered $0, \dots, i, \dots, n$. The local reference system (x_i, y_i, z_i) at position i , i.e. the displacement and direction of the reference orbit with respect to the system (X, Y, Z) are defined by three displacements (X_i, Y_i, Z_i) and three angles $(\theta_i, \phi_i, \psi_i)$. The above quantities are defined more precisely as follows:

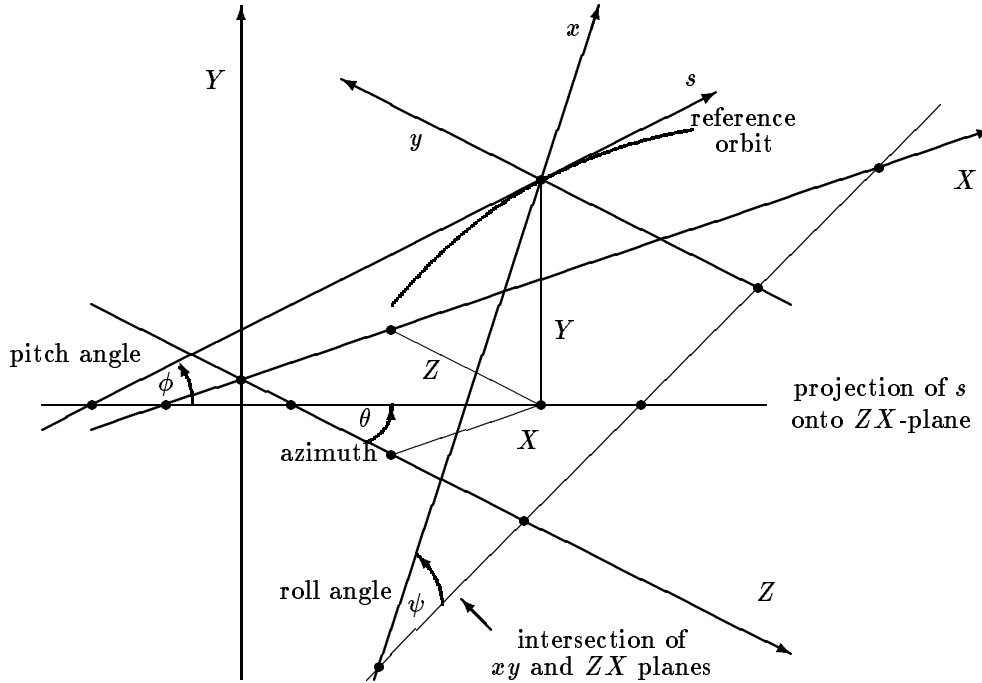


Figure 1.2: Global Reference System

X Displacement of the local origin in X -direction.

Y Displacement of the local origin in Y -direction.

Z Displacement of the local origin in Z -direction.

THETA Angle of rotation (azimuth) about the global Y -axis, between the global Z -axis and the projection of the reference orbit onto the (Z, X) -plane. A positive angle θ forms a right-hand screw with the Y -axis.

PHI Elevation angle, i.e. the angle between the reference orbit and its projection onto the (Z, X) -plane. A positive angle ϕ correspond to increasing Y . If only horizontal bends are present, the reference orbit remains in the (Z, X) -plane. In this case ϕ is always zero.

PSI Roll angle about the local s -axis, i.e. the angle between the intersection of the (x, y) and (Z, X) -planes and the local x -axis. A positive angle ψ forms a right-hand screw with the s -axis.

The angles (θ, ϕ, ψ) are *not* the Euler angles. The reference orbit starts at the origin and points by default in the direction of the positive Z -axis. The initial local axes (x, y, s) coincide with the global axes (X, Y, Z) in this order. The six quantities $(X_0, Y_0, Z_0, \theta_0, \phi_0, \psi_0)$ thus all have zero initial values by default. The program user may however specify different initial conditions.

Internally the displacement is described by a vector V , and the orientation by a unitary matrix W . The column vectors of W are the unit vectors spanning the local coordinate axes in the order (x, y, s) . V and W have the values

$$V = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}, \quad W = \Theta \Phi \Psi$$

where

$$\Theta = \begin{pmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{pmatrix}, \quad \Phi = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & \sin \phi \\ 0 & -\sin \phi & \cos \phi \end{pmatrix}, \quad \Psi = \begin{pmatrix} \cos \psi & -\sin \psi & 0 \\ \sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

The reference orbit should be closed and it should not be twisted. This means that the displacement of the local reference system must be periodic with the revolution frequency of the accelerator, while the position angles must be periodic (modulo 2π) with the revolution frequency. If ψ is not periodic (modulo 2π), coupling effects are introduced. When advancing through a beam element, MAD computes V_i and W_i by the recurrence relations

$$V_i = W_{i-1} R_i + V_{i-1}, \quad W_i = W_{i-1} S_i$$

The vector R_i is the displacement and the matrix S_i is the rotation of the local reference system at the exit of the element i with respect to the entrance of the same element. The values of R and S are listed below for each physical element type.

1.3.1 Straight Elements

In straight elements the local reference system is simply translated along the local s -axis by the length of the element. This is true for

- Drift spaces,
- Quadrupoles,
- Sextupoles,
- Octupoles,
- Solenoids,
- RF cavities,
- Electrostatic separators,

- Closed orbit correctors,
- Beam monitors.

The corresponding R and S are

$$R = \begin{pmatrix} 0 \\ 0 \\ L \end{pmatrix}, \quad S = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

A rotation of the element about the s -axis has no effect on R and S , since the rotations of the reference system before and after the element cancel.

1.3.2 Bending Magnets

Bending magnets have a curved reference orbit. For both rectangular and sector bending magnets

$$R = \begin{pmatrix} \rho(\cos \alpha - 1) \\ 0 \\ \rho \sin \alpha \end{pmatrix}, \quad S = \begin{pmatrix} \cos \alpha & 0 & -\sin \alpha \\ 0 & 1 & 0 \\ \sin \alpha & 0 & \cos \alpha \end{pmatrix},$$

where α is the bend angle. A positive bend angle represents a bend to the right, i.e. towards negative x values. For sector bending magnets, the bend radius is given by $\rho = L/\alpha$, and for rectangular bending magnets it has the value $\rho = L/(2 \sin(\alpha/2))$. If the magnet is rotated about the s -axis by an angle ψ , R and S are transformed by

$$\overline{R} = TR, \quad \overline{S} = TST^{-1}$$

where T is the rotation matrix

$$T = \begin{pmatrix} \cos \psi & -\sin \psi & 0 \\ \sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

The special value $\psi = \pi/2$ represents a bend down.

1.3.3 Rotation of the Reference System

For a rotation of the reference system by an angle ψ about the beam (s) axis:

$$S = \begin{pmatrix} \cos \psi & -\sin \psi & 0 \\ \sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

while for a rotation of the reference system by an angle θ about the vertical (y) axis:

$$S = \begin{pmatrix} \cos \theta & 0 & -\sin \theta \\ 0 & 1 & 0 \\ \sin \theta & 0 & \cos \theta \end{pmatrix}.$$

In both cases the displacement R is zero.

1.3.4 Elements which do not Change the Local Reference

MARKER elements do not affect the reference orbit. They are ignored for geometry calculations.

1.4 Sign Conventions for Magnetic Fields

The MAD program uses the following Taylor expansion [8] for the field on the mid-plane ($y = 0$):

$$B_y(x, 0) = \sum_{n=0}^{\infty} \frac{B_n x^n}{n!}$$

Note the factorial in the denominator. The field coefficients have the following meaning:

B_0 Dipole field, with a positive value in the positive y direction; a positive field bends a positively charged particle to the right.

B_1 Quadrupole coefficient $B_1 = \partial B_y / \partial x$; a positive value corresponds to horizontal focussing of a positively charged particle.

B_2 Sextupole coefficient $B_2 = \partial^2 B_y / \partial x^2$.

B_3 Octupole coefficient $B_3 = \partial^3 B_y / \partial x^3$.

Using this expansion and the curvature h of the reference orbit, the longitudinal component of the vector potential to order 4 is:

$$\begin{aligned} A_s = & + B_0 \left(x - \frac{hx^2}{2(1+hx)} \right) + B_1 \left(\frac{1}{2}(x^2 - y^2) - \frac{h}{6}x^3 + \frac{h^2}{24}(4x^4 - y^4) + \dots \right) \\ & + B_2 \left(\frac{1}{6}(x^3 - 3xy^2) - \frac{h}{24}(x^4 - y^4) + \dots \right) + B_3 \left(\frac{1}{24}(x^4 - 6x^2y^2 + y^4) + \dots \right) + \dots \end{aligned}$$

Taking $\text{curl} A_s$ in curvilinear coordinates the field components can be computed as

$$\begin{aligned} B_x(x, y) = & + B_1 \left(y + \frac{h^2}{6}y^3 + \dots \right) + B_2 \left(xy - \frac{h^3}{6}y^3 + \dots \right) + B_3 \left(\frac{1}{6}(3x^2y - y^3) + \dots \right) + \dots \\ B_y(x, y) = & + B_0 + B_1 \left(x - \frac{h}{2}y^2 + \frac{h^2}{2}xy^2 + \dots \right) \\ & + B_2 \left(\frac{1}{2}(x^2 - y^2) - \frac{h}{2}xy^2 + \dots \right) + B_3 \left(\frac{1}{6}(x^3 - 3xy^2) + \dots \right) + \dots \end{aligned}$$

It can be easily verified that $\text{curl} \vec{B}$ and $\text{div} \vec{B}$ are both zero to the order of the B_3 term. Introducing the magnetic rigidity $B\rho$, the multipole coefficients are computed as $K_n = eB_n/p_0 = B_n/B\rho$.

1.5 Variables Used in MAD

For each variable the physical units are listed in square brackets.

1.5.1 Canonical Variables Describing Orbits

MAD uses the following canonical variables to describe the motion of particles:

X Horizontal position x of the (closed) orbit, referred to the ideal orbit [m].

PX Horizontal canonical momentum p_x of the (closed) orbit referred to the ideal orbit, divided by the reference momentum: $PX = p_x/p_0$, [1].

Y Vertical position y of the (closed) orbit, referred to the ideal orbit [m].

PY Vertical canonical momentum p_y of the (closed) orbit referred to the ideal orbit, divided by the reference momentum: $PY = p_y/p_0$, [1].

T Velocity of light times the negative time difference with respect to the reference particle: $T = -c\Delta t$, [m]. A positive T means that the particle arrives ahead of the reference particle.

PT Energy error, divided by the reference momentum times the velocity of light: $PT = \delta = \Delta E/p_0 c$, [1]. This value is only non-zero when synchrotron motion is present. It describes the deviation of the particle from the orbit of a particle with the momentum error DELTAP.

DELTAP Difference of the reference momentum and the design momentum, divided by the reference momentum: $DELTAP = \delta_s = \Delta p_s/p_0$, [1]. This quantity is used to normalize all element strengths. See Chapter 15.1.1 for explanation.

The independent variable is:

S Arc length along the reference orbit, [m].

In the limit of fully relativistic particles ($v = c$, $E = pc$), the variables T , PT used here agree with s , δ used in TRANSPORT [9]. This means that T becomes the negative path length difference, while PT becomes the fractional momentum error. The reference momentum p_s must be constant in order to keep the system canonical.

1.5.2 Normalised Variables and other Derived Quantities

XN The normalised horizontal displacement $x_n = \Re E_1^T S Z$ [$\text{m}^{-1/2}$].

PXN The normalised horizontal transverse momentum $p_{xn} = \Im E_1^T S Z$ [$\text{m}^{-1/2}$].

WX The horizontal Courant-Snyder invariant $W_x = x_n^2 + p_{xn}^2$ [m^{-1}].

PHIX The horizontal phase $\phi_x = -\arctan(p_{xn}/x_n)/2\pi$ [1].

YN The normalised vertical displacement $y_n = \Re E_2^T S Z$ [$\text{m}^{-1/2}$].

PYN The normalised vertical transverse momentum $p_{yn} = \Im E_2^T S Z$ [$\text{m}^{-1/2}$].

WY The vertical Courant-Snyder invariant $W_y = y_n^2 + p_{yn}^2$ [m^{-1}].

PHIY The vertical phase $\phi_y = -\arctan(p_{yn}/y_n)/2\pi$.

TN The normalised vertical displacement $t_n = \Re E_3^T S Z$ [$\text{m}^{-1/2}$].

PTN The normalised vertical transverse momentum $p_{tn} = \Im E_3^T S Z$ [$\text{m}^{-1/2}$].

WT The longitudinal invariant $W_t = t_n^2 + p_{tn}^2$ [m^{-1}].

PHIT The vertical phase $\phi_t = -\arctan(p_{tn}/t_n)/2\pi$ [1].

in the above formulas Z is the phase space vector

$$Z = \begin{pmatrix} x \\ p_x \\ y \\ p_y \\ t \\ p_t \end{pmatrix},$$

the matrix S is the “symplectic unit matrix”

$$S = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 0 \end{pmatrix},$$

and the vectors E_i are the three complex eigenvectors.

1.5.3 Linear Lattice Functions (Optical Functions)

Several MAD commands refer to linear lattice functions. Since MAD uses the canonical momenta p_x, p_y instead of the slopes x', y' , their definitions differ slightly from those in Reference [13]. The linear lattice functions are known to MAD under the following names:

BETX Amplitude function β_x , [m].

ALFX Correlation function $\alpha_x = -(1/2)(\partial\beta_x/\partial s)$, [1].

MUX Phase function μ_x , $[2\pi]$.

DX Dispersion of x , i.e. $D_x = \partial x/\partial\delta$, [m].

DPX Dispersion of p_x/p_0 , i.e. $D_{p_x} = (1/p_0)(\partial p_x/\partial\delta)$, [1].

BETY Amplitude function β_y , [m].

ALFY Correlation function $\alpha_y = -(1/2)(\partial\beta_y/\partial s)$, [1].

MUY Phase function μ_y , $[2\pi]$.

DY Dispersion of y , i.e. $D_y = \partial y/\partial\delta$, [m].

DPY Dispersion of p_y/p_0 , i.e. $D_{p_y} = (1/p_0)(\partial p_y/\partial\delta)$, [1].

The TWISS table also defines the following expressions which can be used in plots:

GAMX $\gamma_x = (1 + \alpha_x)/\beta_x$,

GAMY $\gamma_y = (1 + \alpha_y)/\beta_y$,

SIGX $\sigma_x = \sqrt{\beta_x E_x}$,

SIGY $\sigma_y = \sqrt{\beta_y E_y}$,

1.5.4 Chromatic Functions

Several MAD commands refer to the chromatic functions. Since MAD uses the canonical momenta p_x, p_y instead of the slopes x', y' , their definitions differ slightly from those in Reference [30]. The chromatic functions are known to MAD under the following names:

WX Chromatic amplitude function $W_x = \sqrt{a_x^2 + b_x^2}$, [1], where

$$b_x = \frac{1}{\beta_x} \frac{d\beta_x}{d\delta}, \quad a_x = \frac{d\alpha_x}{d\delta} - \frac{\alpha_x}{\beta_x} \frac{d\beta_x}{d\delta}.$$

PHIX Chromatic phase function $\Phi_x = \arctan(a_x/b_x)$, $[2\pi]$.

DMUX Chromatic derivative of phase function $d\mu_x/d\delta$, $[2\pi]$.

DDX Chromatic derivative of dispersion $dD_x/d\delta = d^2x/d\delta^2$, [m].

DDPX Chromatic derivative of dispersion $dD_{p_x}/d\delta = (1/p_0)(d^2p_x/d\delta^2)$, [1].

WY Chromatic amplitude function $W_y = \sqrt{a_y^2 + b_y^2}$, [1], where

$$b_y = \frac{1}{\beta_y} \frac{d\beta_y}{d\delta}, \quad a_y = \frac{d\alpha_y}{d\delta} - \frac{\alpha_y}{\beta_y} \frac{d\beta_y}{d\delta}.$$

PHIY Chromatic phase function $\Phi_y = \arctan(a_y/b_y)$, $[2\pi]$.

DMUY Chromatic derivative of phase function $d\mu_y/d\delta$, $[2\pi]$.

DDY Chromatic derivative of dispersion $dD_y/d\delta = d^2y/d\delta^2$, [m].

DDPY Chromatic derivative of dispersion $dD_{p_y}/d\delta = (1/p_0)(d^2p_y/d\delta^2)$, [1].

1.5.5 Variables in the TUNES Tables

The TUNES option of the TWISS command produces a table which contains the following variables:

DELTAP Energy difference, divided by the reference momentum times the velocity of light: $\text{DELTAP} = \delta_s = \Delta E/p_0c$, [1].

ALFA The momentum compaction $\alpha = \frac{\Delta C/C}{\delta}$ [1].

GAMMATR The transition energy $\gamma_{tr} = E_{tr}/m_0c^2$ [1].

QX The horizontal tune [1].

QY The vertical tune [1].

XIX The horizontal chromaticity $\partial Q_x/\partial\delta$ [1].

XIY The vertical chromaticity $\partial Q_y/\partial\delta$ [1].

XRMS The horizontal r.m.s. closed orbit deviation [m].

YRMS The vertical r.m.s. closed orbit deviation [m].

XMAX The absolute maximum of horizontal closed orbit deviation [m].

YMAX The absolute maximum of vertical closed orbit deviation [m].

BXMAX The largest horizontal β_x [m].

BYMAX The largest vertical β_y [m].

DXMAX The absolutely largest horizontal dispersion [m].

DYMAX The absolutely largest vertical dispersion [m].

1.5.6 Variables in the TRACK Table

The command RUN writes tables with the following variables:

X Horizontal position x of the orbit, referred to the ideal orbit [m].

PX Horizontal canonical momentum p_x of the orbit referred to the ideal orbit, divided by the reference momentum: $PX = p_x/p_s$, [1].

Y Vertical position y of the orbit, referred to the ideal orbit [m].

PY Vertical canonical momentum p_y of the orbit referred to the ideal orbit, divided by the reference momentum: $PY = p_y/p_s$, [1].

T Velocity of light times the negative time difference with respect to the reference particle: $T = -c\Delta t$, [m]. A positive T means that the particle arrives ahead of the reference particle.

DELTAP Energy difference, divided by the reference momentum times the velocity of light: $DELTAP = \delta = \Delta p/p_s c$, [1].

When tracking Lyapunov companions, the TRACK table defines the following dependent expressions:

DISTANCE $\sqrt{x^2 + p_x^2 + y^2 + p_y^2 + t^2 + p_t^2}/d_0$, the relative Lyapunov distance between the two particles.

LYAPUNOV $\log(DISTANCE)/TURNS$, The estimated Lyapunov Exponent.

LOGDIST $\log(DISTANCE)$, the natural logarithm of the relative distance.

LOGTURNS $\log n$, the natural logarithm of the turn number.

1.6 Physical Units Used

Throughout the computations MAD uses international (SI, Système International) units. These units are summarised in Table 1.6.

1.7 Wild Card Patterns

Some commands allow selection of items via “wild-card” strings. Such a pattern string *must* be enclosed in single or double quotes; and the case of letters is significant. The meaning of special characters is similar to the UNIX grep utility:

. Stands for an arbitrary character,

[xyz] Stands for one character belonging to the string contained in brackets (example: [abc] means one of a, b, c).

[a-ex-z...] Stands for ranges of characters (example: [a-zA-Z] means any letter).

* Allows zero or more repetitions of the preceding item (example: [A-Z]* means zero or more upper-case letters).

[\backslash-c] Removes the special meaning of character c.

All other characters stand for themselves. Examples:

```
REMOVE,PATTERN="D. ."
```

```
SAVE,PATTERN="K.*QD.*\(\backslashbackslash\).R1"
```

Table 1.1: Physical Units used by MAD

Quantity	Unit
Length	m (metres)
Angle	rad (radians)
Quadrupole coefficient	m^{-2}
Multipole coefficient, 2n poles	m^{-n}
Electric voltage	MV (Megavolts)
Electric field strength	MV/m
Frequency	MHz (Megahertz)
Phase angles	multiples of 2π
Particle energy	GeV
Particle mass	GeV/c^2
Particle momentum	GeV/c
Beam current	A (Ampères)
Particle charge	e (elementary charges)
Impedances	$M\Omega$ (Megohms)
Emittances	$\pi \cdot m$
RF power	MW (Megawatts)
Higher mode loss factor	V/pc

The first command removes all elements whose names have exactly three characters and begin with the letter D. The second command saves definitions beginning with the letter K, containing the string QD, and ending with the string .R1. The two occurrences of .* each stand for an arbitrary number (including zero) of any character, and the occurrence \. stands for a literal period.

1.8 Save Sequence as a Line, Saveline Command

A representation of the range set in the latest USE can be saved as an equivalent LINE definition using the command

```
SAVELINE,NAME=line-name,FILENAME=filename
```

where

NAME is the name to be given on the generated LINE definition,

FILENAME is the name of the file to be written.

The currently active range of the latest USE command will be written together with all parameter and element definitions required. The definitions will all be given in terms of constants, and any dependencies between element parameters will be lost.

1.9 Save Complete Machine Structure, including Imperfections

A representation of the range set in the latest USE can be saved as an ASCII file by the command

```
STRUCTURE,FILENAME=filename,ORDER=integer
```

where

FILENAME is the name of the file to be written.

ORDER is the maximum order to be written for thin multipoles.

The currently active range of the latest USE command will be written in the form of a flat ASCII file. The format is described in Appendix A.

1.10 Build SEQUENCE from USE'd Beam Line

A representation of the range set in the latest USE can be converted to a beam line sequence (see Section 4.2) by the command

```
MAKESEQUENCE, LABEL=name, REFER=name
```

where

LABEL is the name to be given to the new sequence.

REFER defines the reference points. It may be one of

ENTRY Reference points are at each element entrance,

CENTRE Reference points are at each element centre,

EXIT Reference points are at each element exit.

The currently active range of the latest USE command will be converted to a beam line sequence. The end marker is given the name end. The new sequence can be edited by the sequence editor, mentioned in a subsequent USE command, or saved on a file by a SAVE command.

Chapter 2

Commands and Statements

Table 2.1: Utility Commands

Name	Meaning	Section
HELP	Help on command names	2.2.1
SHOW	Help on defined names	2.2.2
TITLE	Define page header for output	2.5
STOP	End program run	2.6
OPTION	Set command options	2.7
:=	Define parameter dependencies	2.8.1
SET	Set parameter value	2.8.2
VALUE	Show parameter values	2.8.3
SYSTEM	Execute operating system command	2.10

Input for MAD is free format. Blank characters or blank input lines do not affect program execution. Input lines are normally printed on the ECHO file as soon as they are read, but this feature can be turned off for long input files. In the following descriptions, words in lower case stand for syntactic units which are to be replaced by actual text. UPPER CASE is used for keywords or names. These must be entered as shown. Items enclosed in braces (`{...}`) may be repeated any number of times, including zero times. A vertical bar (`|`) represents an alternative.

2.1 Statements, Attributes, Options, Comments

The input file consists of a sequence of commands, also known as statements. Normally a statement occupies a single input line. Several statements may be placed on the same line, if they are separated by semicolons (`;`). A longer statement can be continued on several input lines. All but the last line of the statement are then terminated by an ampersand (`&`). Blank input lines, or input lines beginning with an exclamation mark are accepted as comment lines. When an ampersand (`&`) or an exclamation mark (`!`) is found on an input line, the remaining characters of the line are skipped. After an ampersand (`&`) MAD expects continuation of the statement on the next input line, while an exclamation mark (`!`) terminates the statement. An input line cannot be longer than 80 characters; the length of one command is limited at 4000 non-blank characters.

The general format for a command is

```
label: keyword {,attribute}
```

It has three parts:

`label` Gives a name to the stored command. A `label` is required for a definition statement.

keyword Identifies the action desired.

attribute Most commands require attributes for their operation. These are normally entered in the form

```
attribute-name=attribute-value
```

The attribute-name selects the attribute, and attribute-value gives it a value.

The format of label and keyword is described in Section 2.3.

In many cases the input can be abbreviated by omitting attribute-name and the equals sign. The attribute-values must then be entered in the order in which attributes appear in the command dictionary. For some attributes it is sufficient to enter the name only. The attribute is then given a default value taken from the command dictionary. Example: TILT attribute for various magnets.

Whenever this makes sense, an attribute can be left out altogether. To avoid overloading the syntax descriptions, this usage is not indicated.

The command dictionary also defines one of the following types for each command attributes:

- Name attribute (see Section 2.4.1),
- String attribute (see Section 2.4.2),
- Logical attribute (see Section 2.4.3),
- Integer attribute (see Section 2.4.4),
- Real expression (see Section 2.4.5),
- Deferred expression (see Section 2.4.8),
- Constraint (see Section 2.4.9),
- Variable name (see Section 2.4.10),
- Line definition (see Section 4.5),
- Range selection (see Section 4.7),

When a command is entered with a label, MAD keeps it in memory. This allows repeated execution of the same command by just entering its label. If the label of such a command appears together with new attributes, the attributes will be replaced first:

```
QF: QUADRUPOLE,L=1,K1=0.01    ! first definition of QF
QF,L=2                        ! redefinition of QF, new length
TW1: TWISS,BETX=1,BETY=1      ! first execution of TW1
                                ! with BETX=1, BETY=1
TW1,BETX=2,BETY=3             ! second execution of TW1
                                ! with BETX=2, BETY=3
```

2.2 Getting Help

2.2.1 HELP Command

A user who is uncertain about the attributes of a command should try the HELP command

```
HELP,keyword
```


The program then prints the list of attributes for the command keyword with their type and, if they exist, their defaults and limits. HELP entered alone lists all available commands. Examples:

```
HELP          ! List all valid keywords
HELP,TWISS    ! List attributes of "TWISS"
```

2.2.2 SHOW Command

The SHOW statement displays the current values of a command, element, line, or parameter.

```
SHOW,name
```

It displays the named command or definition on the ECHO file in the same format as the original input statement used to create the item viewed. SHOW entered alone lists all known definitions. Examples:

```
SHOW          ! Show all defined names
SHOW,QD       ! Show definition of QD
```

2.3 Identifiers or Labels

A label begins with a letter, followed by up to fifteen letters, digits, decimal points (.), primes ('), or underscores (_). Characters beyond the sixteenth are dropped, and the resulting sequence must be unique. Other special characters are allowed, if they are enclosed in single or double quotes. It makes no difference which type of quotes is used, as long as the same are used at both ends. The preferred form is double quotes.

A label may refer to a keyword, an element, a beam-line, a parameter, etc. Once it is defined, a reference can be abbreviated by dropping trailing characters, as long as the abbreviation fits only one defined label. Note that abbreviations may introduce ambiguities which cannot always be resolved correctly by MAD.

MAD maintains two name spaces, one for keywords, and one for user-defined names. Thus no conflict arises if a user-defined name is the same as a predefined keyword.

2.4 Command Attributes

2.4.1 Name Attributes

A name attribute often selects one of a set of options:

```
RUN,METHOD=TRANSPORT    ! track by the "TRANSPORT" method
```

It may also refer to a user-defined object:

```
ARCHIVE,TABLE=TWISS       ! archive the table "TWISS"
```

2.4.2 String Attributes

A string attribute makes alphanumeric information available, e.g. a title or a file name. Examples:

```
TITLE,"This is a title for the program run"
POOLDUMP,FILENAME="pool.dump.file"
```

2.4.3 Logical Attributes

Many commands in MAD require the setting of logical values (flags) to represent the on/off state of an option. A logical value flag can be set in several ways:

```
\ flag | flag=.YES. | flag=.TRUE. | flag=.T. | flag=.ON.
```

It can be reset by any of the following:

```
-flag | flag=.NO. | flag=.FALSE. | flag=.F. | flag=.OFF.
```

Example:

```
USE,...,SYMM           ! the beam line is symmetric
```

The default for a logical flag is always .FALSE..

2.4.4 Integer Attributes

An integer attribute usually denotes a count. Example:

```
USE,...,SUPER=4        ! there are four superperiods
```

2.4.5 Real Expressions

To facilitate the definition of interdependent quantities, any real value can be entered as an arithmetic expression. When a value used in an expression is redefined by the user or changed in a matching process, the expression is reevaluated. Expression definitions may be entered in any order. MAD evaluates them in the correct order before it performs any computation. At evaluation time all operands used must have values assigned.

2.4.6 Operators in Arithmetic Expressions

An expression can be formed using the following operators:

- *Arithmetic operators:*

- + Addition,
- Subtraction,
- * Multiplication,
- / Division,
- ^ Exponentiation.

- *Ordinary functions:*

- SQRT(X) Square root,
- LOG(X) Logarithm,
- EXP(X) Exponential,
- SIN(X) Trigonometric sine,
- COS(X) Trigonometric cosine,
- TAN(X) Trigonometric tangent,
- ASIN(X) Arc sine,
- ABS(X) Absolute value,

MAX(X,Y) Maximum of two values,

MIN(X,Y) Minimum of two values.

- *Random number generators:*

RANF() Random number, uniformly distributed in [0,1],

GAUSS() Random number, Gaussian distribution with unit standard deviation,

TGAUSS(X) Random number, Gaussian distribution with unit standard deviation, truncated at X standard deviations,

USER0() Random number, user-defined distribution without arguments,

USER1(X) Random number, user-defined distribution with one argument,

USER2(X,Y) Random number, user-defined distribution with two arguments.

Parentheses indicate operator precedence if required. Constant sub-expressions are evaluated immediately, and the result is stored as a constant. Exponentiation is not directly available. However, it may be performed by the identity $A^B = \exp(B * \log(A))$.

In an ordinary expression a random generator is only permitted if it has no variable arguments. A random number is then generated at definition time and stored as a constant. Example:

```
D:DRIFT,L=0.01*RANF()      ! a drift space with random length,
                             ! generated at definition time.
E:DRIFT,L=USER1(X)         ! not permitted, if X is a variable
```

2.4.7 Operands in Arithmetic Expressions

An expression may contain the following operands:

- *Literal constants:* Numerical values are entered like FORTRAN constants. Real values are accepted in INTEGER or REAL format. The use of a decimal exponent, marked by the letter D or E, is permitted. Examples:

1, 10.35, 5E3, 314.1592E-2

- *Symbolic constants:* MAD recognises the mathematical and physical constants listed in Table 2.2. Their names should not be used for user-defined labels.

Additional symbolic constants may be defined to simplify their repeated use in statements and expressions. The CONSTANT command

```
label: CONSTANT=constant-expression
```

defines a constant with the name label. Label must be unique. An existing symbolic constant can be redefined, but it cannot change in a matching procedure. A reference to a constant is immediately replaced by its value. Thus redefinition of a constant does not affect any preceding definitions. Example:

```
IN: CONSTANT=0.0254
```

- *Parameter labels:* Often a set of numerical values depends on a common variable parameter. Such a parameter must be defined according to Section 2.8.1. When its name is used in an expression, MAD uses the current value of the parameter. Example:

Table 2.2: Symbolic Constants

symbol	name	Value used 0	unit
π	PI	3.1415926535898	1
2π	TWOPI	6.2831853071796	1
$180/\pi$	DEGRAD	57.295779513082	$^\circ/\text{rad}$
$\pi/180$	RADDEG	0.017453292519943	$\text{rad}/^\circ$
e	E	2.7182818284590	1
m_e	EMASS	$0.51099906 \cdot 10^{-3}$	GeV
m_p	PMASS	0.93827231	GeV
c	CLIGHT	$2.99792458 \cdot 10^8$	m/s

```

X:=1.0
D1:    DRIFT,L=X
D2:    DRIFT,L=2.0-X

```

When the value of X is changed, the lengths of the drift spaces are recalculated as X and 2-X respectively.

- *Element or command attributes:* In arithmetic expressions the attributes of physical elements or commands can occur as operands. They are named respectively by

```

element-name[attribute-name]
command-name[attribute-name]

```

Values are assigned to attributes in element definitions or commands. Example:

```

D1:    DRIFT,L=1.0
D2:    DRIFT,L=2.0-D1[L]

```

D1[L] refers to the length L of the drift space D1.

2.4.8 Deferred Expressions and Random Values

The definition of random machine imperfections requires evaluation of expressions containing random functions. These are not evaluated like other expressions before a command begins execution, but sampled as required from the distributions indicated when errors are generated. Such an expression is known as a *deferred expression*. Its value cannot occur as an operand in another expression. Example:

```
ERROR:  EALIGN,range,DX=SIGMA*GAUSS()
```

All elements in range are assigned independent random displacements sampled from a Gaussian distribution with standard deviation SIGMA. The quantity ERROR[DX] must not occur as an operand in another expression.

2.4.9 Constraints

In matching it is desired to specify equality constraints, as well as lower and upper limits for a quantity. MAD accepts the following forms of constraints:

```

name=expression          ! equality constraint
name<expression          ! upper limit
name>expression          ! lower limit
name<expression,name>expression ! both upper and lower limit
                           ! for the same name

```

2.4.10 Variable Names

A variable name can have one of the formats:

```
parameter-name
command-name[attribute-name]
element-name[attribute-name]
```

The first format refers to the value of the global parameter `parameter-name` (see Section 2.8.1), the second format refers to the attribute `attribute-name` of the command `command-name` or element `element-name` respectively.

2.5 TITLE Statement

The TITLE statement

```
TITLE,S=page-header
```

expects a string `page-header`, enclosed in single or double quotes, which will be used as a title for subsequent output pages. Before a TITLE statement is encountered, the page header is blank. It can be redefined at any time.

2.6 STOP Statement

The STOP statement

```
STOP
```

terminates execution of the program. Any statement following the STOP statement is ignored.

2.7 OPTION Statement

The OPTIONS command (former SETOPTS statement) controls global command execution:

```
OPTION, RESET, INTER, ECHO, TRACE, DOUBLE, VERIFY, WARN, DEBUG, &
      INFO, SYMPL, &
      KEYWORD=integer, COMMAND=integer, DEFINE=integer, &
      EXPRESS=integer, LINE=integer, TELL
```

The first seven logical flags activate or deactivate execution options:

RESET If true, all options are reset to their default values listed below before setting.

INTER Normally MAD determines itself whether it runs interactively. On some computer systems this may not be possible; the user may then set this flag manually.

ECHO Controls printing of an echo of input lines on the file `ECHO`.

TRACE When the TRACE option is on, MAD writes additional information on the `ECHO` file for each executable command. This information includes the command name, elapsed CPU time before and after the command, and the CPU time used for the command.

DOUBLE When a table is created and the DOUBLE option is on, the table will be built in double precision. Otherwise the table will be built in single precision. This option has no effect in the single-precision version of MAD.

VERIFY If this option is on, MAD gives a message for each undefined variable.

WARN If this option is turned off, MAD suppresses all warning messages.

INFO If this option is turned off, MAD suppresses all information messages.

SYMPL If this option is turned off, MAD suppresses matrix symplectification.

DEBUG This option is reserved for the programmer.

The next five integer flags control debugging output. Each of them can have four different values:

0 No output.

1 Output in "comprehensive" format.

2 Output by the ZEBRA dump routines.

3 Output in both formats.

The relevant output flags are:

KEYWORD Dump every new keyword definition.

COMMAND Dump every executable command before execution.

DEFINE Dump every new element or parameter definition.

EXPRESS Dump arithmetic expressions when they are decoded or evaluated.

LINE Dump every each new line definition.

One attribute controls the closed orbit finder:

COFACT The former parameter COFACT is now ignored.

The last attribute requests listing of the current settings:

TELL If true, the current settings are listed.

Example:

```
OPTION, -ECHO, TELL
```

Turns off the command echo print-out and lists the current settings. If contradicting options are entered, the last option given prevails. Example:

```
OPTION, ECHO, -ECHO
```

turns off the ECHO option. The default settings at program start-up time are listed in Table 2.3.

Table 2.3: Initial Defaults for command options

option	setting	option	setting	option	setting	option	setting
RESET	off	INTER	on	ECHO	on	TRACE	off
DOUBLE	off	VERIFY	on	WARN	on	DEBUG	off
INFO	off	SYMPL	on				
KEYWORD	0	COMMAND	0	DEFINE	0	EXPRESS	0
LINE	0	TELL	off				

2.8 Parameter Statements

2.8.1 Relations between Variable Parameters

A relation is established between variables by the statement

```
parameter-name:=expression
```

It creates a new parameter `parameter-name` and discards any old parameter with the same name. Its value depends on all quantities occurring in `expression` on the right-hand side. Whenever an operand in `expression` changes, a new value is calculated. The definition may be thought of as a mathematical equation; but MAD is not able to solve the equation for a quantity on the right-hand side. Example:

```
GEV:=100
BEAM,ENERGY=GEV
```

Circular definitions are not allowed (but see Section 2.8.2):

```
X:=X+1      ! X cannot be equal to X+1
A:=B
B:=A        ! A and B are equal, but of unknown value
```

2.8.2 Assignment to Parameters

A value is assigned to a parameter by the SET statement

```
SET,VARIABLE=parameter-name,VALUE=expression
```

This statement acts like a FORTRAN assignment. If the parameter `parameter-name` does not yet exist, it is created. Then the expression is evaluated, and the result is assigned to the parameter `parameter-name`. Finally the expression is discarded. Therefore a sequence like the following is permitted:

```
...                ! some definitions
USE,line
X:=0                ! create parameter X with value zero
                    ! could also use "SET,X,0"
DO,TIMES=10         ! repeat ten times
  TWISS             ! uses X=0, 0.01, ..., 0.10
  SET,X,X+.01       ! increment parameter X by 0.01
ENDDO
```

2.8.3 Output of Parameters

The VALUE statement

```
VALUE,VALUE=expression{,expression}
```

evaluates up to five expressions using the most recent values of any operands and prints the results on the ECHO file. Example:

```
P1:=5
P2:=7
VALUE,P1*P2-3
```

After echoing the command, this prints:

```
AAVALU. Value of expression "P1*P2-3" is      32.00000000
```

The main use of this commands is for printing a quantity which depends on matched attributes. It allows use of MAD as a programmable calculator. One may also tabulate functions.

2.9 Commands to Manipulate Global MAD Variables

The information in this section is valid as from Version 8.19 of MAD. All global variables can be used in any mathematical expression or saved in a special MAD table (to be further archived or plotted) by means of the command PUSH. The syntax of this command is

```
PUSH,NAME{,NAME}
```

where NAME is the name of any global variable. Up to 100 variables can be simultaneously PUSHed on a special stack: this command is intended for repetitive use within a D0 loop and can be iterated up to 1000 times. The first call to PUSH fixes the number of columns and the corresponding headings of a MAD table. Subsequent calls to PUSH may contain different variables (or expressions) which will be saved under the same headings, in the order in which they appear. Their number should not exceed the initial number of columns in the table and the remaining columns, if any, are filled with zeroes. The stack is then closed by the new command ENDPUSH, which creates a MAD table called SPECIAL. The syntax of this command is

```
ENDPUSH, [SAVE=string]
```

If the optional attribute SAVE is selected, the table is also saved in a file (with default name SPECIAL). After calling ENDPUSH, the stack is emptied and it is possible to start a new series of calls to PUSH. Using this mechanism, the user can keep track of interesting results obtained during MAD calculations for different machine parameters or generate special correlation plots by simple D0 loops, possibly invoked by MAD SUBROUTINES. Examples of such plots are shown in Figs. 2.1, 2.2 and 2.3, which refer to LHC, while applications to LEP can be found in Refs. [32, 28].

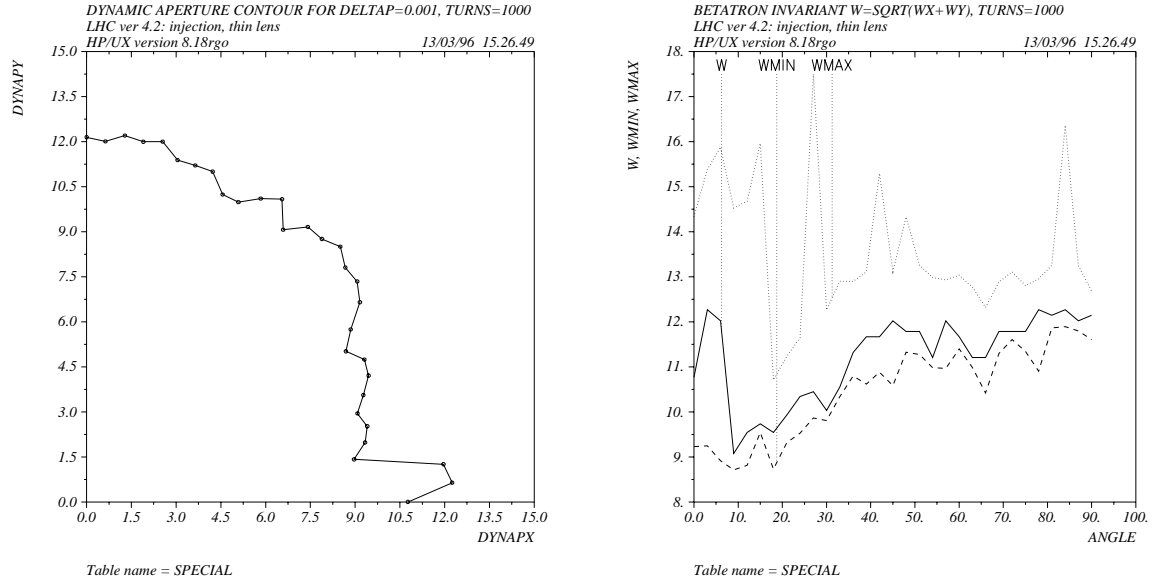


Figure 2.1: Contour plot of the dynamic aperture in the $x-y$ plane (left) and corresponding linear betatron invariant versus the $x-y$ angle in degrees (right), both expressed in terms of the r.m.s. beam size, for LHC version 4.2 at injection, with systematic and random magnetic imperfections: tracking is performed over 1000 turns, including synchrotron oscillations with $\Delta p/p = 10^{-3}$ (see the sample file `dynapcont.mad`). Note that the conventional definition of dynamic aperture may be rather sensitive to the choice of the initial phases (usually set to zero). As shown in the picture on the right, the result is comprised between the minimum and maximum value of the betatron invariant over the tracking interval, related to the smear.

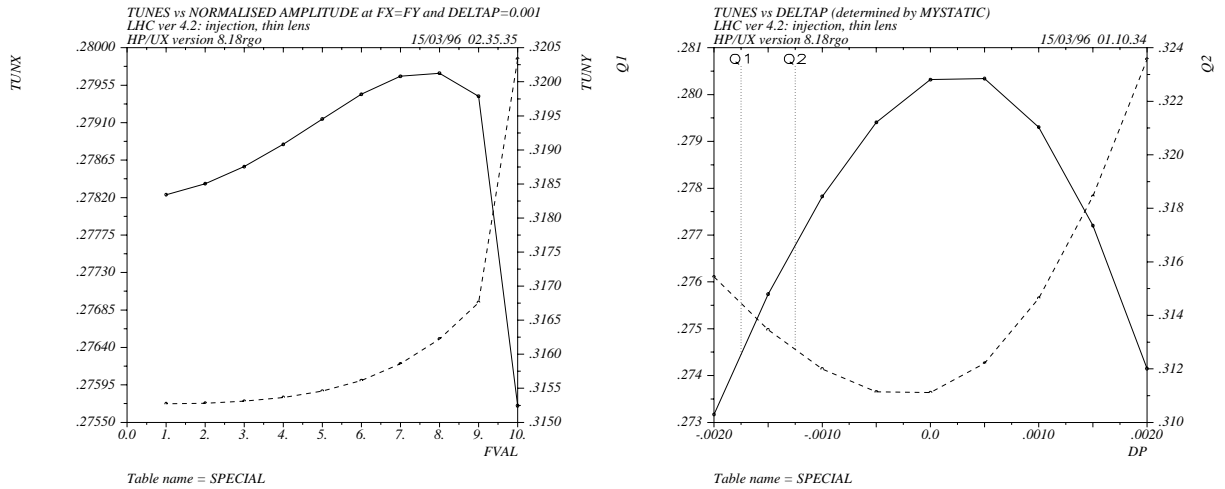


Figure 2.2: Tunes versus normalised amplitude at constant $\Delta p/p$ (left) and chromatic detunings (right) for LHC version 4.2 at injection, with systematic and random magnetic imperfections (see the sample file `detuning.mad`).

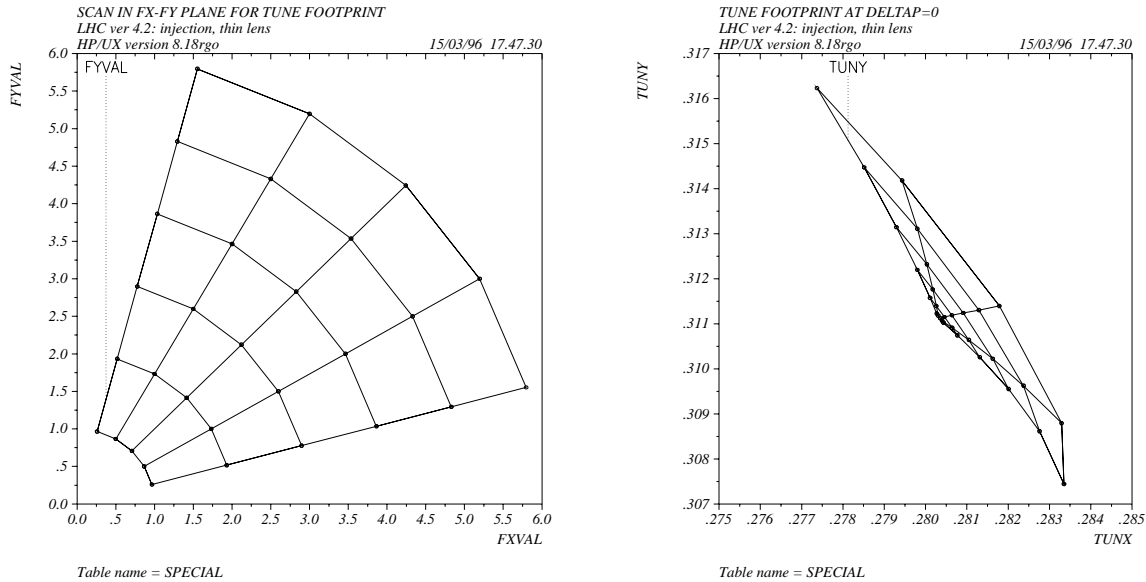


Figure 2.3: Tune footprint at constant $\Delta p/p$ (right) obtained by tracking with the fasttune option and corresponding to a prescribed scan in the $x - y$ plane (left), for LHC version 4.2 at injection, with systematic and random magnetic imperfections (see the sample file footprint.mad).

2.10 SYSTEM: Execute Operating System Command

During an interactive MAD session the command SYSTEM allows to execute operating system commands. After execution of the system command, successful or not, control returns to MAD. At present this command is only available under UNIX or VM/CMS. Its format is:

```
SYSTEM,"string"
```

where string is a valid operating system command.

2.10.1 SYSTEM Command under UNIX

Most UNIX commands can be issued directly. Example:

```
SYSTEM,"ls -l"
```

causes a listing of the current directory in long form on the terminal.

2.10.2 SYSTEM Command under VM/CMS

If the system command refers to an EXEC file the string must begin with the word EXEC. Synonyms or abbreviations are not always accepted. It is recommended to use standard CMS command names and to spell them out in full. Examples:

```
SYSTEM,"ERASE TESTDATA MAD A"
SYSTEM,"EXEC FILELIST * * D"
SYSTEM,"XEDIT TESTDATA MAD A"
```

2.11 COMMENT/ENDCOMMENT Statements

Commands bracketed between COMMENT and ENDCOMMENT are not executed, but skipped. These commands may be nested. Example:

```
EMIT      ! executed
COMMENT
SURVEY    ! not executed (nesting level 1)
COMMENT
SURVEY    ! not executed (nesting level 2)
ENDCOMMENT
TWISS     ! still not executed (nesting level 1)
ENDCOMMENT
TWISS     ! executed again
```

Chapter 3

Physical Elements and Markers

Table 3.1: Element Definition Commands

Name	Meaning	Section
MARKER	Marker for beam observation	3.2
DRIFT	Drift space	3.3
SBEND	Sector bending magnet	3.4
RBEND	Rectangular bending magnet	3.4
QUADRUPOLE	Quadrupole	3.5
SEXTUPOLE	Sextupole	3.6
OCTUPOLE	Octupole	3.7
MULTIPOLE	Thin multipole	3.8
SOLENOID	Solenoid	3.9
HKICKER	Horizontal orbit corrector	3.10
VKICKER	Vertical orbit corrector	3.10
KICKER	Corrector for both planes	3.10
RFCAVITY	RF cavity	3.11
ELSEPARATOR	electrostatic separator	3.12
HMONITOR	Horizontal orbit position monitor	3.13
VMONITOR	Vertical orbit position monitor	3.13
MONITOR	Orbit position monitor (both planes)	3.13
INSTRUMENT	Space for beam instrumentation	3.13
ECOLLIMATOR	Elliptic collimator	3.14
RCOLLIMATOR	Rectangular collimator	3.14
YROT	Rotation about vertical axis	3.15
SROT	Rotation about longitudinal axis	3.15
BEAMBEAM	Beam-beam interaction	3.16
MATRIX	Arbitrary matrix	3.17
LUMP	Concatenation of elements	3.20

3.1 Input Format

All physical elements are defined by statements of the form

```
label: keyword [,TYPE=name] {,attribute}
```

Example:

```
QF: QUADRUPOLE,TYPE=MQ,L=1.8,K1=0.015832
```

where

label A name to be given to the element (in the example QF),

keyword An element type keyword (in the example QUADRUPOLE),

TYPE A label to be attached to the element. It denotes the “engineering type” as defined document-label in earlier versions of MAD, and may be used for selection of elements in various commands like error definitions. (in the example MQ).

attribute Normally has the form

attribute-name=attribute-value

Attribute-name selects the attribute, as defined for the element type keyword (in the example L and K1), and attribute-value gives it a value (in the example 1.8 and 0.015832).

Omitted attributes are assigned a default value, normally zero. For some attributes it is permitted to enter their name only. In this case the attributes are assigned a special default value (Example: TILT angles for magnets). If such usage is allowed, it is indicated with the respective attribute.

3.2 Marker Definitions

label: MARKER,TYPE=name

The simplest element which can occur in a beam line is the MARKER. It has no effect on the beam, but it allows one to identify a position in the beam line, for example to apply a matching constraint. A marker has only the TYPE attribute: Example:

M27: MARKER,TYPE=MM}

3.3 Drift Space

label: DRIFT,TYPE=name,L=real

A DRIFT space has one real attribute:

L The drift length (default: 0 m)

Examples:

DR1: DRIFT,L=1.5

DR2: DRIFT,L=DR1[L],TYPE=DRF

The length of DR2 will always be equal to the length of DR1. The reference system for a drift space is shown in Figure 3.1.

3.4 Bending Magnets

Two different type keywords are recognised for bending magnets, they are distinguished only by the reference system used:

RBEND Rectangular bending magnet (Reference see Figure 3.2),

SBEND Sector bending magnet (Reference see Figure 3.3).

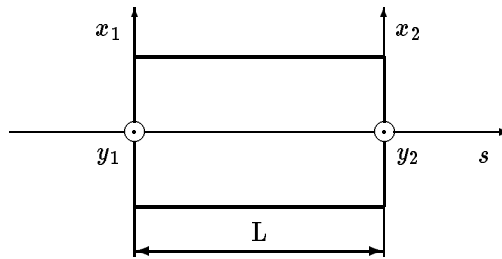


Figure 3.1: Reference System for Straight Beam Elements

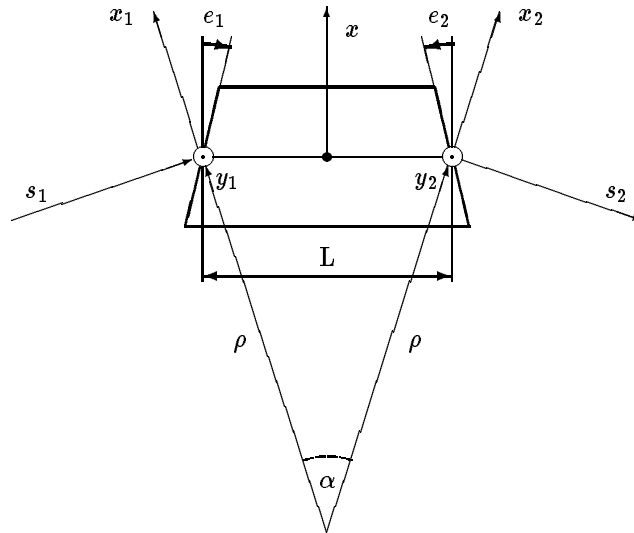


Figure 3.2: Reference System for a Rectangular Bending Magnet; the signs of pole-face rotations are positive as shown.

```

SBEND, TYPE=name, L=real, ANGLE=real, K1=real, E1=real, E2=real, &
      TILT=real, K2=real, H1=real, K2=real, HGAP=real, FINT=real, &
      K3=real
RBEND, TYPE=name, L=real, ANGLE=real, K1=real, E1=real, E2=real, &
      TILT=real, K2=real, H1=real, K2=real, HGAP=real, FINT=real, &
      K3=real

```

For both types, the following first-order attributes are permitted:

L The length of the magnet (default: 0 m). For a rectangular magnet the length is measured along a straight line, while for a sector magnet it is the arc length of the reference orbit.

ANGLE The bend angle (default: 0 rad). A positive bend angle represents a bend to the right, i.e. towards negative x values.

K1 The quadrupole coefficient $K_1 = (1/B\rho)(\partial B_y/\partial x)$. The default is 0 m^{-2} . A positive quadrupole strength implies horizontal focussing of positively charged particles.

E1 The rotation angle for the entrance pole face (default: 0 rad).

E2 The rotation angle for the exit pole face (default: 0 rad).

FINT The field integral, see [8] and below. The default value is 0.

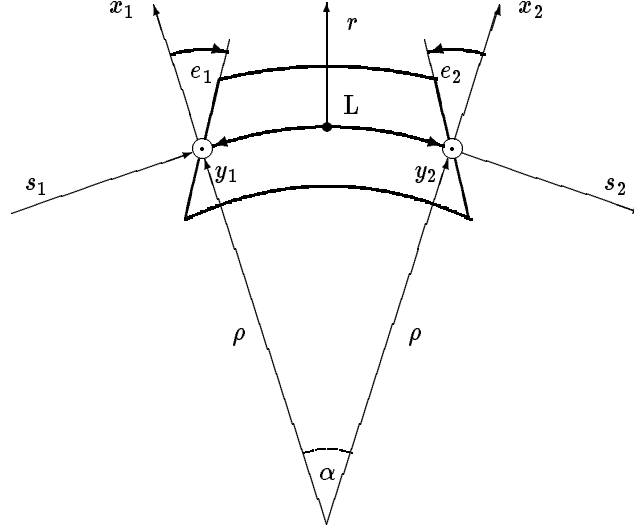


Figure 3.3: Reference System for a Sector Bending Magnet; the signs of pole-face rotations are positive as shown.

HGAP The half gap of the magnet (default: 0 m).

The pole face rotation angles are referred to the magnet model (see Figure 3.2 and Figure 3.3). The quantities FINT and HGAP specify the finite extent of the fringe fields [8]. There they are defined as follows:

$$K_1 = \text{FINT} = \int_{-\infty}^{\infty} \frac{B_y(s)(B_0 - B_y(s))}{g \cdot B_0^2} ds, \quad g = 2 \cdot \text{HGAP}.$$

The default values of zero corresponds to the hard-edge approximation, i.e. a rectangular field distribution. For other approximations, enter the correct value of the half gap, and one of the following values for FINT:

Linear drop-off of the field	1/6
Clamped “Rogowski” fringing field	0.4
Un-clamped “Rogowski” fringing field	0.7
“Square-edged” non-saturating magnet	0.45

Entering the keyword FINT alone sets the integral to 0.5. This is a reasonable average of the above values. The magnet may be rotated about the longitudinal axis by use of the following parameter:

TILT The roll angle about the longitudinal axis (default: 0 rad, a positive bend angle then denotes a bend to the right). A vertical bend is defined by entering TILT with no value; this implies a roll of $\pi/2$ rad, i.e. a positive bend angle denotes a deflection down. A positive angle represents a clockwise rotation. The following second-order attributes are permitted:

K2 The sextupole coefficient $K_2 = (1/B\rho)(\partial^2 B_y)/(\partial x^2)$ (default: 0 m⁻³).

H1 The curvature of the entrance pole face (default: 0 m⁻¹).

H2 The curvature of the exit pole face (default: 0 m⁻¹). A positive pole face curvature induces a negative sextupole component; i.e. for positive H1 and H2 the centres of curvature of the pole faces are placed inside the magnet.

One third-order parameter is accepted, but at present it is ignored:

K3 The octupole coefficient $K_3 = (1/B\rho)(\partial^3 B_y/\partial x^3)$ (default: 0 m⁻⁴).

Examples:

```
BR: RBEND,L=5.5,ANGLE=+0.001           ! Deflection to the right
BD: SBEND,L=5.5,ANGLE=+0.001,TILT      ! Deflection down
BL: SBEND,L=5.5,ANGLE=-0.001           ! Deflection to the left
BU: SBEND,L=5.5,ANGLE=-0.001,TILT      ! Deflection up
```

3.5 Quadrupole

label: QUADRUPOLE,TYPE=name,L=real,K1=real,TILT=real

A QUADRUPOLE has three real attributes:

L The quadrupole length (default: 0 m).

K1 The quadrupole coefficient $K_1 = (1/B\rho)(\partial B_y/\partial x)$. The default is 0 m⁻². A positive quadrupole strength implies horizontal focussing of positively charged particles.

TILT The roll angle about the longitudinal axis (default: 0 rad, i. e. a normal quadrupole). A skewed quadrupole is defined by entering TILT with no value; this implies a roll of $\pi/4$ rad about the *s*-axis. A positive angle represents a clockwise rotation.

Example:

```
QF: QUADRUPOLE,L=1.5,K1=0.001
```

The reference system for a quadrupole is shown in Figure 3.1.

3.6 Sextupole

label: SEXTUPOLE,TYPE=name,L=real,K2=real,TILT=real

A SEXTUPOLE has three real attributes:

L The sextupole length (default: 0 m).

K2 The sextupole coefficient $K_2 = (1/B\rho)(\partial^2 B_y/\partial x^2)$ (default: 0 m⁻³).

TILT The roll angle about the longitudinal axis (default: 0 rad, i. e. a normal sextupole). A skewed sextupole is defined by entering TILT with no value; this implies a roll of $\pi/6$ rad about the *s*-axis. A positive angle represents a clockwise rotation.

Example:

```
S: SEXTUPOLE,L=0.4,K2=0.00134
```

The reference system for a sextupole is shown in Figure 3.1.

3.7 Octupole

label: OCTUPOLE,TYPE=name,L=real,K3=real,TILT=real

An OCTUPOLE has three real attributes:

L The octupole length (default: 0 m).

K3 The octupole coefficient $K_3 = (1/B\rho)(\partial^3 B_y/\partial x^3)$ (default: 0 m⁻⁴).

TILT The roll angle about the longitudinal axis (default: 0 rad, i. e. a normal octupole). A skewed octupole is defined by entering TILT with no value; this implies a roll of $\pi/8$ rad about the s -axis. A positive angle represents a clockwise rotation.

Example:

03: OCTUPOLE,L=0.3,K3=0.543

The reference system for a octupole is shown in Figure 3.1. Octupoles are normally treated as thin lenses, except when tracking by Lie-algebraic methods.

3.8 General Thin Multipole

label: MULTIPOLE,TYPE=name,LRAD=real,&
 K0L=real,T0=real,K1L=real,T1=real,&
 K2L=real,T2=real,K3L=real,T3=real,&
 K4L=real,T4=real,K5L=real,T5=real,&
 K6L=real,T6=real,K7L=real,T7=real,&
 K8L=real,T8=real,K9L=real,T9=real

A MULTIPOLE is thin lens of arbitrary order, including a dipole:

LRAD A fictitious length, which is only used to compute synchrotron radiation effects.

KnL The multipole coefficient of order n : $K_n L = (L/B\rho)(\partial^n B_y/\partial x^n)$. (default: 0 m⁻ⁿ). This is the multipole coefficient integrated over the length of the multipole. The digit n may take the values $0 \leq n \leq 9$. The number of poles of the component is $2n + 2$. The most important error components of quadrupoles are: K5L, the twelve-pole, and K9L, the twenty-pole. Superposition of several multipole components is permitted.

Tn The roll angle for the multipole component of order n about the s -axis (default: 0 rad, i. e. a normal multipole). A skewed multipole is defined by entering Tn with no value; this implies a roll of $\pi/(2n + 2)$ rad about the s -axis. A positive angle represents a clockwise rotation.

The multipole element has no length. Example:

M27: MULTIPOLE,K3L=0.0001,T3,K2L=0.0001

A multipole with no dipole component has no effect on the reference orbit, i.e. the reference system at its exit is the same as at its entrance. If it includes a dipole component, it has the same effect on the reference orbit as a dipole with zero length and deflection angle K0L.

3.9 Solenoid

label: SOLENOID, TYPE=name, L=real, KS=real

A SOLENOID has two real attributes:

L The length of the solenoid (default: 0 m)

KS The solenoid strength $B_0/B\rho$ (default: 0 rad/m). For positive KS and positive particle charge, the solenoid field points in the direction of increasing s .

Example:

SOLO: SOLENOID, L=2., K=0.001

The reference system for a solenoid is shown in Figure 3.1.

3.10 Closed Orbit Correctors

Three types of closed orbit correctors are available:

HKICKER A corrector for the horizontal plane,

VKICKER A corrector for the vertical plane,

KICKER A corrector for both planes.

label: HKICKER, TYPE=name, L=real, KICK=real, TILT=real

label: VKICKER, TYPE=name, L=real, KICK=real, TILT=real

label: KICKER, TYPE=name, L=real, HKICK=real, VKICK=real, TILT=real

They have the following attributes:

L The length of the closed orbit corrector (default: 0 m).

KICK The kick angle for either horizontal or vertical correctors. (default: 0 rad).

HKICK The horizontal kick angle for a corrector in both planes (default: 0 rad).

VKICK The vertical kick angle for a corrector in both planes (default: 0 rad).

TILT The roll angle about the longitudinal axis (default: 0 rad). A positive angle represents a clockwise rotation of the corrector.

A positive kick increases p_x or p_y respectively. Examples:

HK1: HKICKER, KICK=0.001, TILT=RADDEG*10.0

VK3: VKICKER, KICK=0.0005

KHV: KICKER, HKICK=0.001, VKICK=0.0005

The first kicker is rotated about the longitudinal axis by 10 degrees. The reference system for a closed orbit corrector is shown in Figure 3.1.

3.11 RF Cavity

```
label: RFCAVITY,TYPE=name,L=real,VOLT=real,LAG=real,&
      HARMON=integer,BETRF=real,PG=real,&
      SHUNT=real,TFILL=real
```

An RFCAVITY has seven real attributes and one integer attribute:

L The length of the cavity (default: 0 m)

VOLT The peak RF voltage (default: 0 MV). The effect of the cavity is

$$\delta E = \text{VOLT} \sin(2\pi(\text{LAG} - hf_0\Delta t)).$$

LAG The phase lag in multiples of 2π (default: 0).

HARMON The harmonic number h (no default). Note that the RF frequency is computed from the harmonic number and the revolution frequency f_0 . *The frequency attribute FREQ must no longer be used.*

BETRF RF coupling factor (default: 0).

PG The RF power per cavity (default: 0 MW).

SHUNT The relative shunt impedance (default: 0 M Ω /m).

TFILL The filling time of the cavity (default: 0 μ s).

A cavity requires the particle energy (ENERGY) and the particle charge (CHARGE) to be set by a BEAM command (see Section 5.3) before any calculations are performed. Example:

```
BEAM,PARTICLE=ELECTRON,ENERGY=50.0
CAVITY: RFCAVITY,L=10.0,VOLT=150.0,LAG=0.0,HARMON=31320
```

The reference system for a cavity is shown in Figure 3.1.

3.12 Electrostatic Separator

```
label: ELSEPARATOR,TYPE=name,L=real,E=real,TILT=real
```

An ELSEPARATOR (electrostatic separator) has three real attributes:

L The length of the separator (default: 0 m).

E The electric field strength (default: 0 MV/m). A positive field increases p_y for positive particles.

TILT The roll angle about the longitudinal axis (default: 0 rad). A positive angle represents a clockwise rotation of the separator.

A separator requires the particle energy (ENERGY) and the particle charge (CHARGE) to be set by a BEAM command (see Section 5.3) before any calculations are performed. Example:

```
BEAM,PARTICLE=POSITRON,ENERGY=50.0
SEP: ELSEPARATOR,L=5.0,E=0.5
```

The reference system for a separator is shown in Figure 3.1.

3.13 Beam Position Monitor

A beam monitor acts on the beam like a drift space. In addition it serves to record the beam position for closed orbit corrections. Four different types of beam position monitors are recognised:

HMONITOR Monitor for the horizontal beam position,

VMONITOR Monitor for the vertical beam position,

MONITOR Monitor for both horizontal and vertical beam position.

INSTRUMENT A place holder for any type of beam instrumentation. Optically it behaves like a drift space; it returns *no beam observation*. It represent a class of elements which is completely independent from drifts and monitors.

```
label: HMONITOR,    TYPE=name,L=real
label: VMONITOR,    TYPE=name,L=real
label: MONITOR,     TYPE=name,L=real
label: INSTRUMENT,  TYPE=name,L=real
```

A beam position monitor has one real attribute:

L The length of the monitor (default: 0 m). If the length is different from zero, the beam position is recorded in the centre of the monitor.

Examples:

```
MH: HMONITOR,L=1
MV: VMONITOR
```

The reference system for a monitor is shown in Figure 3.1.

3.14 Collimators (Aperture Definitions)

Two types of collimators are defined:

ECOLLIMATOR Elliptic aperture,

RCOLLIMATOR Rectangular aperture.

```
label: ECOLLIMATOR,TYPE=name,L=real,XSIZE=real,YSIZE=real
label: RCOLLIMATOR,TYPE=name,L=real,XSIZE=real,YSIZE=real
```

Either type has three real attributes:

L The collimator length (default: 0 m).

XSIZE The horizontal half-aperture (default: unlimited).

YSIZE The vertical half-aperture (default: unlimited).

For elliptic apertures, XSIZE and YSIZE denote the half-axes respectively, for rectangular apertures they denote the half-width of the rectangle. Optically a collimator behaves like a drift space, but during tracking, it also introduces an aperture limit. The aperture is checked at the entrance. If the length is not zero, the aperture is also checked at the exit. The reference system for a collimator is shown in Figure 3.1. Example:

```
COLLIM: ECOLLIMATOR,L=0.5,XSIZE=0.01,YSIZE=0.005
```

3.15 Coordinate Transformations

3.15.1 Rotation About the Vertical Axis

label: YROT, TYPE=name, ANGLE=real

The element YROT rotates the reference system about the vertical (y) axis. The reference system is shown in Figure 3.4. YROT has no effect on the beam, but it causes the beam to be referred to the new coordinate system

$$x_2 = x_1 \cos \theta - s_1 \sin \theta, \quad s_2 = x_1 \sin \theta + s_1 \cos \theta.$$

It has one real attribute:

ANGLE The rotation angle θ (default: 0 rad). It must be a *small* angle, i.e. an angle comparable to the transverse angles of the orbit.

A positive angle means that the new reference system is rotated clockwise about the local y -axis with respect to the old system. Example:

KINK: YROT, ANGLE=0.0001

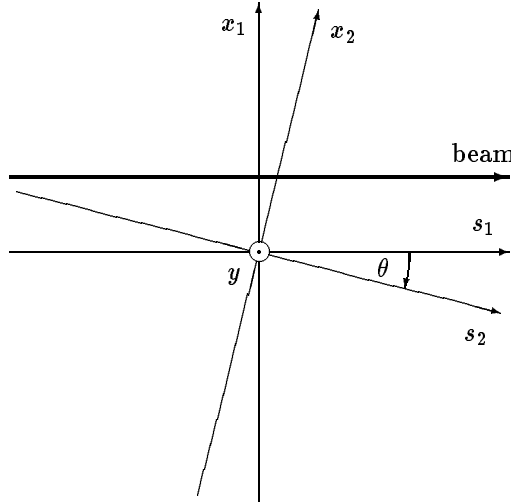


Figure 3.4: Reference System for a Rotation Around the y -Axis

3.15.2 Rotation Around the Longitudinal Axis

label: SROT, TYPE=name, ANGLE=real

The element SROT rotates the reference system about the longitudinal (s) axis. The reference system is shown in Figure 3.5. SROT has no effect on the beam, but it causes the beam to be referred to the new coordinate system

$$x_2 = x_1 \cos \psi + y_1 \sin \psi, \quad y_2 = x_1 \sin \psi + y_1 \cos \psi.$$

It has one real attribute:

ANGLE The rotation angle ψ (default: 0 rad)

A positive angle means that the new reference system is rotated clockwise about the s -axis with respect to the old system. Example:

```

ROLL1: SROT,ANGLE=PI/2.
ROLL2: SROT,ANGLE=-PI/2.
HBEND: SBEND,L=6.0,ANGLE=0.01
VBEND: LINE=(ROLL1,HBEND,ROLL2)

```

The above is a way to represent a bend down in the vertical plane, it could be defined more simply by

```
VBEND: SBEND,L=6.0,ANGLE=0.01,TILT
```

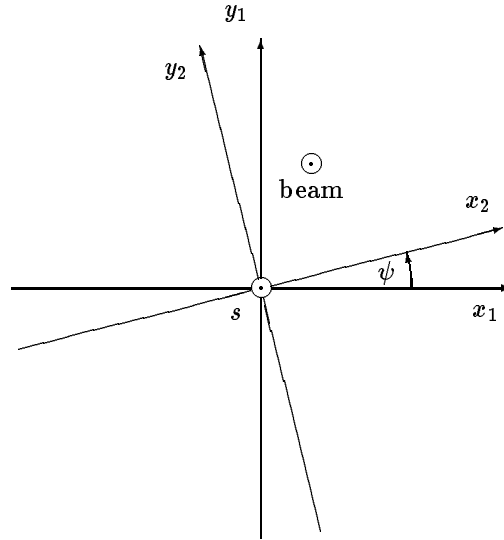


Figure 3.5: Reference System for a Rotation Around the s -Axis

3.16 BEAMBEAM Element

```

label: BEAMBEAM, TYPE=name,SIGX=real,SIGY=real,XMA=real,YMA=real,&
      CHARGE=real

```

The command BEAMBEAM may be inserted in a beam line to simulate a beam-beam interaction point.¹ It has five real attributes:

SIGX The horizontal extent (standard deviation) of the opposite beam (default: 0 m).

SIGY The vertical extent (standard deviation) of the opposite beam (default: 0 m).

XMA The horizontal displacement of the opposite beam with respect to the ideal orbit (default: 0 m).

YMA The vertical displacement of the opposite beam with respect to the ideal orbit (default: 0 m).

CHARGE The charge of particles in the opposite beam in proton charges (default: 0). Entering the keyword CHARGE alone sets the charge to -1 .

A beam-beam interaction requires the particle energy (ENERGY) and the charge of the particles in the beam considered (CHARGE), as well as the number of particles per bunch (NPART) to be set by a BEAM command (see Section 5.3) before any calculations are performed. Example:

```

BEAM, PARTICLE=POSITRON,NPART=1.E12,ENERGY=50.0
BB:   BEAMBEAM,SIGX=1.E-3,SIGY=5.E-4,CHARGE=1.

```

¹Contributed by J. M. Veuillen (1987).

3.17 Arbitrary MATRIX Element

```
MATRIX, TYPE=name, RM(1,1)=real, ..., RM(6,6)=real, &
      TM(1,1,1)=real, ..., TM(6,6,6)=real
```

The MATRIX permits definition of an arbitrary transfer matrix. It has two real array attributes:

RM(*i*,*k*) Defines the element (*i*,*k*) of the linear transfer matrix.

TM(*i*,*k*,*l*) Defines the element (*i*,*k*,*l*) of the second-order TRANSPORT matrix.

Matrix elements not entered are taken from the identity transformation, i.e.

$$\text{RM}(i, k) = \begin{cases} 1, & \text{if } i = k, \\ 0, & \text{if } i \neq k, \end{cases} \quad \text{TM}(i, k, l) = 0.$$

3.18 Editing Element Definitions

The EDIT command of former MAD versions is no longer available, but an element definition can be changed in two ways:

- *Entering a new definition:* The element will be replaced in the main beam line expansion.
- *Entering the element name together with new attributes:* The element will be updated in place, and the new attribute values will replace the old ones.

This example shows two ways to change the strength of a quadrupole:

```
QF: QUADRUPOLE, L=1, K1=0.01      ! Original definition of QF
QF: QUADRUPOLE, L=1, K1=0.02      ! Replace whole definition of QF
QF, K1=0.02                       ! Replace value of K1
```

When the type of the element remains the same, replacement of an attribute is the more efficient way.

Element definitions can be edited freely. The changes do not affect already defined objects which belong to its class (see Section 3.19). The element to be modified should not be member of a beam line sequence (see Section 4.2).

Beam elements or beam lines (see Section 4) can be replaced by other beam elements or beam lines. However, if a beam line is involved in the replacement, and the replaced item occurs in the working beam line (see Section 5.1), the working beam line becomes obsolete and is deleted automatically.

3.19 Element Classes

The concept of element classes solves the problem of addressing instances of elements in the accelerator in a convenient manner. It will first be explained by an example. All the quadrupoles in the accelerator form a class QUADRUPOLE. Let us define three subclasses for the focussing quadrupoles, the defocussing quadrupoles, and the skewed quadrupoles:

```
MQF: QUADRUPOLE, L=LQM, K1=KQD    ! Focussing quadrupoles
MQD: QUADRUPOLE, L=LQM, K1=KQF    ! Defocussing quadrupoles
MQT: QUADRUPOLE, L=LQT, TILT      ! Skewed quadrupoles
```

These classes can be thought of as new keywords which define elements with specified default attributes. We now use these classes to define the real quadrupoles:

```

QD1: MQD          ! Defocussing quadrupoles
QD2: MQD
QD3: MQD
...
QF1: MQF          ! Focussing quadrupoles
QF2: MQF
QF3: MQF
...
QT1: MQT,K1=KQT1  ! Skewed quadrupoles
QT2: MQT,K1=KQT2
...

```

These quadrupoles inherit all unspecified attributes from their class. This allows to build up a hierarchy of objects with a rather economic input structure.

The full power of the class concept is revealed when object classes are used to select instances of elements for printing. Example:

```

PRINT,QUADRUPOLE  ! Print at all quadrupoles
PRINT,MQT[1/2]    ! Print at the first two skewed quadrupoles

```

More examples appear in Section 5.4.

More formally, for each element keyword MAD maintains a class of elements with the same name. A defined element becomes itself a class which can be used to define new objects, which will become members of this class. A new object inherits all attributes from its class; but its definition may override some of those values by new ones. All class and object names can be used in range selections, providing a powerful mechanism to specify positions for matching constraints and printing.

3.20 LUMP Element

The command LUMP concatenates transfer maps. It has the form

```
label: LUMP,TYPE=name,ORDER=integer,LINE=line
```

It expands the beam line line, and evaluates its transfer map to order ORDER. The map is associated with the name label. In other words this command pre-calculates a transfer map and stores it for future use.

The ORDER parameter must lie in the range $2 \leq \text{ORDER} \leq 6$. It refers to the order of the corresponding Hamiltonian; ORDER = 2 denotes a linear transformation, ORDER = 3 includes all terms up to second order in the TRANSPORT sense.

In a TWISS command MAD considers terms up to ORDER = 3. To handle chromatic effects properly, all LUMPs used in a beam line should have ORDER ≥ 3 . However, in a TWISS command MAD ignores any terms with ORDER > 3 . MAD is not capable to find the integral part of the phase advance over a LUMP. Thus, for ORDER ≥ 2 α, β, D are correct, as well as the fractional parts of μ . If ORDER ≥ 3 , the chromatic functions are also correct.

For tracking by the TRANSPORT method MAD limits the order of tracking to three. For tracking by the Lie-algebraic method, it always uses the order of the LUMP definition. For linear elements, tracking can be speeded up considerably, if one selects ORDER = 2.

Some important points must be kept in mind about this element:

- The transfer map is evaluated when used for the first time, it is only re-evaluated when parameters of the line change.
- Any sub-lines must be defined before using the LUMP. MAD may otherwise complain about undefined sub-lines when tracking.

- The name `label` behaves like an element, it is not a beam line. The individual elements building up the LUMP cannot be misaligned nor mispowered. It is however permissible to misalign (not to mispower) the LUMP as a whole. A LUMP can also be used in a SURVEY command. It must not be used in a HARMON run.
- Concatenation of too many elements causes truncation of high-order effects. The number of concatenated elements should therefore not be too large.
- As from Version 8.2/4 of MAD transfer maps are recalculated automatically for a lump if one of the elements contained in this lump is changed. Use of a lump in matching is always safe. However lumping saves time only if the elements in a lump are not varied.

Examples for LUMP definitions:

```
SEC:  LINE=(QF,D,QD,D)
LSEC: LUMP,ORDER=4,LINE=SEC
```

Putting the line sequence directly in the LUMP has the same effect:

```
LSEC: LUMP,ORDER=4,LINE=(QF,D,QD,D)
```

In earlier versions of MAD it could save time when bending magnets were defined as lumps. From Version 8 MAD treats bending magnets as lumps automatically, it will be useless if the user attempts to do so.

Chapter 4

Beam Lines, Sequences, Lists, Ranges, Selections

Table 4.1: Line Definition Commands

Name	Meaning	Section
LINE	Beam line definition	4.1
SEQUENCE	Beam line sequence	4.2
LIST	Replacement list definition	4.6

4.1 Beam Lines

The accelerator to be studied is known to MAD as a sequence of physical elements called a *beam line*. A beam line is built from simpler beam lines whose definitions can be nested to any level. A powerful syntax allows to repeat, to reflect, or to replace pieces of beam lines. Formally a beam line is defined by a LINE command:

```
label(arg{,arg}): LINE=(member{,member})
```

Label gives a name to the beam line for later reference. The formal argument list (arg{,arg}) is optional (see below). Each member may be one of the following:

- Element label,
- Beam line label,
- Sub-line, enclosed in parentheses,
- Formal argument name,
- Replacement list label.

Beam lines may be nested to any level.

4.1.1 Simple Beam Lines

The simplest beam line consists of single elements:

```
label: LINE=(member{,member})
```

Example:

```
L:      LINE=(A,B,C,D,A,D)
        USE,L
```

The USE command tells MAD to perform all subsequent calculations on the sequence

A,B,C,D,A,D

4.1.2 Sub-lines

Instead of referring to an element, a beam line member can refer to another beam line defined in a separate command. This provides a shorthand notation for sub-lines which occur several times in a beam line. Lines and sub-lines can be entered in any order, but when a line is expanded, all its sub-lines must be known. Example:

```
L:      LINE=(A,B,S,B,A,S,A,B)
S:      LINE=(C,D,E)
        USE,L
```

This example produces the following expansion steps:

1. Replace sub-line S:

(A,B,(C,D,E),B,A,(C,D,E),A,B)

2. Omit parentheses:

A,B,C,D,E,B,A,C,D,E,A,B

4.1.3 Reflection and Repetition

An unsigned repetition count and an asterisk indicate repetition of a beam line member. A minus prefix causes reflection, i.e. all elements in the subsequence are taken in reverse order. Sub-lines of reflected lines are also reflected, but physical elements are not. If both reflection and repetition are desired, the minus sign must precede the repetition count. Example:

```
R:      LINE=(G,H)
S:      LINE=(C,R,D)
T:      LINE=(2*S,2*(E,F),-S,-(A,B))
        USE,T
```

Proceeding step by step, this example produces

1. Replace sub-line S:

((C,R,D),(C,R,D),(E,F),(E,F),(D,-R,C),(B,A))

2. Replace sub-line R:

((C,(G,H),D),(C,(G,H),D),(E,F),(E,F),(D,(H,G),C),(B,A))

3. Omit parentheses:

C,G,H,D,C,G,H,D,E,F,E,F,D,H,G,C,B,A

Note that the inner sub-line R is reflected together with the outer sub-line S.

4.1.4 Replaceable Arguments

A beam line definition may contain a formal argument list, consisting of labels separated by commas and enclosed in parentheses. Such a line can be expanded for different values of its arguments. When it is referred to, its label must be followed by a list of actual arguments separated by commas and enclosed in parentheses. Each top level member of the list represents one argument; it can contain anything which may occur as a beam line member. MAD assumes an implicit pair of parentheses around each argument. The number of actual arguments must agree with the number of formal arguments. All occurrences of a formal argument on the right-hand side of the line definition are replaced by the corresponding actual argument. Example:

```
S:      LINE=(A,B,C)
L(X,Y): LINE=(D,X,E,3*Y)
        USE,L(4*F,-2*S)
```

Proceeding step by step, this example generates the expansion

1. Replace formal arguments:

```
(D,(4*F),E,3*(-2*(A,B,C)))
```

2. Combine repetition counts for last sub-line:

```
(D,(4*F),E,-6*(A,B,C))
```

3. Expand repetitions and reflection:

```
(D,(F,F,F,F),E,(C,B,A),(C,B,A),(C,B,A),(C,B,A),(C,B,A),(C,B,A))
```

4. Omit parentheses:

```
D,F,F,F,F,E,C,B,A,C,B,A,C,B,A,C,B,A,C,B,A,C,B,A
```

Second example:

```
CEL(SF,SD):  LINE=(QF,D,SF,D,B,D,QD,D,SD,D,B,D)
ARC:         LINE=(CEL(SF1,SD1),CEL(SF2,SD2),CEL(SF1,SD1))
            USE,ARC
```

This example generates the expansion

1. Replace the line CEL and its formal arguments:

```
((QF,D,(SF1),D,B,D,QD,D,(SD1),D,B,D)
 (QF,D,(SF2),D,B,D,QD,D,(SD2),D,B,D)
 (QF,D,(SF1),D,B,D,QD,D,(SD1),D,B,D))
```

2. Omit parentheses:

```
QF,D,SF1,D,B,D,QD,D,SD1,D,B,D
QF,D,SF2,D,B,D,QD,D,SD2,D,B,D
QF,D,SF1,D,B,D,QD,D,SD1,D,B,D
```

4.2 Beam Line Sequences

When MAD runs in an accelerator control system, the sequence of elements is usually generated from a data base. For this purpose there is a command

```
label: SEQUENCE,REFER=keyword
      object: class,AT=real{,attributes}
      ...
ENDSEQUENCE
```

It reads a sequence of element definitions, compiles a data bank which resembles a beam line definition, and gives it the name `label`. The resulting sequence can be used like a beam line. For each non-drift element in the sequence one element definition appears following the `SEQUENCE` command and preceding the `ENDSEQUENCE` command. These look like the definitions described in Chapter 3, with the following differences:

- The name `object` must be unique; it must not be defined earlier in the data.
- The second name `class` on the definition must be a class name.
- The class name *must be followed immediately* by the attribute `AT=real`. This specifies the longitudinal position of the element with respect to the beginning of the sequence. The attribute `REFER` specifies the reference points for the elements:

`CENTRE` The element centre is placed at the given position (default).

`ENTRY` The element entry is placed at the given position.

`EXIT` The element exit is placed at the given position.

The elements must be entered in order of increasing position, and must not overlap.

- The elements `object` inherit the attributes from `class`. If no further attributes appear, MAD uses the data bank of `class` for the element data. In this case `class` cannot be modified subsequently. If further attributes exist, MAD makes a copy of the data for the class, as if it had read a normal element definition.
- The last object in the sequence should be a `MARKER`; its position defines the total length of the sequence.

When the sequence is expanded in a `USE` command, MAD generates the drift spaces for proper positioning.

For efficiency reasons MAD imposes an *important restriction* for variable lengths. If the lengths of elements and their positions are defined as constant values, MAD gives the intervening drifts a constant length as well, and checks that the drift lengths are all positive. In this case the drift lengths can never be varied, and a change in element length affects the total length of the system.

Variation of an element position and/or length is possible. The element length or position to be varied *must* then be defined by an expression. This causes MAD to adjust the dependent drift length(s) such as to keep all definitions consistent. However, it is the responsibility of the program user that drift spaces do not become negative. Example:

```
! Define element classes for a simple cell:
B:    SBEND,L=35.09, ANGLE = 0.011306116
QF:    QUADRUPOLE,L=1.6,K1=-0.02268553
QD:    QUADRUPOLE,L=1.6,K1=0.022683642
SF:    SEXTUPOLE,L=0.4,K2=-0.13129
```

```

SD:      SEXTUPOLE,L=0.76,K2=0.26328
! Define the cell as a sequence:
CELL:    SEQUENCE
      B1:      B,      AT=19.115
      SF1:     SF,      AT=37.42
      QF1:     QF,      AT=38.70
      B2:      B,      AT=58.255,ANGLE=B1[ANGLE]
      SD1:     SD,      AT=76.74
      QD1:     QD,      AT=78.20
      ENDM:    MARKER, AT=79.0
ENDSEQUENCE

```

In this example all members of the sequence use the storage of their class except the element B2 which uses a copy because one of its attribute has been redefined.

4.3 Editing a Beam Line Sequence

The information in this section is valid for Version 8.18 of MAD and later. Note that earlier versions used mechanisms incompatible with the one described here. A beam line sequence may be edited *before* it is used in a USE command. Seven commands are available for this purpose:

```

SEQEDIT,SEQUENCE=name
SELECT,FLAG=SEQEDIT[,RANGE=range][,CLASS=name][,PATTERN=string] &
      [,FULL=logical][,CLEAR=logical]
INSTALL,ELEMENT=name1[CLASS=name2],AT=value[,FROM=name]
MOVE,ELEMENT=name[,BY=value][,TO=value[,FROM=name]]
REFLECT
REMOVE,CLASS=name
REPLACE,ELEMENT=name1,BY=name2
CYCLE,START=name
ENDEDIT

```

They are explained in detail in the next sub-sections.

4.3.1 Making a sequence known for editing

Before modifying a sequence, its name must be set for editing by the command

```
SEQEDIT,SEQUENCE=name
```

MAD will make the sequence name the current sequence being edited. Once editing is finished, one should give the command

```
ENDEDIT
```

4.3.2 Cyclic interchange

The command

```
CYCLE,START=name}
```

makes a cyclic interchange of all elements in the current edit sequence so as to start at the element name. This element should preferably be a MARKER. All further edit commands must refer to the *new* positions.

4.3.3 Installing an Element

A new element can be installed in the edited sequence by the command

```
INSTALL,ELEMENT=name1[,CLASS=name2],AT=value[,FROM=name]
```

It has the following attributes:

ELEMENT The name of the element to be installed.

CLASS The name of the class from which the element shall be derived.

AT The position where to install the element.

FROM The name of the element to be taken as the origin. If the FROM attribute does not appear, AT implies an absolute displacement from the beginning of the sequence; if it does, the distance is relative to the position of that element. A relative position may be negative.

4.3.4 Moving an Element

The command

```
MOVE,ELEMENT=name[,BY=value][,TO=value[,FROM=name]]
```

moves an element to a new location. Its attributes are:

ELEMENT The name of the element to be moved. If this name is the literal SELECTED, all selected elements are moved, and the only further value allowed is BY.

BY The amount by which the element is to be moved. If this value is given, FROM and TO are not allowed.

TO The new position for the element.

FROM The name of the element to be taken as the origin. If the FROM attribute does not appear, TO implies an absolute displacement from the beginning of the sequence; if it does, the distance is relative to the position of that element. A relative position can be negative.

The MOVE command can change the order of elements in the sequence.

4.3.5 Reflecting a Sequence

The command

```
REFLECT
```

reverses the order of all elements in the sequence currently being edited. All further edit commands must refer to the *new* positions.

4.3.6 Removing Elements

The command

```
REMOVE,ELEMENT=name
```

removes element(s) using the value of the attribute

ELEMENT If this is the literal SELECTED, all elements selected by previous SELECT,FLAG=SEQEDIT commands are removed. Otherwise the *first* element with the name entered is removed.

Examples:

```
REMOVE,ELEMENT=QF
```

removes the first QF from the sequence, while

```
SELECT,FLAG=SEQEDIT,CLASS=QF
```

```
REMOVE,ELEMENT=SELECTED
```

removes all elements belonging to the class QF.

4.3.7 Replacing Elements

The command

```
REPLACE,ELEMENT=name1,BY=name2
```

replaces elements as follows:

ELEMENT The element(s) to be replaced. If this is the litteral **SELECTED**, all elements selected by a **SELECT,SEQEDIT** command are replaced, otherwise the first element whose name matches name1.

BY The name of the element which replaces name1.

4.3.8 Example for the Sequence Editor

```
SEQ:SEQUENCE
```

```
  B1:  B,  AT=19.115
```

```
  SF1: SF, AT=37.42
```

```
  QF1: QF, AT=38.70
```

```
  B2:  B,  AT=58.255
```

```
  SD1: SD, AT=76.74
```

```
  QD1: QD, AT=78.20
```

```
  ENDM: MARKER, AT=79.0
```

```
ENDSEQ
```

```
B2W:B2,ANGLE=0.1*B2[ANGLE]
```

```
SEQEDIT,SEQUENCE=SEQ
```

```
MOVE,ELEMENT=SF1,TO=-1.27,FROM=QF1
```

```
MOVE,ELEMENT=SD1,BY=0.01
```

```
REMOVE,ELEMENT=B2
```

```
INSTALL,ELEMENT=B2W,AT=58.255
```

```
ENDEDIT
```

This example moves the two sextupoles and replaces the element B2 by the element B2W. Replacing an element can be done as follows:

```
SEQEDIT,SEQUENCE=...
```

```
NEWQF:QF1,K=...
```

```
REPLACE,ELEMENT=QF1,BY=NEWQF1
```

```
ENDEDIT
```

This is equivalent to

```
SEQEDIT,SEQUENCE=...
```

```
NEWQF:QF1,K=...
```

```
INSTALL,ELEMENT=NEWQF1,AT=0,FROM=QF1
```

```
REMOVE,ELEMENT=QF1
```

```
ENDEDIT
```


4.4 More Examples for Beam Lines

4.4.1 CERN SPS Lattice

The CERN SPS lattice may be represented using the following beam elements:

```
QF:    QUADRUPOLE,...    ! focussing quadrupole
QD:    QUADRUPOLE,...    ! defocussing quadrupole
B1:    RBEND,...         ! bending magnet of type 1
B2:    RBEND,...         ! bending magnet of type 2
DS:    DRIFT,...         ! short drift space
DM:    DRIFT,...         ! drift space replacing two bends
DL:    DRIFT,...         ! long drift space
```

The SPS machine is represented by the lines

```
SPS:    LINE=(6*SUPER)
SUPER:  LINE=(7*P44,INSERT,7*P44)
INSERT: LINE=(P24,2*P00,P42)
P00:    LINE=(QF,DL,QD,DL)
P24:    LINE=(QF,DM,2*B2,DS,PD)
P42:    LINE=(PF,QD,2*B2,DM,DS)
P44:    LINE=(PF,PD)
PD:     LINE=(QD,2*B2,2*B1,DS)
PF:     LINE=(QF,2*B1,2*B2,DS)
```

In order not to overload the example, small gaps between magnetic elements have been omitted.

4.4.2 LEP Lattice

A similar method of defining accelerator structures has been used in Reference [20] to define the LEP structure. Translation of those element sequences to the MAD input format gives:

```
LEP:    LINE=(4*SUP)
SUP:    LINE=(OCT,-OCT)
OCT:    LINE=(LOBS,RFS,DISS,ARC,DISL,RFL,LOBL)
LOBS:   LINE=(L1,QS1,L2,QS2,L3,QS3,L4,QS4)
RFS:    LINE=(L5,QS5,L5,QS6,L5,2*(QS7,L5,QS8,L5))
DISS:   LINE=(QS11,L25,BW,L22,QS12,L25,B4,L22,QS13,L25,B4,
              L22,QS14,L25,B4,L31,QS15,L25,B4,L32,SF,L23,QS16)
ARC:    LINE=(L21,B6,L22,SD,L23,QD,
              7*(CELL(SF1,SD1),CELL(SF,SD)),CELL(SF1,SD1),
              L24,B6,L41,QF,L21,B6,L22,SD4,L23,QD,
              7*(CELL(SF4,SD3),CELL(SF3,SD4)),CELL(SF4,SD3),
              L24,B6,L22,SF3,L23)
DISL:   LINE=(QL16,L34,B4,L22,QL15,L33,B4,L22,QL14,L25,B4,
              L22,QL13,L25,B4,L22,QL12,L25,BW,L22,QL11)
RFL:    LINE=(2*(L5,QL8,L5,QL7),L5,QL6,L5,QL5,L5)
LOBL:   LINE=(QL4,L14,QL3,L13,QL2,L12,QL1,L11)
BW:     LINE=(W,L26,W)
B4:     LINE=(B,L26,B)
B6:     LINE=(B,L26,B,L26,B)
CELL(SF,SD): LINE=(L24,B6,L22,SF,L23,QF,L21,B6,L22,SD,L23,QD)
```

Here the element definitions have been left open.

4.5 Beam Line References

Many MAD commands require a reference to a beam line. Normally one enters the label of a beam line. If the beam line is defined with formal arguments, the label is followed by an actual argument list. MAD also accepts a sequence enclosed by parentheses instead of a name. Examples:

```
USE,CELL1
USE,CELL(SF1,SD1)
USE,(QF,D,QD,D)
```

4.6 Replacement Lists

The replacement list provides a powerful tool to describe complex machines. It replaces a set of identical labels in a sequence in turn by different items. A successive refinement of the machine description is made possible by redefining elements as beam lines or as replacement lists. A replacement list definition resembles a beam line definition:

```
label: LIST=(member{,member})
```

All items which can occur in a beam line can also occur in a list. The effect of a replacement list on a beam line is defined as follows:

1. Expand all replacement lists to the level of single labels. If a list member is a beam line name, do not expand it. Mark the first label of each expanded list as current.
2. Expand the given beam line. Each time the label of a replacement list is encountered, replace it by the current label in this list and mark the following label of the list as current.
3. If a substituted label is a beam line, expand it.

To study several arrangements of sextupole families, one may define one or more replacement lists for each arrangement. There is no need to change anything else in the machine structure. Example:

```
RING:  LINE=(5*(QF,D,SF,D,B,D,QD,D,SD,D,B))
SF:    LIST=(2*(SF1,SF2),SF1)
SD:    LIST=(2*(SD1,SD2),SD1)
USE,RING
```

In this example MAD proceeds as follows:

1. Expand the two lists:

```
SF:    LIST=(SF1,SF2,SF1,SF2,SF1)
SD:    LIST=(SD1,SD2,SD1,SD2,SD1)
```

2. Expand the line RING:

```
RING:  LINE=(QF,D,SF,D,B,D,QD,D,SD,D,B,D,
             QF,D,SF,D,B,D,QD,D,SD,D,B,D,
             QF,D,SF,D,B,D,QD,D,SD,D,B,D,
             QF,D,SF,D,B,D,QD,D,SD,D,B,D,
             QF,D,SF,D,B,D,QD,D,SD,D,B,D)
```

3. Replace labels in RING:

```

RING:   LINE=(QF,D,SF1,D,B,D,QD,D,SD1,D,B,D,
             QF,D,SF2,D,B,D,QD,D,SD2,D,B,D,
             QF,D,SF1,D,B,D,QD,D,SD1,D,B,D,
             QF,D,SF2,D,B,D,QD,D,SD2,D,B,D,
             QF,D,SF1,D,B,D,QD,D,SD1,D,B,D)

```

In a second example, we replace some drift spaces by special insertions:

```

FODO:   LINE=(4*(QF,D,QD,D))
D:      LIST=(DD,DC,3*DD,DC,2*DD)
DD:     DRIFT,L=2.0
DC:     LINE=(D1,PICKUP,D1)
D1:     DRIFT,L=0.5
PICKUP: MONITOR,L=1.0
        USE,FODO

```

In this example we have the following steps:

1. Expand the list D (note that DC is not expanded yet):

```

D:      LIST=(DD,DC,DD,DD,DD,DC,DD,DD)

```

2. Expand the beam line FODO:

```

FODO:   LINE=(QF,D,QD,D,QF,D,QD,D,QF,D,QD,D,QF,D,QD,D)

```

3. Replace labels in the beam line FODO:

```

FODO:   LINE=(QF,DD,QD,DC,QF,DD,QD,DD, \+
             QF,DD,QD,DC,QD,DD,QD,DD)

```

4. Expand the line DC:

```

FODO:   LINE=(QF,DD,QD,D1,PICKUP,D1,QF,DD,QD,DD, \+
             QF,DD,QD,D1,PICKUP,D1,QF,DD,QD,DD)

```

Replacement lists are circular. When a list is exhausted, and more replacement labels are required, the current element is reset to the first one, and replacement continues. Without changing the result, the lists in the first example could therefore also be written as

```

SF:     LIST=(SF1,SF2)
SD:     LIST=(SD1,SD2)

```

Replacement lists should *not* be used when lines are reflected. If a replacement list occurs within a reflected part of a beam line, MAD never reflects the replacement list and gives a warning message. Unless the list happens to be symmetric, the expansion would be wrong. Care should also be taken when expanding sub-lines of lines which refer to replacement lists. By definition, MAD always begins replacement at the first label of a replacement list. Here is an example what could go wrong:

```

CELL1:  LINE=(... ,SF,...)
CELL2:  LINE=(... ,SF,...)
CELL3:  LINE=(... ,SF,...)
SF:     LIST=(SF1,SF2,SF3)
ARC:    LINE=(CELL1,CELL2,CELL3)
        USE,CELL3

```

After the command USE, ARC the cell CELL3 will contain SF3. However, after the command USE, CELL3 the cell CELL3 will contain SF1. In case of doubt, use the PRINT command and check the correct replacement.

4.7 Ranges and Sets of Observation Points in a Line

Many MAD commands allow for the possibility to process or display a range or a subset of the elements occurring in the beam line expanded by the latest USE command. The selection mechanism has been updated as from version 8.18, and the information in this section is *not* valid for previous versions of MAD. The selection mechanism is used in the following commands

PRINT (see Section 5.4),
 SELECT (see Section 5.4),
 SPLIT (see Section 5.5),
 EALIGN using SELECT, FLAG=ERROR (see Section 8.1),
 EFIELD using SELECT, FLAG=ERROR (see Section 8.2),
 EFCOMP using SELECT, FLAG=ERROR (see Section 8.2),
 EPRINT (see Section 8.4),
 ESAVE (see Section 8.5),
 CONSTRAINT (see Section 12.3),
 USEKICK (see Section 9.4),
 USEMONITOR (see Section 9.4),
 INSTALL (see Section 4.3),
 MOVE (see Section 4.3),
 REMOVE (see Section 4.3),
 REPLACE (see Section 4.3),

These all have the following five parameters:

RANGE A range (an interval), from a first to a last element, both included, is defined as first/last. first and last may be one of:

- #S The start of the *full* beam line expanded by USE.
- #E The end of the *full* beam line expanded by USE.
- #F The start of the RANGE set by USE.
- #L The start of the RANGE set by USE.
 - A class or element name and its occurrence count: label[occur],
 - An explicit element name, if it is unique in the line.

CLASS The single name of a class, without an occurrence count.

PATTERN A UNIX regular expression.

FULL If this logical flag is true, all elements are selected, and the other attributes are ignored.

CLEAR If this logical flag is true, all selections are cleared, and the other attributes are ignored.

An element is selected if it belongs to the intersection of the three following sets (i. e. it belongs to all three sets):

- The element is within the specified RANGE. If no RANGE is given, this set contains all elements.
- The element is derived from the given CLASS. If no CLASS is given, this set contains all elements.
- The name of the element matches the regular expression PATTERN. If no PATTERN is given, this set contains all elements.

To get the effect of set union, use several commands. Example:

```
PRINT,RANGE=IP1/IP2                ! (1)
PRINT,CLASS=BB                      ! (2)
PRINT,PATTERN=".*\.L1"              ! (3)
PRINT,RANGE=IP1/IP2,CLASS=BB,PATTERN=".*\.L1" ! (4)
```

Command (1) selects all elements between IP1 and IP2, both included. Command (2) selects all elements with the name BB. Command (3) selects all elements whose name ends in ".L1". Command (4) selects all BB's whose name ends in ".L1" in the range between IP1 and IP2.

4.7.1 Examples for Ranges

Assume the following definitions:

```
M: MARKER
S: LINE=(C,M,D)
L: LINE=(A,M,B,2*S,A,M,B)
   USE,L,RANGE=M[2]/M[3]
```

The line L is equivalent to the sequence

A,M,B,C,M,D,C,M,D,A,M,B

Examples for ranges:

#4/#6 C[1],M[2],D[1],

A[1]/A[2] Positions A[1] to A[2] including all intermediate positions,

M[2]/M[3] From second to third M, also known as #F/#L,

Examples for single positions:

#0 or #S At the beginning of the line L,

C[1] After the first element C, can be written as #4,

M[3] After the third marker M, also #8 or #L,

#E After the last element of the line L.

More examples can be found in Section 5.11.

Chapter 5

Action Commands

Table 5.1: Environment Setting and Action Commands

Name	Meaning	Section
USE	Select working beam line	5.1
COGUESS	Enter guess for initial closed orbit	5.2
BEAM	Beam data: particle energy and charge, emittances etc.	5.3
PRINT	Select print positions	5.4
SELECT	Select output positions	5.4
SURVEY	Print geometry of machine	5.6
TWISS	Print lattice functions	5.7
IBS	Intra-beam scattering	5.8
EMIT	Equilibrium emittances	5.9.1
EIGEN	Eigenvectors for normal modes	5.9.2
ENVELOPE	Beam envelopes in 3 degrees of freedom	5.9.2
TWISS3	Mais-Ripken lattice functions	5.9.2
DYNAMIC	Dynamic normal form analysis	5.10
STATIC	Static normal form analysis	5.10
SPLIT	Interpolate within element for OPTICS command	5.5
OPTICS	Output lattice functions and element strengths for control system	5.11
SECTORMAP	TRANSPORT map for sectors of the ring	5.13

5.1 USE Statement

The USE statement specifies the beam line and its range to be used in subsequent commands. It must be entered before any physics computation is requested. It has the form

```
USE,PERIOD=line,RANGE=range,SYMM,SUPER=integer
```

If the PERIOD attribute appears, the USE command causes the following actions:

- Replace formal arguments by the corresponding actual arguments,
- Insert sub-lines in the proper places,
- Expand reflections and repetitions of sub-lines,
- Reset all selection flags (see Section 5.4),

- Clear all error definitions (see Chapter 8),

A data structure is generated containing one entry for each beam element occurring in the beam line, and one entry for the entrance and exit of each beam line or sub-line. This structure remains in memory until another USE creates a new one, or until the replacement of a beam element or beam line makes it invalid. A subsequent USE command referring to the same beam line will recreate the structure, and effectively clear all selection flags and error definitions.

If the RANGE attribute appears, USE marks beginning and end of range for future use. All subsequent commands which require a beam line operate on the range selected. Example:

```
A:      LINE=(...,B,...)  ! definition of A
B:      LINE=(...)        ! definition of B
      USE,A               ! expand beam line A
      ...                 ! some actions
B:      LINE=(...)        ! redefinition of B kills sequence
      USE,A               ! new expansion of beam line A
      ...                 ! more actions
```

The USE statement has four attributes:

PERIOD The beam line to be expanded (see Section 4.5). If omitted, the previous line is used without a new expansion.

RANGE The range of the beam line to be used. If PERIOD is given and RANGE is omitted, the range is the complete line. If PERIOD and RANGE are both omitted, the previous line and range are assumed. If RANGE is given, but PERIOD omitted, a new range is selected from the previous line.

SYMM A logical flag. If set, subsequent calculations are made as if the mirror image had been appended to the range.

SUPER An integer. Specifies the number of superperiods desired in the calculations. Quantities like tunes, chromaticities, and the like which refer to the machine circumference will be scaled with the value of SUPER (default: 1).

Example:

```
OCT: LINE=(...)          ! one octant of the machine
      USE,OCT,SYMM,SUPER=4
```

Here the beam line OCT is expanded. The effect is the same as if we had expanded the full machine as follows:

```
TURN: LINE=(4*(OCT,-OCT))
```

However, MAD needs only to walk through one octant, instead of through the whole machine. If the beam line is defined with a formal argument list, an actual argument list must be provided:

```
CELL(SF,SD): LINE=(...)
      USE,PERIOD=CELL(SF1,SD1)
```

In this example, one super-period is used. The beam line CELL is expanded with the formal arguments SF and SD replaced by SF1 and SD1 respectively. It is neither repeated nor made symmetric, since both SUPER and SYMM have been omitted.

5.2 COGUESS Command

For large closed orbit deviations some MAD commands have trouble to find the closed orbit. Often the position of the closed orbit at the start of the ring is known. In this case it may be entered by the command

```
COGUESS,X=real,PX=real,Y=real,PY=real,T=real,PT=real,TOLERANCE=real
```

The input parameters are the ones defined in Section 1.5. All subsequent commands will start the closed orbit search at the position given, until it is reset by another COGUESS command. The TOLERANCE parameter specifies the tolerance. Its default value is 10^{-6} .

5.3 BEAM Command

Many commands in MAD require the setting of various quantities related to the beam in the machine. These are entered by a BEAM command:

```
BEAM, PARTICLE=name,MASS=real,CHARGE=real,&
      ENERGY=real,PC=real,GAMMA=real,&
      EX=real,EXN=real,EY=real,EYN=real,&
      ET=real,SIGT=real,SIGE=real,&
      KBUNCH=integer,NPART=real,BCURRENT=real,&
      BUNCHED=logical,RADIATE=logical
```

Warning: BEAM updates, i. e. it replaces attributes explicitly mentioned, but does not return to default values for others! To reset to defaults, use RESBEAM (see below). The particle mass and charge are defined by:

PARTICLE The name of particles in the machine. MAD knows the mass and the charge for the following particles:

POSITRON The particles are positrons (default $MASS=m_e$, $CHARGE=1$),

ELECTRON The particles are electrons ($MASS=m_e$, $CHARGE=-1$),

PROTON The particles are protons ($MASS=m_p$, $CHARGE=1$),

ANTI-PROTON The particles are anti-protons ($MASS=m_p$, $CHARGE=-1$).

For other particle names one may enter:

MASS The particle mass in GeV.

CHARGE The particle charge expressed in elementary charges.

By default the total particle energy is 1 GeV. A different value can be defined by one of the following:

ENERGY The total energy per particle in GeV. If given, it must be greater than the particle mass.

PC The momentum per particle in GeV/c. If given, it must be greater than zero.

GAMMA The ratio between total energy and rest energy of the particles $\gamma = E/m_0$. If given, it must be greater than one. If the mass is changed a new value for the energy should be entered. Otherwise the energy remains unchanged, and the momentum and γ are recalculated. The emittances are defined by:

EX The horizontal emittance $E_x = \sigma_x^2/\beta_x$ (default: 1 m).

EY The vertical emittance $E_y = \sigma_y^2 / \beta_y$ (default: 1 m).

ET The longitudinal emittance $E_t = \sigma_e / (p_0 c) \cdot c \sigma_t$ (default: 1 m). The emittances can be replaced by the normalised emittances and the energy spread:

EXN The normalised horizontal emittance [m]: $E_{xn} = 4\beta\gamma E_x$ (ignored if E_x is given).

EYN The normalised vertical emittance [m]: $E_{yn} = 4\beta\gamma E_y$ (ignored if E_y is given).

SIGT The bunch length $c\sigma_t$ [m].

SIGE The *relative* energy spread $\sigma_e / p_0 c$ [1].

Certain commands compute the synchrotron tune Q_s from the RF cavities. If $Q_s \neq 0$, the relative energy spread $\sigma_e / p_0 c$ and the bunch length $c\sigma_t$ are

$$\frac{\sigma_e}{p_0 c} = \sqrt{\frac{2\pi Q_s E_t}{\eta C}}, \quad c\sigma_t = \sqrt{\frac{\eta C E_t}{2\pi Q_s}},$$

where C is the machine circumference, and $\eta = (1/\gamma^2) - (1/\gamma_{tr}^2)$. Finally, the BEAM command accepts

KBUNCH The number of particle bunches in the machine (default: 1).

NPART The number of particles per bunch (default: 0).

BCURRENT The bunch current (default: 0 A).

BUNCHED A logical flag. If set, the beam is treated as bunched whenever this makes sense.

RADIATE A logical flag. If set, synchrotron radiation is considered in all bipolar magnets.

The BEAM command changes only the parameters entered. The command

RESBEAM

resets all beam data to their defaults (listed in Table 5.2). It is defined in the command dictionary as

```
RESBEAM:BEAM, PARTICLE="POSITRON",ENERGY=1.0,EX=1.0,EY=1.0,ET=1.0,&
          KBUNCH=1.0,NPART=0.0,BCURRENT=0.0,-BUNCHED,-RADIATE,-LIST
```

Table 5.2: Default Beam Data

attribute	value	unit	attribute	value	unit
PARTICLE	POSITRON		ENERGY	1	GeV
EX	1	rad m	EY	1	rad m
ET	1	GeV s	KBUNCH	1	1
NPART	0	1	BCURRENT	0	A
BUNCHED	false		RADIATE	false	

Examples:

```
BEAM,    PARTICLE=ELECTRON,ENERGY=50,EX=1.E-6,EY=1.E-8,SIGE=1.E-3
...
BEAM,    RADIATE
...
RESBEAM
BEAM,    EX=2.E-5,EY=3.E-7,SIGE=4.E-3
```

The first command selects electrons, and sets energy and emittances. The second one turns on synchrotron radiation. The last two select positrons (by default), set the energy to 1 GeV (default), clear the synchrotron radiation flag, and set the emittances to the values entered.

The BEAM command stores all specified attributes in a data bank with the name BEAM. Some program modules of MAD also store data into this bank. Real expressions may refer to data in the BEAM bank using the notation

```
BEAM[attribute-name]
```

This notation refers to the value of attribute-name found in the BEAM bank before the command containing the reference is executed. For example, after an EMIT command both emittances are stored in the BEAM bank, but the vertical emittance EY may become zero. For subsequent tracking one may set the *new* value of EY to 1/2 times the *old* value of EX by entering

```
BEAM,EY=0.5*BEAM[EX]
```

5.4 PRINT and SELECT Statements

Each position in the beam line carries several associated selection flags. They are initially cleared by the USE command when the beam line is expanded. Output is selected by setting some of these flags by one of the commands

```
PRINT,RANGE=range,CLASS=name,PATTERN=string,FULL,CLEAR
SELECT,FLAG=name,RANGE=range,CLASS=name,PATTERN=string,FULL,CLEAR
```

The PRINT command always affects the print flag for SURVEY (see Section 5.6), or TWISS (see Section 5.7). The SELECT command has a FLAG attribute which may be one of the following:

ERROR This flag affects the selection for error definitions (see Section 8.1, 8.2).

SEQEDIT This flag affects the selection for the sequence editor (see Section 4.3).

TWISS A SELECT,TWISS statement is equivalent to PRINT. The two commands

```
PRINT,FULL
SELECT,FLAG=TWISS,FULL
```

have identical effect. Regardless of the setting of print flags, start and end points of the computation range are always printed by EIGEN, ENVELOPE, SURVEY, TWISS, and TWISS3.

OPTICS Selects output positions for OPTICS (see Section 5.11).

TRACK Selects print positions for tracking (see Chapter 13). Care must be taken in using this option, as it may generate a lot of output.

Four more values, intended for the programmer, specify debugging output:

FIRST Selects dumping of first-order transfer matrices for selected elements during closed orbit search in TWISS.

SECOND Selects dumping of second-order TRANSPORT maps for selected elements during their accumulation in TWISS.

REFER Selects dumping of first-order transfer matrices for selected elements during accumulation for adjusting RF cavities.

LIE Selects dumping of Lie-algebraic maps during their accumulation.

The other attributes are described in Section 4.7. PRINT and/or SELECT command(s) must be placed after the USE command, and before any command (e.g. TWISS) to be affected. Examples:

```
USE,OCT                ! print at beginning and end only
PRINT,#35/37           ! print at positions number 35 to 37
SELECT,TWISS,FULL      ! set all print flags
PRINT,CLEAR            ! clear all print flags
PRINT,OCT              ! set all print flags
PRINT,CELL[3],CLEAR    ! clear all flags,
                        ! then set flags for all of third CELL
```

More examples can be found near the end of this chapter.

5.5 SPLIT Command, Request Interpolation for OPTICS

The OPTICS command writes one table line for each element selected by SELECT, OPTICS. The output line contains the element parameters and the lattice functions for the element centre or its exit. The command

```
SPLIT,NAME=name,FRACTION=real&
    [,RANGE=range][,CLASS=name][,PATTERN=string][,FULL][,CLEAR]
```

selects additional positions for output of the lattice functions only. The positions are given the name NAME, and element parameters in those positions are output as zero. The elements are selected as by SELECT, and FRACTION specifies a fraction of the length of the element where output is desired. Any number of points can be selected in the same element; output occurs in order of increasing s . Example:

```
SPLIT,NAME=B1,FRACTION=0.25,CLASS=B
SPLIT,NAME=B2,FRACTION=0.5,CLASS=B
SPLIT,NAME=B3,FRACTION=0.75,CLASS=B
```

Gives three lines for each B, at 1/4, 1/2, and 3/4 of its length respectively and assigns the names B1,B2,B3 to the three positions.

5.6 SURVEY Statement

The SURVEY command computes the geometry of the machine:

```
SURVEY, X0=real,Y0=real,Z0=real,&
    THETA0=real,PHI0=real,PSI0=real,TAPE=file-name
```

It operates on the working beam line entered in the latest USE command (see Section 5.1). Its parameter list specifies the initial position and orientation of the reference orbit in the global coordinate system (X, Y, Z). Omitted attributes assume zero values. Valid attributes are:

X0 The initial X coordinate [m].

Y0 The initial Y coordinate [m].

Z0 The initial Z coordinate [m].

THETA0 The initial angle θ [rad].

PHIO The initial angle ϕ [rad].

PSIO The initial angle ψ [rad].

TAPE If TAPE=file-name appears MAD writes a full survey table on a disk file file-name. Appendix A describes the format of the file written. Appendix B explains the format for file names. TAPE alone is equivalent to TAPE="survey".

SURVEY prints one line for either end of the computation range and a summary. It also prints one line for each element and for the entrance and exit of each beam line, if this position has been selected by PRINT or by SELECT, FLAG=TWISS (see Section 5.4). The layout coordinates and angles have been defined in Section 1.3. Example:

```
SURVEY, TAPE=LAYOUT
```

This example computes the machine layout with zero initial conditions and writes the results on a file called LAYOUT.

5.7 TWISS Statement

The TWISS command causes computation of the linear lattice functions, and optionally of the chromatic functions. It operates on the working beam line defined in the latest USE command (see Section 5.1). TWISS prints one or two lines for either end of the computation range and a summary. It also prints one or two lines for each element and for the entrance and exit of each beam line, if this position has been selected by PRINT or by SELECT, FLAG=TWISS (see Section 5.4). The variables used have been defined in Section 1.5. Three forms are distinguished for the TWISS command. In all three forms initial values for the phase angles MUX and MUY are accepted. The relative energy error DELTAP may be entered in one of the forms

```
DELTAP=real{,real}
```

```
DELTAP=initial:final:step
```

The first form lists several numbers, which may be general expressions, separated by commas. The second form specifies an initial value, a final value, and a step, which must be constant expressions, separated by colons. Mixtures of both forms are accepted. Examples:

```
DELTAP=0.001                ! a single value
DELTAP=0.001,0.005          ! two values
DELTAP=0.001:0.007:0.002    ! four values
DELTAP=0.001,0.005:0.03:0.005 ! seven values
```

If DELTAP is missing, MAD uses the value 0.0. Further attributes common to all three forms of the TWISS statements are:

CHROM A logical flag. If set, MAD also computes the chromatic functions defined in [30].

COUPLE A logical flag. If set, MAD computes the coupled linear lattice functions as defined in [17, 33]. In this case CHROM is ignored.

TAPE If TAPE=table-name appears MAD writes a full Twiss table on a disk file file-name. Appendix A describes the format of the file written. Appendix B explains the format for file names. TAPE alone is equivalent to TAPE="twiss".

SAVE If SAVE=table-name appears on the command, MAD creates a full Twiss table in memory and gives it the name table-name (an identifier). Entering SAVE alone is equivalent to SAVE=TWISS. This table includes linear lattice functions as well as the chromatic functions for all positions.

TUNES If **TUNES=table-name** appears on the command, MAD creates a table of tunes and chromaticities versus the selected values of **DELTAP** and gives it the name **table-name** (an identifier). Entering **TUNES** alone is equivalent to **TUNES=TUNES**.

The tables are suited for plotting (see Chapter 10). They may be written on disk by the **ARCHIVE** command (see Section 7.5).

As from Version 8.19 of MAD the following variables are accessible after calling **TWISS**:

QX, **QY**, **QX'**, **QY'**,
ALFX, **ALFY**, **BETX**, **BETY**,
X0, **PX0**, **Y0**, **PY0**, **T0**, **PT0**.

The first five are global variable denoting horizontal and vertical tunes, horizontal and vertical chromaticities respectively, while the local Twiss parameters and closed orbit coordinates refer to the beginning of the beam line being USED.

5.7.1 Twiss Parameters for a Period

The simplest form of the **TWISS** command is

```

TWISS, DELTAP=real{,value},CHROM,COUPLE,&
      TAPE=file-name,SAVE=table-name

```

It computes the periodic solution for the specified beam line for all values of **DELTAP** entered (or for **DELTAP = 0**, if none is entered). Example:

```

USE,OCT,SYMM,SUPER=4
TWISS,DELTAP=0.001,CHROM,TAPE=OPTICS

```

This example computes the periodic solution for the linear lattice and chromatic functions for the beam line **OCT**, made symmetric and repeated in four superperiods. The **DELTAP** value used is 0.001. Apart from saving computing time, it is equivalent to the command sequence

```

RING: LINE=(4*(OCT,-OCT))
      USE,RING
      TWISS,DELTAP=0.001,CHROM,TAPE=OPTICS

```

5.7.2 Initial Values for Twiss Parameters taken from a Periodic Line

It is often useful to track the lattice functions starting with the periodic solution for another beam line. If this is desired the **TWISS** command takes the form

```

TWISS, DELTAP=real{,value},LINE=beam-line,&
      MUX=real,MUY=real&
      TAPE=file-name,SAVE=table-name

```

No other attributes should appear in the command. For each value of **DELTAP** MAD first searches for the periodic solution for the beam line mentioned in **LINE=beam-line**. The result is used as an initial condition for the lattice function tracking. In this form of the **TWISS** command the **SYMM** flag and the value for **SUPER** are ignored. Example:

```

CELL:  LINE=(...)
INSERT: LINE=(...)
      USE,INSERT
      TWISS,LINE=CELL,DELTAP=0.0:0.003:0.001,CHROM,TAPE

```

For four values of DELTAP the following happens: First MAD finds the periodic solution for the beam line CELL. Then it uses this solution as initial conditions for tracking the lattice functions of the beam line TWISS. The initial phases are not specified, they are set to zero. Output is also written on the file TWISS.

If any of the beam lines was defined with formal arguments, actual arguments must be provided:

```
CELL(SF,SD): LINE=(...)
INSERT(X):   LINE=(...)
             USE,INSERT
             TWISS,LINE=CELL(SF1,SD1)
```

5.7.3 Twiss Parameters with Numerical Initial Values

Initial values for linear lattice functions may also be numerical. Two equivalent methods are available to specify initial values:

- In the TWISS command itself, as in

```
TWISS,   BETX=real,ALFX=real,MUX=real,&
        BETY=real,ALFY=real,MUY=real,&
        DX=real,DPX=real,DY=real,DPY=real,&
        X=real,PX=real,Y=real,DPY=real,&
        WX=real,PHIX=real,DMUX=real,&
        WY=real,PHIY=real,DMUY=real,&
        DDX=real,DDY=real,DDPX=real,DDPY=real,&
        TAPE=file-name,SAVE=table-name,&
        DELTAP=real:real:real
```

- In a separate command, BETA0 (see also Section 5.7.4):

```
INITIAL: BETA0,&
        BETX=real,ALFX=real,MUX=real,&
        BETY=real,ALFY=real,MUY=real,&
        DX=real,DPX=real,DY=real,DPY=real,&
        X=real,PX=real,Y=real,PY=real,T=real,PT=real,&
        WX=real,PHIX=real,DMUX=real,&
        WY=real,PHIY=real,DMUY=real,&
        DDX=real,DDY=real,DDPX=real,DDPY=real
TWISS,   BETA0=INITIAL,TAPE=file-name,SAVE=table-name,&
        DELTAP=real:real:real
```

In the second case the TWISS command may override some of the values set by BETA0. All variables listed in Section 1.5.3 are permitted as input attributes, but BETX and BETY are required in either case. The flags SYMM and COUPLE, and the value for SUPER are ignored in this use of the TWISS command. As entered in this command, the initial conditions cannot depend on the quantity DELTAP, and can thus be correct only for one such value. This should be remembered when using this form of the TWISS command.

5.7.4 SAVEBETA Command, Save Lattice Parameters for Later Use

Sometimes it is useful to transfer computed lattice parameters to later commands. The command sequence

```
USE,...
SAVEBETA,LABEL=name,PLACE=place
TWISS,...
```

may help to do this. When reaching the place `place` during execution of `TWISS MAD` will create a `BETA0` bank with the name `name`. This bank is filled with the values of all lattice parameters in `place`. The `SAVEBETA` command has two attributes:

LABEL The name to be given to the created `BETA0` bank.

PLACE A position within the selected range of the working beam line.

Example 1:

```
USE,CELL
SAVEBETA,LABEL=END,PLACE=#E
TWISS
USE,INSERT
TWISS,BETA0=END
```

This will first calculate the periodic solution of the line `CELL`, and then track lattice parameters through `INSERT`, using all end conditions (including phases and orbit) in `CELL` to start.

Example 2:

```
USE,CELL
SAVEBETA,LABEL=END,PLACE=#E
TWISS
USE,INSERT
TWISS,BETX=END[BETY],BETY=END[BETX]
```

This is similar to the first example, but the beta functions are interchanged (overwritten).

5.8 IBS Command, Intra-Beam Scattering

The `IBS` command takes an internal table (default: `TWISS`) generated by a previous `TWISS,SAVE` command and prints out information about intra-beam scattering. This command uses the method by Bjorken and Mtingwa [5, 12].

```
IBS,TABLE=table
```

This single attribute `TABLE` specifies the Twiss table to be used (default: `TWISS`). The `IBS` command uses the same print flags as `TWISS` to select output positions. Example:

```
TWISS,SAVE=LATTICE
IBS,TABLE=LATTICE
```

5.9 Emittances, Eigenvectors, Normal Modes, Beam Envelope

5.9.1 Electron Beam Parameters

The command

```
EMIT,DELTAP=real
```

adjusts the RF frequencies such as to obtain the specified average energy error. More precisely, the revolution frequency f_0 is determined for a fictitious particle with constant momentum error $\delta p/p_0 c = \delta_s = \text{DELTA P}$ which travels along the design orbit. The RF frequencies are then set to $h f_0$.

If the machine contains at least one RF cavity, and if synchrotron radiation is on (Section 5.3), the EMIT command computes the equilibrium emittances and other electron beam parameters using the method of A. Chao [10]. In this calculation the effects of quadrupoles, sextupoles, and octupoles along the closed orbit is also considered. Thin multipoles are ignored, unless they have a fictitious length LRAD different from zero.

If the machine contains no RF cavity, or if synchrotron radiation is off, it only computes the parameters which are not related to radiation. Example:

```
RFC: RFCAVITY,HARMON...,VOLT=...
      BEAM,ENERGY=100.0,RADIATE
      EMIT,DELTA P=0.01
```

As from Version 8.19 of MAD the following variables are accessible after calling EMIT:

```
Q1, Q2, Q3,
X0, PX0, Y0, PY0, T0, PT0.
```

The first three are global variable denoting horizontal, vertical and synchrotron tunes respectively, while the local Twiss parameters and closed orbit coordinates refer to the beginning of the beam line being USED.

5.9.2 Representations of Beam with Full Coupling

The three commands

```
EIGEN,SAVE=name
ENVELOPE,SAVE=name,SIGMA0=name,LINE=line
TWISS3,SAVE=name,LINE=line
```

track different representations of the beam in three degrees of freedom. If the ring contains RF cavities, the command EMIT should be run before any of these commands in order to adjust the RF frequencies. Their actions are the following:

EIGEN Tracks the eigenvectors for the three eigen-modes and prints the principal phase for each of them. Example:

```
RFC: RFCAVITY,HARMON...,VOLT=...
      BEAM,ENERGY=100.0,RADIATE
      EMIT,DELTA P=0.01
      EIGEN,SAVE=name
```

The EIGEN table defines the linear normal modes of the motion. When each normal mode is projected onto the (x, y) plane one obtains an ellipse. The table then defines the following expressions, taken from [3]:

```
TILT1 The tilt of the first normal mode ellipse, expressed in degrees, (-45 .. +45 degrees).
TILT2 The tilt of the second normal mode ellipse, expressed in degrees, (-45 .. +45 degrees).
ECCN1 The excentricity of the first normal mode ellipse,
ECCN2 The excentricity of the second normal mode ellipse.
```


ENVELOPE Tracks the beam envelope in a way similar to the TRANSPORT program. This uses the eigenvectors and emittances for the three modes to determine the initial beam envelope, and tracks this envelope along the ring. Example:

```
RFC: RFCAVITY,HARMON...,VOLT=...
      BEAM,ENERGY=100.0,RADIATE
      EMIT,DELTAP=0.01
      ENVELOPE,SAVE=name,SIGMA0=name,LINE=line
```

TWISS3 Lists the coupled lattice functions defined in two papers by H. Mais [27] and G. Ripken [31]. These are essentially the projections of the lattice functions for the eigen-modes on the three planes. Example:

```
RFC: RFCAVITY,HARMON...,VOLT=...
      BEAM,ENERGY=100.0,RADIATE
      EMIT,DELTAP=0.01
      TWISS3,SAVE=name,LINE=line
```

The attributes mean:

SAVE If this keyword is present, it requests saving of the computed functions in a table. It may be followed by a name to be given to the table. If it appears without a value the table has the same name as the command. If the SAVE keyword is omitted, no table is generated. If the table is to be processed outside MAD, it must be written to disk using the ARCHIVE command (Section 7.5). The positions appearing in the tables are selected by means of the command

```
SELECT,OPTICS,RANGE=...
```

in a way similar to the OPTICS command.

SIGMA0 May contain the name of a SIGMA0 block, to be used as the initial beam envelope in ENVELOPE.

LINE May specify a beam line whose periodic solution should be taken for initial condition in ENVELOPE or TWISS3.

For all three commands printing and selection for saving is controlled by the PRINT command (5.4). A SIGMA0 block can be generated by two means. First, it may be entered like any definition:

```
name:SIGMA0,  X,PX,Y,PY,T,PT,DX,DPX,DY,DPY,DT,DPT,&
              SIGX,R21,R31,R41,R51,R61,SIGPX,R32,&
              R42,R52,R62,SIGY,R43,R53,R63,SIGPY,&
              R54,R64,SIGT,R65,SIGT
```

The first group of six values defines the initial orbit. For a static machine (constant energy) the second group of six values defines the initial dispersion. The values SIGxx are the standard deviations and the Rij the correlations defining the beam ellipsoid in TRANSPORT sense.

A SIGMA0 block can also be generated and used as follows:

```
USE,RING
SAVESIGMA,LABEL=SIGMAHERE,PLACE=HERE
EMIT ! required only if RF cavities are present
ENVELOPE
...
USE,RING,RANGE=HERE/THERE
ENVELOPE,SIGMA0=SIGMAHERE
```

The first envelope command computes the periodic beam envelope for the line RING, and fills in the SIGMA0-block with the name SIGMAHERE. The second ENVELOPE uses the values in position HERE to track the envelope from HERE to THERE.

5.10 DYNAMIC and STATIC, Lie-Algebraic Analysis

The two commands

```
DYNAMIC, DELTAP=real,MAP,ORBIT,A,N,&
        RESONANCE,EXPONENT,HAMILTON,INVARIANT
```

```
STATIC,  DELTAP=real,MAP,ORBIT,FIXED,T,A,N,&
        RESONANCE,EXPONENT,HAMILTON,BETATRON,&
        NONLINEAR,CONJUGATE,INVARIANT
```

both evaluate the fourth-order Lie transformation for one turn and print the eigenvectors and lattice parameters at the end of the system. Both have various options to print selected maps and functions. The differences are the following:

DYNAMIC The RF cavities are adjusted as for the command NORMAL. The transfer map is interpreted as dynamic, i.e. there is synchrotron motion with an average momentum error DELTAP.

STATIC The program assumes that there are no cavities, and it interprets the transfer map as static, i.e. having constant momentum error DELTAP.

Both commands have the following logical flags:

MAP Print the original map to be analysed.

ORBIT Print the map around the closed orbit.

A Print the conjugating map (the transformation from the closed orbit to normal form).

N Print the normal form map.

RESONANCE Print the resonance coefficients left in the normal form.

EXPONENT Print the exponent for the normal form.

HAMILTON Print the Hamiltonian for the normal form.

INVARIANT Print the linear invariants.

The DYNAMIC command adjusts the RF frequencies according to the revolution frequencies. It uses the following attributes for this purpose:

RFCAVITY The name of a class of cavities whose phase lag is to be adjusted to make the time lag of the closed orbit approximately zero.

DELTAP The average momentum error $\Delta p/p_0 c$ for the closed orbit.

The STATIC command also finds the fixed point of the map, that is the variation of the closed orbit with momentum. It has the following additional attributes:

DELTAP The constant momentum error $\Delta p/p_0 c$ for the closed orbit.

FIXED Print the map about the fixed point.

T Print the transformation to the fixed point.

BETATRON Print the betatron map.

NONLINEAR Print the non-linear factor of the betatron map.

CONJUGATE Print the map conjugate to the betatron factor.

After a call to the DYNAMIC command, the following global variables can be used in any arithmetic expression:

Q1, Q2, DQ1DE1, DQ1DE2, DQ1DE3, DQ2DE2, DQ2DE3, DQ3DE3.

They denote normal mode tunes, and anharmonicities, respectively.

After a call to the STATIC command, the following global variables can be used in any arithmetic expression:

Q1, Q2, Q1', Q2', Q1'', Q2'', DQ1DE1, DQ1DE2, DQ2DE2.

They denote normal mode tunes, (linear and nonlinear) chromaticities and anharmonicities, respectively. The command has an added attribute LONG which must be set true to get the same output as in previous versions. If it is false (the default), the output is reduced to the minimum. For speed reasons the closed orbit is only recomputed when a parameter change requires to do so.

5.11 OPTICS Command

Control system applications often require tables with arbitrary selections of element data and/or lattice functions. Such tables can be generated by the command

```
OPTICS, BETX=real,ALFX=real,MUX=real,&
        BETY=real,ALFY=real,MUY=real,&
        DX=real,DPX=real,DY=real,DPY=real,&
        X=real,PX=real,Y=real,PY=real,T=real,PT=real&
        WX=real,PHIX=real,DMUX=real,&
        WY=real,PHIY=real,DMUY=real,&
        DDX=real,DDY=real,DDPX=real,DDPY=real,&
        LINE=beam-line,BETA0=name,CENTRE,&
        FILENAME=file-name,DELTAP=real:real:real,&
        COLUMNS=name{,name}
```

It first creates a table in memory. It then writes it on a disk file in coded TFS format (see Appendix C). Initial conditions for the optical functions are specified like for the TWISS command. They are explained in Sections 1.5 and 5.7. Further attributes are:

CENTRE Normally output occurs at the exit of each selected element. If the CENTRE flag is on, output occurs at the centre of each selected element;

FILENAME The output is written on the file file-name (default: optics). Note that the resulting table is written on this file and erased from the computer's memory. In order to plot it, it must be read back by an RETRIEVE command.

COLUMNS Up to 50 table columns may be selected by name for output. All optical and chromatic functions listed in Sections 1.5.1, 1.5.3, and 1.5.4 are accepted. Further possibilities are:

NAME The element name.

CLASS The smallest class containing the element as an object.
 KEYWORD The element keyword.
 TYPE The TYPE attribute of the element.
 S The longitudinal position.
 DP The current $\Delta E/p_s$. It is not called DELTAP to avoid conflicts with the command attribute of this name which selects a single value for the calculation.
 L The element length.
 RADLOSS The systematic part of the relative radiation loss for this element.
 KOL The integrated normal dipole component of this element (excluding kicks).
 KnL The integrated normal multipole component. The value of n specifies the multipole with $2n + 2$ poles, and $0 \leq n \leq 9$.
 KSOL The integrated skewed dipole component of this element (excluding kicks).
 KSnL The integrated skewed multipole component. The value of n specifies the multipole with $2n + 2$ poles, and $0 \leq n \leq 9$.
 VOLT The cavity voltage.
 LAG The RF frequency lag.
 FREQ The RF frequency.
 HARMON The RF harmonic number.
 TILT The roll angle of the element, zero for multipoles.
 KS The integrated solenoid strength.
 HKICK The horizontal corrector deflection.
 VKICK The vertical corrector deflection.
 E1 The entrance pole face angle for dipoles.
 E2 The exit pole face angle for dipoles.
 H1 The entrance pole face curvature for dipoles.
 H2 The exit pole face curvature for dipoles.
 EFIELD The electrostatic field for a separator.

Example:

```

! Define element classes for a simple cell:
B:      SBEND,L=35.09, ANGLE = 0.011306116
QF:      QUADRUPOLE,L=1.6,K1=-0.02268553
QD:      QUADRUPOLE,L=1.6,K1=0.022683642
SF:      SEXTUPOLE,L=0.4,K2=-0.13129
SD:      SEXTUPOLE,L=0.76,K2=0.26328
! Define the cell as a sequence:
CELL:  SEQUENCE
      B1:  B,      AT=19.115
      SF1:  SF,      AT=37.42
      QF1:  QF,      AT=38.70
      B2:  B,      AT=58.255,ANGLE=B1[ANGLE]
      SD1:  SD,      AT=76.74
      QD1:  QD,      AT=78.20
      ENDM:  MARKER, AT=79.0
  
```

```

ENDSEQUENCE
USE,CELL
SELECT,OPTICS,SBEND,QUAD,SEXT
OPTICS,FILENAME="cell.optics.f",EXIT,COLUMNS=NAME,S,BETX,BETY

```

The resulting table file is listed in Table 5.3.

Table 5.3: An OPTICS Output Table Example

```

@ GAMTR          %f      64.3336
@ ALFA           %f      0.241615E-03
@ XIY            %f      -.455678
@ XIX            %f      2.05279
@ QY             %f      0.250049
@ QX             %f      0.249961
@ CIRCUM          %f      79.0000
@ DELTA          %f      0.000000E+00
@ COMMENT         %20s "DATA FOR TEST CELL"
@ ORIGIN          %24s "MAD 8.01      IBM - VM/CMS"
@ DATE           %08s "19/06/89"
@ TIME           %08s "09.47.40"
* NAME            S          BETX          BETY
$ %16s           %f          %f          %f
  B1              36.6600      24.8427      126.380
  SF1             37.6200      23.8830      130.925
  QF1             39.5000      23.6209      132.268
  B2              75.8000      124.709      25.2153
  SD1             77.1200      130.933      23.8718
  QD1             79.0000      132.277      23.6098

```

5.12 Examples for SPLIT and OPTICS Commands

The following is an excerpt of the LEP description:

```

! Bending magnet pairs:
! The definitions take into account the different magnetic length
! for the inner and outer pairs of a group of six.
B2:    RBEND,      L=11.55,ANGLE=KMB2,K1=KQB,K2=KSB, &
        E1=-.25*B2[ANGLE],E2=-.25*B2[ANGLE]
B2OUT: B2,        ANGLE=1.00055745184472*KMB2, &
        E1=-.25*B2OUT[ANGLE],E2=-.25*B2OUT[ANGLE]
B2MID: B2,        ANGLE=1.00111490368947*KMB2, &
        E1=-.25*B2MID[ANGLE],E2=-.25*B2MID[ANGLE]

! Quadrupoles:
MQ:    QUADRUPOLE,L=1.6      ! standard quadrupoles =
QD:    MQ,      K1=KQD      ! cell quadrupoles, defocussing
QF:    MQ,      K1=KQF      ! cell quadrupoles, focussing

```

! Sextupoles:

```
MSF:  SEXTUPOLE, L=0.40      ! F sextupoles
MSD:  SEXTUPOLE, L=0.76      ! D sextupoles
SF1.2: MSF,      K2=KSF1.2    ! F family 1, circuit 2
SF2.2: MSF,      K2=KSF2.2    ! F family 2, circuit 2
SF3.2: MSF,      K2=KSF3.2    ! F family 3, circuit 2
SD1.2: MSD,      K2=KSD1.2    ! D family 1, circuit 2
SD2.2: MSD,      K2=KSD2.2    ! D family 2, circuit 2
SD3.2: MSD,      K2=KSD3.2    ! D family 3, circuit 2
```

! Orbit correctors and monitors:

```
CH:   HKICK,      L=0.4      ! Horizontal orbit correctors
CV:   VKICK,      L=0.4      ! Vertical orbit correctors
MHV:  MONITOR,    L=0        ! Orbit position monitors
```

LEP:SEQUENCE

```
...
QF23.R1:      QF,      AT=639.180037
  SF2.QF23.R1: SF2.2,  AT=640.460037
  B2L.QF23.R1: B2OUT,  AT=647.257037
  B2M.QD24.R1: B2MID,  AT=659.147037
  B2R.QD24.R1: B2OUT,  AT=671.037037
  CV.QD24.R1:  CV,      AT=677.392037, KICK=KCV24.R1
  PU.QD24.R1:  MHV,      AT=677.712037
QD24.R1:      QD,      AT=678.680037
  SD2.QD24.R1: SD2.2,  AT=680.140037
  B2L.QD24.R1: B2OUT,  AT=686.757037
  B2M.QF25.R1: B2MID,  AT=698.647037
  B2R.QF25.R1: B2OUT,  AT=710.537037
  CH.QF25.R1:  CH,      AT=716.942037, KICK=KCH25.R1
QF25.R1:      QF,      AT=718.180037
  SF1.QF25.R1: SF1.2,  AT=719.460037
  B2L.QF25.R1: B2OUT,  AT=726.257037
  B2M.QD26.R1: B2MID,  AT=738.147037
  B2R.QD26.R1: B2OUT,  AT=750.037037
  CV.QD26.R1:  CV,      AT=756.392037, KICK=KCV26.R1
  PU.QD26.R1:  MHV,      AT=756.712037
QD26.R1:      QD,      AT=757.680037
  SD1.QD26.R1: SD1.2,  AT=759.140037
  B2L.QD26.R1: B2OUT,  AT=765.757037
  B2M.QF27.R1: B2MID,  AT=777.647037
  B2R.QF27.R1: B2OUT,  AT=789.537037
QF27.R1:      QF,      AT=797.180037
  SF3.QF27.R1: SF3.2,  AT=798.460037
  B2L.QF27.R1: B2OUT,  AT=805.257037
  B2M.QD28.R1: B2MID,  AT=817.147037
  B2R.QD28.R1: B2OUT,  AT=829.037037
  PU.QD28.R1:  MHV,      AT=835.712037
QD28.R1:      QD,      AT=836.680037
  SD3.QD28.R1: SD3.2,  AT=838.140037
```

```

B2L.QD28.R1: B2OUT, AT=844.757037
B2M.QF29.R1: B2MID, AT=856.647037
B2R.QF29.R1: B2OUT, AT=868.537037
CH.QF29.R1: CH, AT=874.942037, KICK=KCH29.R1
...
ENDSEQUENCE

```

In the above structure it is easy to select many sets of observation points:

- Print at all F sextupoles:

```
PRINT,MSF
```

- Split all quadrupoles at 1/3 of their length for OPTICS command:

```
SPLIT,QUADRUPOLE,FACTOR=1/3
```

- Misalign two quadrupole QF25.R1 and QD26.R1:

```
EALIGN,QF25.R1,QD26.R1,DX=0.001*GAUSS(),DY=0.0005*GAUSS()
```

- Print first-order matrices for elements B2L.QD24.R1 through CV.QD26.R1:

```
SELECT,FIRST,B2L.QD24.R1[1]/CV.QD24.R1[1]
```

Print lattice functions at all F-sextupoles of the first family, if connected to the second circuit:

```
PRINT,SF1.2
```

5.13 SECTORMAP Command, Maps for Parts of the Ring

The commands

```
SELECT,FLAG=OPTICS,...
SECTORMAP,FILENAME=string,DELTAP=real
```

cause computation of the TRANSPORT maps for the ring sectors delimited by the positions selected by the SELECT command. The attributes of SECTORMAP are:

FILENAME The name of the file to be written.

DELTAP The average momentum error to be used.

For use with the TRAIN program, the sector limits and the DELTAP value must agree in all commands. Example:

```

USE,LEP
SELECT,OPTICS,CLASS=IP
OPTICS,FILENAME="optics",COL=NAME,S,X,BETX,MUX,DX,Y,BETY,MUY,DY
SECTORMAP,FILENAME="forward",DELTAP=0.001

```

Chapter 6

Calculation of Beam Parameters

Table 6.1: Electron Beam Parameter Command

Name	Function	Section
BMPM	Calculate and print beam parameters	6.2

6.1 Introduction

The program BEAMPARAM (see [21]) which served as the basis for this module, is valid for e^+e^- storage rings only. Inside MAD, however, this new module will use the correct mass and radius of the proton if this is chosen by the user. This means that all formulae will remain correct even in the case of the proton. However, it has not been checked whether some of them have been derived under the assumption that the particles are electrons. In the text, only e^+e^- storage rings are mentioned.

The utility of the BMPM command is achieved at the price of a few simplifying assumptions and approximations concerning the accelerator lattice. Although these conditions are quite well satisfied by many e^+e^- storage rings, the user does need to be aware of them. Accordingly we list the most important among them below:

Flat machine If any element has a non-zero TILT parameter then the command will abort. This excludes, in particular, skew quadrupoles and vertical bends, hence the vertical dispersion will be zero everywhere. Combined-function dipoles are not accepted.

Emittance coupling Although the previous assumption implies that the betatron coupling vanishes so that the vertical emittance will be determined by the small transverse components of the photon recoil, the vertical emittance is in fact calculated using a non-zero betatron coupling parameter κ (KAPPA). This factor is defined by the equation $E_y/E_x = \kappa^2(J_x/J_y)$.

Damping partition number variation In reality, this is achieved by small shifts of the equilibrium momentum which place the beam on a slightly different orbit. BMPM computes the values of the damping partition numbers on the central orbit with Eqn. 6.1–Eqn. 6.1. These are later used in the calculations of damping times, emittances and energy spread. In reality, their values on other (off-momentum) orbits involve further synchrotron radiation integrals. However, changes in the damping partition number are effected by artificially changing the value of the synchrotron integral I_4 . For convenience, it is the value of the ratio I_4/I_2 that is specified in the input rather than I_4 itself. It is the user's responsibility to ensure that the damping partition number variation brought about in this way does not violate the aperture constraints of his machine.

$$\begin{aligned}
 \text{Horizontal } J_x &= 1 - I_4/I_2 \\
 \text{Vertical } J_y &= 1 \\
 \text{Energy } J_e &= 2 + I_4/I_2
 \end{aligned}$$

Luminosity The luminosity value computed assumes that the beam sizes have their natural values, and does not allow for any beam-beam blow-up.

6.2 BMPM Command

The BMPM command performs the calculation of beam parameters for an e^+e^- storage ring according to the parameters and options specified below. It requires the prior execution of the commands BEAM and TWISS with SAVE option. This command requires the beam energy to be set, and the machine to contain an RF cavity whose shunt impedance and filling time are specified. It does not handle combined function dipoles.

```
BMPM, NINT=integer,DELQ=real,TAUQ=real,BUCKET=real,&
      KAPPA=real,I4I2=real,EXDATA=real,FX=real,FY=real,&
      KHM=real,SYNRAD=logical,CLORB=logical,&
      TOUSCH=logical,SINGLE=logical,EVARY=logical,&
      MIDARC=logical,INTERACT=range,RANGE=range
```

NINT Number of crossing points (default: twice the number of bunches).

DELQ Maximum permissible beam-beam tune shift per crossing (default: 0.06).

TAUQ Quantum lifetime (default: 600 Minutes).

BUCKET σ_b/σ_e , where σ_b is the bucket half-height, and σ_e the relative r.m.s. energy spread (default: 0).

KAPPA Coupling factor between horizontal and vertical betatron oscillations (default: 0).

I4I2 Ratio of the two synchrotron integrals I_4 and I_2 (default: 0).

EXDATA Horizontal emittance E_x (default: 0) in μm .

FX Horizontal multiplication factor (default: 10).

FY Vertical multiplication factor (default: 10).

KHM Higher mode loss factor (default: 0) in V/pC.

SYNRAD Performs synchrotron radiation calculations (default: F).

CLORB Performs closed orbit calculations (default: F).

TOUSCH Performs Touschek lifetime calculations (default: F).

SINGLE True for one beam, false for two beams (default: F).

EVARY Vary energy between current nominal energy and twice its value until the calculated power is equal to the power given (default: F).

MIDARC If MIDARC=T, then the e^- and e^+ are assumed to be interleaved with equal spacing; otherwise they are assumed to be coincident. The small difference in arrival times which occurs in practice is neglected. For single beams, the MIDARC parameter is irrelevant (default: F).

INTERACT Position (name, number, etc.) of the interaction element. If a range is entered, the first element of the range is used (default: #E, see Section 4.7).

RANGE Element range for which a detailed table is printed (default: no range, see Section 4.7).

Example:

```
xxx:   RFCAVITY,SHUNT=... ,TFILL=...
BEAM,  ENERGY=100.0
TWISS, SAVE
BMPM
```

6.3 Output of overall machine parameters

6.3.1 Synchrotron integrals

This table gives the synchrotron integrals I_1 to I_5 as defined in reference [22]. Further integrals, I_{6x} and I_{6y} , defined by M.Basseti, as well as I_8 are also calculated and printed.

6.3.2 Machine parameters

This table contains the tunes Q_x, Q_y , the betatron functions β_x^* and β_y^* and the dispersions D_x^* and D_y^* at the crossing point. Since the program does not handle vertical dispersion, D_y^* is set to zero. Also printed are the momentum compaction α , the machine circumference C , the revolution time t_0 , the damping partition numbers J_x, J_y, J_e , the derivative of J_x with respect to dE/E , then dE/E itself, and finally the frequency change df_{RF} .

6.3.3 Beam parameters and luminosities

This table gives the beam energy, the coupling, the larger of the horizontal and vertical tune shifts ΔQ , the energy loss per turn due to synchrotron radiation U_0 , the r.m.s. energy spread σ_e , the magnetic rigidity $B\rho$, and the damping times for the horizontal and vertical betatron oscillations, τ_x and τ_y , and for the synchrotron oscillations τ_e .

σ_x^* and σ_y^* are the r.m.s. beam radii at the crossing point, defined by INTERACT:

- σ_{x0}^* and σ_{y0}^* are the uncoupled betatron values.
- σ_{xc}^* and σ_{yc}^* are the values due to coupled betatron oscillations.
- σ_{xT}^* and σ_{yT}^* are the total values, including coupled betatron oscillations and energy oscillations.

Convention for the coupling κ : If a *non-zero value* of the coupling parameter κ is read in, it is used for calculating $\sigma_{xc}^*, \sigma_{yc}^*, \sigma_{xT}^*, \sigma_{yT}^*, E_{xc}, E_{yc}$ etc. and never gets changed. In the output κ is labelled (D)KAPPA.

If the input value $\kappa = 0$, BMPM computes κ so that the beam-beam tune shifts ΔQ_x and ΔQ_y vary with the current as shown schematically in Figure 6.1. For currents at the beam-beam limit ΔQ and above, the coupling κ is adjusted so that $\Delta Q_x = \Delta Q_y$ by having $\sigma_{xT}^*/\sigma_{yT}^* = \beta_x^*/\beta_y^*$. (This is usually called optimum coupling.) For currents below the beam-beam limit ΔQ , the coupling κ is adjusted so that $\Delta Q_x < \Delta Q$ but $\Delta Q_y = \Delta Q$.

Beam current Three different methods are foreseen for calculating the circulating current I , the stored number of particles N in each beam, and the luminosity L .

1. If neither a value for the current I_{xy} nor a value for the RF generator power P_g is given in the data, i.e. $I_{xy} = P_g = 0$, then the currents I_x and I_y are calculated so that $\Delta Q_x = \Delta Q$ and $\Delta Q_y = \Delta Q$ respectively. These currents are then used to calculate the stored number of particles N_x and N_y and the luminosities L_x and L_y . For optimum coupling, $I_x = I_y$, $N_x = N_y$

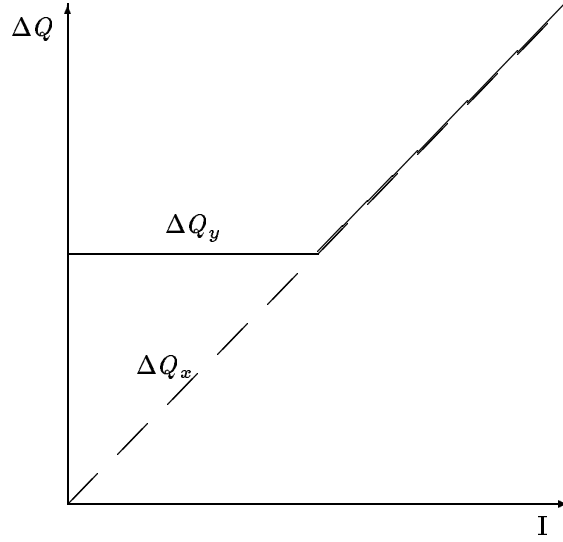


Figure 6.1: Variation of the beam-beam tune shifts ΔQ_x and ΔQ_y with the current obtained by varying the coupling κ

and $L_x = L_y$. This option is typical for storage rings in the energy range where the luminosity is limited by the beam-beam tune shift.

2. If a value for the current I_{xy} is specified in the data, i.e. $I_{xy} \neq 0$, then $I_x = I_y = I_{xy}$ and these values are used to calculate the beam-beam tune shifts ΔQ_x and ΔQ_y , N_x and $N_y (= N_x)$ and the luminosities L_x and $L_y (= L_x)$. This option typically applies to storage rings in the energy range where the current is limited, and to synchrotrons. The ΔQ printed is the larger of ΔQ_x and ΔQ_y , it may well exceed the beam-beam limit.
3. If an RF generator power $P_g \neq 0$ is specified in the data, the current I_x is adjusted so that the RF generator power necessary to sustain it is equal to the input value P_g . The current $I_x = I_y$ is then used to calculate ΔQ_x and ΔQ_y , N_x and $N_y (= N_x)$, L_x and $L_y (= L_x)$.

In the output, the input parameters, $\Delta Q, I_x, P_g$, are labelled (D) and the calculated parameters (C).

The beam-beam bremsstrahlung lifetime τ_{bb} is calculated according to [37], using the bucket height obtained in the calculation of the RF parameters, and assuming that there are N_{int} identical crossings. For machines with different crossings the calculated τ_{bb} must be scaled appropriately.

6.3.4 RF related parameters

The RF parameter calculation is done in two different ways, depending on the value of the RF generator power in the data. If $P_g = 0$ there are two possible values for the current I_x : either $P_g = I_{xy} = 0$, in which case a value for I_x is calculated, or $P_g = 0$ and $I_{xy} \neq 0$ in which case $I_x = I_{xy}$. Once one has a value for I_x then a single-pass calculation is sufficient to obtain all the output parameters described below. If $P_g \neq 0$, the current I_x is adjusted so that the input value and the calculated value of P_g agree, before the rest of the output parameters is calculated.

The output contains the harmonic number f_{RF}/f_0 , the peak RF voltage V_{RF} , the stable phase angle ϕ_s , the synchrotron tune Q_s and the synchrotron frequency f_s , the half-height of the bucket σ_b in units of σ_e , the r.m.s. bunch length σ_s , the quantum lifetime τ_q and the Touschek lifetime $\tau_{1/2}$ calculated according to [34, 11, 35].

If $\gamma\sigma_{x'}/\sigma_b > 100$ then we use an approximation which is good to within 2% [35], otherwise we use the exact formula. Since the approximation involves an integration along the orbit, it is somewhat time-consuming. The exact formula will take even longer as a double integral is involved. So, it is preferable not to ask for the Touschek lifetime unless it is really needed.

Convention for RF frequency f_{RF} : The harmonic number f_{RF}/f_0 is computed such that it is the multiple of k_b which gives an RF frequency f_{RF} closest to the input value.

Convention for RF voltage V_{RF} : The RF voltage V_{RF} may be obtained in one of three ways. Firstly it may be read in as data as an attribute of the RF cavities. Secondly it may be computed such that an energy loss ($U_0 + k_{hm}Q_b$) is compensated, at the specified quantum lifetime τ_q , where Q_b is the bunch charge and k_{hm} is the higher mode loss factor. Finally it can be calculated from the condition that the bucket half height σ_b is BUCKET times the r.m.s. energy spread σ_e .

The computation of the RF generator power P_g involves a complete beam-loading calculation according to [36]. The input data required are: the length of the active RF system L_c (sum of physical lengths of all cavities), its shunt impedance per unit length Z , and its unloaded filling time T_{fill} (all attributes of the RF cavities). The beam is described by parameters which are already known—such as the current I_x and the number of bunches k_b —and by the logical variables SINGLE and MIDARC. If SINGLE=T, then the calculation is done for a single beam circulating in the machine; otherwise the calculation is done for two counter-rotating beams of e^- and e^+ . The arrival times of the e^- and e^+ bunches at the RF cavities are controlled by the variable MIDARC. If MIDARC=T, then the e^- and e^+ are assumed to be interleaved with equal spacing; otherwise they are assumed to be coincident. The small differences in arrival times which occurs in practice [7] is neglected. For single beams, the MIDARC parameter is irrelevant.

The two cavity attributes BETRF and LAG, represent the RF coupling parameter β_{RF} and the cavity tuning angle ψ_{RF} respectively. If both β_{RF} and ψ_{RF} are zero, BMPM will adjust them such that the RF generator power, P_g , is minimised. If β_{RF} is different from zero, ψ_{RF} will be varied such that the RF generator power P_g is minimised. If both are different from zero, no such optimisation takes place.

The output of the beam loading calculation consists of β_{RF} , ψ_{RF} , τ_q and BUCKET, all preceded by (C) or (D) accordingly.

6.4 Evaluation of the beam size

If a valid element range is given (RANGE), a table of beam sizes is printed for the elements in the range. For each element, it gives the number, name, η , η' , β_x , β_y , α_x , α_y , $F_x\sigma_x$, and $F_y\sigma_y$, all taken at the entrance of the element. The values of F_x and F_y are input data (FX and FY).

Convention for beam radii: The horizontal beam radius σ_x is calculated for an uncoupled beam using the emittance E_{x0} , the vertical one for a fully coupled beam using the emittance $E_{x0}/2$.

Because of this convention, the actual beam sizes for all amounts of coupling should be smaller than or equal to the figures printed. It may be found useful to split elements with strong variations of β_x and β_y , e.g. low- β quadrupoles, into several pieces in the data in order to get better results for the beam size variation in these elements.

6.5 Synchrotron radiation, Calculated for One Beam

If in addition to a valid range, SYNRAD=T, the table of beam sizes will contain in addition for each element the horizontal beam divergence $\sigma_{x'}$, the vertical beam divergence $\sigma_{y'}$, the bending radius

ρ , the critical photon energy E_c , the synchrotron radiation power/metre, and the number of photons/m/s/keV at the critical energy.

Convention for divergences: $\sigma_{x'}$ is calculated for an uncoupled beam; $\sigma_{y'}$ is calculated for a fully coupled beam; ρ , E_c etc. are only calculated for bending magnets ($\phi \neq 0$) or quadrupoles ($K_1 \neq 0$). For all these calculations, combined function magnets are not permitted ($K_1 \phi \neq 0$). In quadrupoles, ρ , E_c etc., as defined in [24], are calculated for a fully coupled beam at a distance σ from the quadrupole axis.

Again, it may be found useful to split elements with strong variations of β_x and β_y , e.g. low- β quadrupoles, into several pieces in the data in order to obtain better results for the variation of the synchrotron radiation parameters in those elements.

6.6 Closed orbit calculations

The sensitivity of the closed orbit to alignment errors of the quadrupoles, and excitation errors and tilts of the bending magnets can be expressed by integrals around the machine circumference of appropriate powers of the amplitude functions β_x and β_y , and the focussing parameter K_1 .

If in addition to a valid range, CLORB=T, the table of beam sizes will contain in addition for each element the amplification factors P_x and P_y for all quadrupoles and bending magnets. For quadrupoles, P_x and P_y are amplification factors for horizontal and vertical misalignments. They are the ratio between the closed orbit distortion in that element, which will not be exceeded in 98% of all machines, to the r.m.s. displacement of the quadrupoles, assumed to be the same for all quadrupoles. For bending magnets, P_x is the 98% ratio between closed orbit distortions in metres and relative r.m.s. magnet errors $\Delta B/B$, and similarly P_y is the ratio between closed orbit distortions in metres and the r.m.s. tilt of the magnets.

Chapter 7

File and Pool Handling

Table 7.1: File Handling Commands

Name	Meaning	Section
ASSIGN	Assign standard streams to files	7.2
SAVE	Save machine structure in MAD format	7.3
CALL	Read alternate input file	7.3
RETURN	Return to calling file	7.3
EXCITE	Read element excitations	7.4
INCREMENT	Increment element excitations	7.4
ARCHIVE	Archive an internal table	7.5
RETRIEVE	Retrieve an internal table	7.5
PACKMEMORY	Garbage removal	7.6
STATUS	Show status of files	7.7
POOLDUMP	Dump data pool to disk	7.8
POOLLOAD	Reload data pool from disk	7.8

7.1 I/O Data Sets (Files)

In MAD most data streams are referred to by their name and opened by FORTRAN OPEN statements with a file name. Only the standard input and output are referred to by number and must be assigned to devices by external commands (job control language, execute files or the like). The standard data streams used in MAD are listed in Table 7.2. The format of file names is described in Appendix B for various computer operating systems.

Table 7.2: Standard Files Used by MAD

Purpose	unit	file name
Command dictionary input	4	dict
Normal input	5	
Input lines and error messages	6	
Plot output (GKS metafile)	8	mad.metafile
Plot output (HIGZ metafile)	8	mad.ps
Normal output	14	print
Dynamic tables	15	table

7.2 ASSIGN Statement

The ASSIGN statement is able to reroute three of the standard I/O streams to different files:

```
ASSIGN, DATA=file-name, ECHO=file-name, PRINT=file-name
```

For each stream=file-name clause, it connects the stream stream to the file file-name. The special file name TERMINAL denotes the interactive terminal. Example:

```
ASSIGN, ECHO=PRINT, PRINT=TERMINAL
```

This example sends the ECHO stream to the standard PRINT file, and the PRINT stream to the terminal. Standard connections can be reestablished by

```
ASSIGN, ECHO=ECHO, DATA=DATA, PRINT=PRINT
```

7.3 SAVE, CALL, and RETURN Statements

The SAVE command

```
SAVE, FILENAME=string, PATTERN=string
```

causes all beam element, beam line, and parameter definitions to be written on the named file. The output format is similar to the normal input format. The file may be read again in the same run. The attributes are

FILENAME The name of the file to be written (default: save).

PATTERN A “wild-card” pattern permitting selective saving

The default is to save all definitions; this is equivalent to the PATTERN=".*". For definition of the pattern refer to Section 1.7. Examples:

```
SAVE, FILENAME="structure"
SAVE, FILENAME="quadrupoles", PATTERN="Q. ."
SAVE, FILENAME="strengths", PATTERN="K.*QD.*\R1"
```

The first command saves all definitions onto file structure. The second command saves all definitions whose names begin with the letter Q and have exactly three characters onto file quadrupoles. The third command saves all definitions whose names begin with K, contain QD, and end with .R1. The two occurrences of .* replace arbitrary strings of any length (including zero); and \. replaces a literal period character.

The command The CALL command

```
CALL, FILENAME=string
```

serves to read an alternate input file. Input continues on that file until a RETURN statement or end of file is encountered. The attribute is

FILENAME The name of the file to be read (default: save).

Example:

```
CALL, FILENAME=STRUCT
```

reads back the file written by the above SAVE example.

The command RETURN

RETURN

on an alternate input file causes reading to resume with the next line following the CALL statement on the standard input file. Both CALL and RETURN statements cannot be followed by any other information (except comments) on the same line. The CALL/RETURN structure can be nested up to a level 10; i.e. an alternate input file can issue another CALL. Circular calls are forbidden. The following example shows a possible sequence of input:

```

contents of          contents of          contents of
main input file      file "STRUCT"        file "ACTION"
\vspace{2ex}
TITLE, "page header"
any commands
CALL, FILENAME=STRUCT
----->
                        beam element definitions
                        beam line definitions
                        parameter definitions
                        RETURN
<-----
more definitions
CALL, FILENAME=ACTION
----->
                                action commands
                                RETURN
<-----
more action commands
SAVE, FILENAME=STRUCT
STOP
```

7.4 Element Excitations

Two commands, EXCITE and INCREMENT allow to enter complete lists of element excitations:

```

EXCITE,    FILENAME=file-name,NAME=name,VALUE=name
INCREMENT, FILENAME=file-name,NAME=name,VALUE=name
```

Both have the same attributes:

FILENAME The name of a TFS file (see Appendix C) to be read.

NAME The name of the table column containing the strength names.

VALUE The name of the table column containing the strength values (for EXCITE) or increments (for INCREMENT).

The names in the NAME column must refer to global MAD parameters. If a name is not known, MAD creates a new parameter of this name. The TFS table may contain other columns, but they will be ignored.

7.5 Archiving and Retrieving Dynamic Tables

7.5.1 ARCHIVE Command

ARCHIVE writes a dynamic table on a formatted TFS file (see Appendix C):

ARCHIVE, TABLE=name, FILENAME=file-name

The command has two attributes:

TABLE The name of a previously created table to be written. It may reside in memory or (partly) in bulk storage.

FILENAME The name of the file to be written (default: twiss).

Example:

ARCHIVE, TABLE=TWISS, FILENAME=LEP.TWISS.F

7.5.2 RETRIEVE Command

RETRIEVE reads a formatted TFS file (see Appendix C) into memory and creates a dynamic table.

RETRIEVE, TABLE=name, FILENAME=file-name

The command has two attributes:

TABLE The name of a dynamic table to be created. It need not be the same as when the table was written. If there is already a table with this name, it is deleted before reading a new copy.

FILENAME The name of the file to be read (default: twiss).

Example:

RETRIEVE, TABLE=TWISS1, FILENAME=LEP.TWISS.F

reads file written by the ARCHIVE example and creates a new table TWISS1.

7.6 PACKMEMORY Command, Garbage Removal

The MAD memory management routines, based on the ZEBRA system, reclaim unused memory space automatically. One may however force garbage collection at any time by the PACKMEMORY command

PACKMEMORY

7.7 STATUS

The STATUS command

STATUS

causes MAD to list all known files on the message stream. These include all files which were referred to in the current program run.

7.8 Dumping and Reloading the Memory Pool

The POOLDUMP command

POOLDUMP,FILENAME=file-name

dumps the complete memory pool on the named disk file. Its attribute is:

FILENAME The name of the file to receive the dump (default: pooldump).

Memory dumped by POOLDUMP can be reloaded in a subsequent MAD run by a POOLLOAD command:

POOLLOAD,FILENAME=file-name

This may save considerable time compared to regenerating a complex data structure from input. Its attribute is:

FILENAME The name of the file to reload (default: pooldump).

Chapter 8

Error Definitions

Table 8.1: Error Definition Commands

Name	Function	Section
EALIGN	Specify misalignment(s)	8.1
EFCOMP	Specify field error(s)	8.2
EFIELD	Specify field error(s)	8.2
EOPT	Specify error options	8.3
EPRINT	List errors assigned to elements	8.4
ESAVE	Save errors assigned to file	8.5

This chapter describes the commands which provide error assignment and output of errors assigned to elements.¹ It is possible to assign alignment errors and field errors to single beam elements or to ranges of beam elements. Errors can be specified both with a constant or random values. Error definitions consist of four types of statements listed in Table 8.1. They may be entered after having selected a beam line by means of a USE command (see Section 5.1).

8.1 Misalignment Definitions

Alignment errors are defined by the EALIGN command. The misalignments refer to the local MAD reference system for a perfectly aligned machine (see Section 1.1). Possible misalignments are displacements along the three coordinate axes, and rotation about the coordinate axes. Alignment errors can be assigned to all beam elements except drift spaces. The effect of misalignments is treated in a linear approximation. Position monitors can be given read errors in both horizontal and vertical planes. Monitor read errors (MREX and MREY) are ignored for all other elements. Each new EALIGN statement replaces the misalignment errors for all elements in its range.

Alignment error values are defined by the statement

```
EALIGN, DX=real,DY=real,DS=real,&  
        DPHI=real,DTHETA=real,DPSI=real,&  
        MREX=real,MREY=real
```

The EALIGN command affects all elements selected by preceding SELECT,FLAG=ERROR commands. Its other attributes are:

```
EALIGN,TYPE=MQ,MA,MB,MC,MD
```

¹contributed by E. Nordmark (1985).

RANGE and TYPE can be given in the same command. All elements will be affected which have TYPE=name or which are members of range.

DX The misalignment in the x -direction for the entry of the beam element (default: 0 m). $DX > 0$ displaces the element in the positive x -direction (see Figure 8.1).

DY The misalignment in the y -direction for the entry of the beam element (default: 0 m). $DY > 0$ displaces the element in the positive y -direction (see Figure 8.3).

DS The misalignment in the s -direction for the entry of the beam element (default: 0 m). $DS > 0$ displaces the element in the positive s -direction (see Figure 8.1).

DPHI The rotation around the x -axis. A positive angle gives a greater y -coordinate for the exit than for the entry (default: 0 rad, see Figure 8.3).

DTHETA The rotation around the y -axis according to the right hand rule (default: 0 rad, see Figure 8.1).

DPSI The rotation around the s -axis according to the right hand rule (default: 0 rad, see Figure 8.1).

MREX The horizontal read error for a monitor. This is ignored if the element is not a monitor (see Figure 8.4). If $MREX > 0$, the reading for x is too high (default: 0 m).

MREY The vertical read error for a monitor. This is ignored if the element is not a monitor (see Figure 8.4). If $MREY > 0$, the reading for y is too high (default: 0 m).

Example:

```
EALIGN,QF[2],DX=0.002,DY=0.0004*RANF(),DPHI=0.0002*GAUSS()
```

Refer to Section 2.4.8 for random value formats.

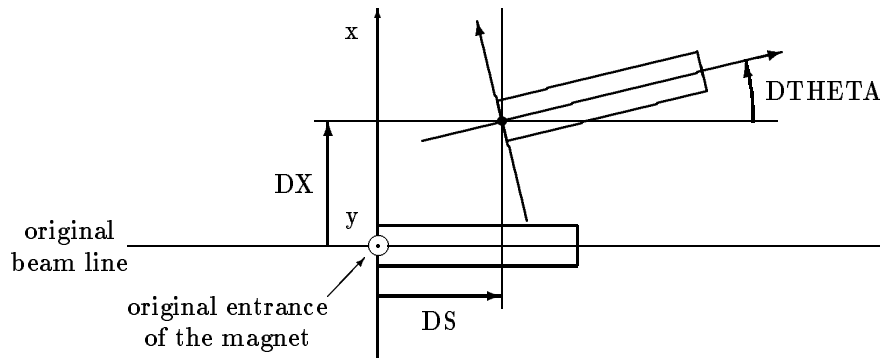
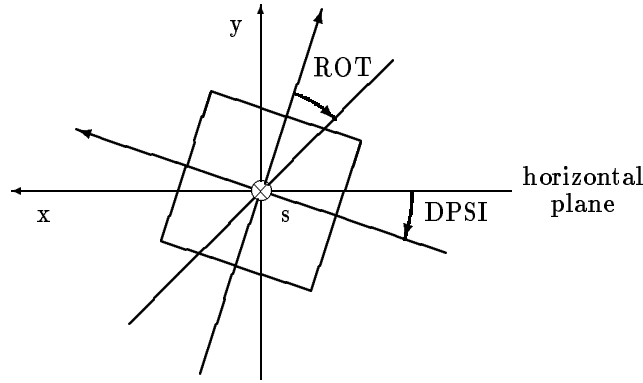
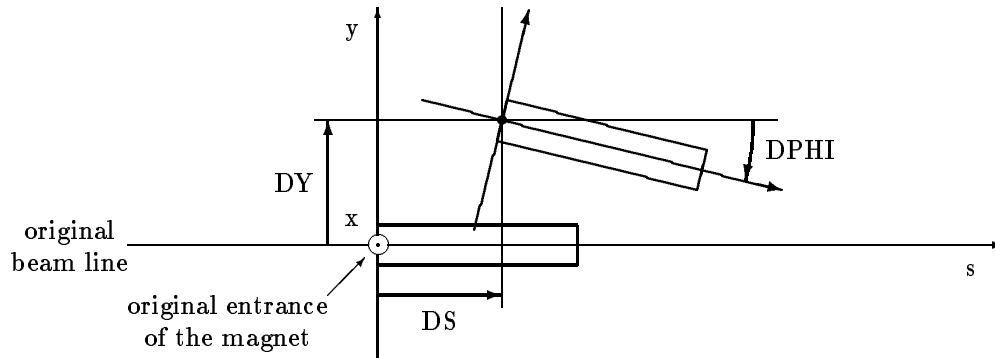


Figure 8.1: Example of Misplacement in the (x, s) -plane

8.2 Field Error Definitions

Field errors can be entered as relative or absolute errors. Different multipole components can be specified with different kinds of errors (relative or absolute). If an attempt is made to assign both a relative and an absolute error to the same multipole component, the absolute error is used and a warning is given. Relative errors cannot be assigned to an element of the type MULTIPOLE. Relations between absolute and relative field errors are listed below.

All field errors are specified as the integrated value $\int K_i ds$ along the magnet axis in m^{-i} (see Section 1.4). At present field errors may only affect field components allowed as normal components

Figure 8.2: Example of Misplacement in the (x, y) -planeFigure 8.3: Example of Misplacement in the (y, s) -plane

in a magnet. This means for example that a dipole may have errors of the type dipole, quadrupole, sextupole, and octupole; but not of the type decapole. There is no provision to specify a global relative excitation error affecting all field components in a combined function magnet. Such an error may only be entered by defining the same relative error for all field components.

Field errors can be specified for all magnetic elements by one of the statements

```
EFIELD, ORDER=integer,RADIUS=real,DBL=real,DKL(i)=real,&
        DBLR=real,DKLR(i)=real,BROT=real,ROT(i)=real
EFCOMP, ORDER=integer,DBLN=real,DKLN(i)=real,&
        DBLS=real,DKLS(i)=real
```

Each new EFIELD or EFCOMP statement replaces the field errors for all elements in its range (s). Any old field errors present in the range are discarded. EFIELD defines the error in terms of relative or absolute amplitude and rotation; while EFCOMP defines them in terms of absolute components.

The EFIELD or EFCOMP commands affect all elements selected by preceding SELECT,FLAG=ERROR commands. Note that this selection mechanism is not backward compatible with MAD Versions preceding 8.18. Their other attributes are:

ORDER This integer is only used for MULTIPOLE elements. It defines why high component the relative errors must be multiplied.

RADIUS Radius R where $DKLR(i), 1 \leq i \leq 10$ is specified (default 1 m). This attribute is required if DBLR or any DKLR(i) is specified.

DBL Absolute error amplitude for the dipole strength (default: 0 rad). Its value is $\int K_0 ds$ over the length of the magnet (see below).

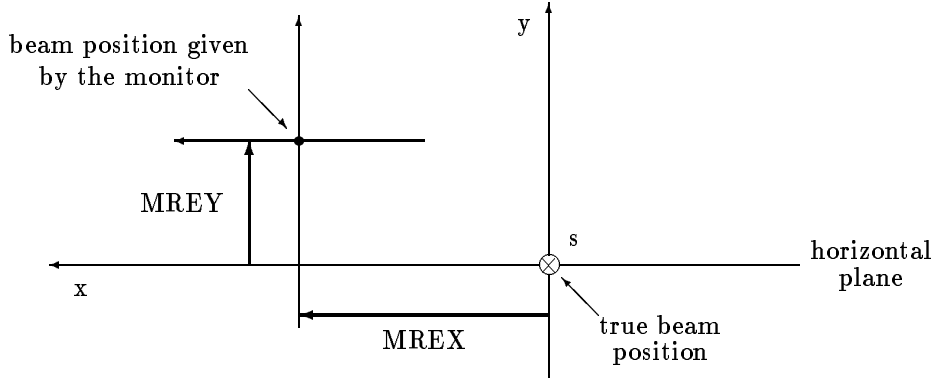


Figure 8.4: Example of Read Errors in a monitor

DKL(*i*) Absolute error amplitude for the multipole strength with $(2 * i + 2)$ poles (default: 0 m⁻ⁿ) Its value is $\int K_i ds$ over the length of the magnet (see below).

DBLR Relative error in the dipole strength (default: 0, see below).

DKLR(*i*) Relative error in multipole strength with $(2 * i + 2)$ poles (default: 0, see below). This attribute requires that RADIUS is also given.

BR0T Rotation angle ϕ_0 for the dipole error (default: 0, see Figure 8.2).

ROT(*i*) Rotation angle ϕ_i for the multipole with $(2 * i + 2)$ poles, where $1 \leq i \leq 10$ (default: 0 rad, see Figure 8.2).

DBLN Absolute error for the horizontal dipole strength (default: 0).

DKLN(*i*) Absolute error for the normal multipole strength with $(2 * i + 2)$ poles (default: 0).

DBLS Absolute error for the vertical dipole strength (default: 0).

DKLS(*i*) Absolute error in skewed multipole strength with $(2 * i + 2)$ poles (default: 0).

The dimensions for DKL(), DKLR(), DKLN(), DKLS(), and ROT() are preset to 10 in the command dictionary, but this can be changed easily. Examples:

```
EFIELD,TYPE=MQ,DBL=5.0E-4,DKL(3)=0.0025*RANF(),DKL(5)=0.0092*GAUSS()
EFIELD,TYPE=MQ,DBLN=5.0E-4,DKLN(3)=0.0025*RANF(),DKLN(5)=0.0092*GAUSS()
```

Refer to Section 2.4.8 for random value formats.

8.2.1 Field Errors in Bending Magnets

Only the normal dipole, quadrupole, and sextupole components of the error field are considered in a bending magnet. They are rotated together with the main field. The error field components have the values:

$$\begin{aligned} \Delta K_0 L &= \text{DBL} = \text{DBLN} = K_0 L \cdot \text{DBLR}, \\ \Delta K_1 L &= \text{DKL}(1) \cos \phi_1 = \text{DKLN}(1) = K_0 L \frac{1}{R} \cdot \text{DKLR}(1) \cdot \cos \phi_1, \\ \Delta K_2 L &= \text{DKL}(2) \cos \phi_2 = \text{DKLN}(2) = K_0 L \frac{2}{R^2} \cdot \text{DKLR}(2) \cdot \cos \phi_2, \\ \Delta K_3 L &= \text{DKL}(3) \cos \phi_3 = \text{DKLN}(3) = K_0 L \frac{3}{R^3} \cdot \text{DKLR}(3) \cdot \cos \phi_3. \end{aligned}$$

8.2.2 Field Errors in Quadrupoles

Only the normal quadrupole component of the error field, that is the excitation error, is considered. It is rotated together with the main field. The error field component has the value:

$$\Delta K_1 = \text{DKL}(1) \cdot \cos \phi_1 = \text{DKLN}(1) = K_1 L \cdot \text{DKLR}(1) \cdot \cos \phi_1.$$

8.2.3 Field Errors in Sextupoles

Only the normal sextupole component of the error field, that is the excitation error, is considered. It is rotated together with the main field. The error field component has the value:

$$\Delta K_2 L = \text{DKL}(2) \cdot \cos \phi_2 = \text{DKLN}(2) = K_2 L \cdot \text{DKLR}(2) \cdot \cos \phi_2.$$

8.2.4 Field Errors in Multipoles

Error fields and main field are rotated separately and then added. The error field components are always absolute values:

$$\begin{aligned} (\Delta B_{x0} + i \cdot \Delta B_{y0})L &= \text{DBL} \cdot \exp(-i \cdot \text{BROT}) = \text{DBLN} + i \cdot \text{DBLS}, \\ (\Delta K_{xk} + i \cdot \Delta B_{yk})L &= \text{DKL}(k) \cdot \exp(-(k+1)i \cdot \text{ROT}(k)) = \text{DKLN}(k) + i \cdot \text{DKLS}(k). \end{aligned}$$

8.2.5 Field Errors in Orbit Correctors

The dipole error field is added to the main field and then rotated. A rotation (TILT) must not be used when closed orbit corrections are to be computed. The error field components are always absolute values:

$$(\Delta B_0 + i \cdot \Delta B_{y0})L = \text{DBL} \exp(-i \cdot \text{BROT}) = \text{DBLN} + i \cdot \text{DBLS}.$$

8.3 Error Option Command

EOPT The random generator for MAD is taken from [25]. The error option command specifies different seeds for random values:

EOPT,SEED=real

SEED Selects a particular sequence of random values. The SEED expression must fall in the range $[0 \dots 999999999]$ (default: 123456789). SEED alone continues with the current sequence (see also Section 2.4.5).

ADD If this logical flag is set, an **EALIGN**, **EFIELD**, or **EFCOMP**, causes the errors to be added on top of existing ones. If it is not set, new errors overwrite any previous definitions.

Example:

EOPT,SEED=987456321

8.4 EPRINT: Print Errors

This command prints a table of errors assigned to elements. MAD prints for all elements selected in preceding **SELECT**, **ERROR** commands. Field errors are printed as absolute errors, because all relative errors are transformed to the corresponding absolute error at definition time. An error print is requested by the **EPRINT** statement

EPRINT

It has no attributes.

8.5 ESAVE: Save Errors

`ESAVE,FILENAME=string,ORDER=integer,RADIUS=real`

This command saves the errors assigned to elements. MAD prints for all elements selected in preceding `SELECT,ERROR` commands. Field errors are saved as relative values. For this purpose the attributes `ORDER` and `RADIUS` are used, which have the same meaning as for `EFIELD` and `EFCOMP`. The `FILENAME` attribute is the name of the file to be written. Example:

`ESAVE,FILENAME="error_file",ORDER=1,RADIUS=1.0`

Chapter 9

Closed Orbit Commands

Table 9.1: Commands Related to the Closed Orbit

Name	Function	Section
CORRECT	Complete correction algorithm	9.6
GETDISP	Read table of dispersion readings	9.1
GETKICK	Read table of corrector settings	9.2
GETORBIT	Read table of monitor readings	9.3
MICADO	Correction by MICADO algorithm	9.5
PUTDISP	Write table of dispersion readings	9.1
PUTKICK	Write table of corrector settings	9.2
PUTORBIT	Write table of monitor readings	9.3
USEKICK	Activate/deactivate Correctors	9.4
USEMONITOR	Activate/deactivate Monitors	9.4

There are six commands related to the closed orbit, listed in Table 9.1. Before using one of them a beam line must be selected by means of a USE command (see Section 5.1).

9.1 GETDISP and PUTDISP Statements

The two commands

GETDISP,FILENAME=file-name

PUTDISP,FILENAME=file-name

serve the following purposes:

GETDISP Reads a TFS file file-name containing dispersion readings for the orbit position monitors (see Appendix C). This file may have been written by a PUTDISP command or by an external program. The readings can be used in a subsequent MICADO command to find corrector settings for correcting the vertical dispersion.

PUTDISP Finds the closed orbit and writes the dispersion readings at all orbit position monitors on a TFS file file-name. (see Appendix C). This file may be read again by MAD or an external program.

The format of file-name is described in Appendix B. The default file-name is dispersion. The table must have the following columns:

PUNAME Name of the monitor (string).

DX Horizontal dispersion (real).

DY Vertical dispersion (real).

STATUS Integer zero for active, nonzero for inactive.

9.2 GETKICK and PUTKICK Statements

The two commands

```
GETKICK,FILENAME=file-name [,ADD]
PUTKICK,FILENAME=file-name [,PLANE=X|Y]
```

serve the following purposes:

GETKICK Reads a TFS file file-name containing the settings for the orbit correctors (see Appendix C).

This file may have been written by a PUTKICK command or by an external program. The excitations will be used for any subsequent closed orbit search, unless changed by MICADO or CORRECT commands. If the flag ADD is true, the setting read are added to any previous settings.

PUTKICK Writes the current setting of the orbit correctors on a TFS file file-name. (see Appendix C).

This file may be read again by MAD or an external program.

The format of file-name is described in Appendix B. The default file-name is setting. For reading the table must have the following columns:

STR_NAME Name of the corrector (string).

K_N_H Horizontal setting (real).

K_N_V Vertical setting (real).

STATUS Integer zero for active, nonzero for inactive.

By default the table written has the same format. If the attribute PLANE=X or PLANE=Y is entered, only the strengths for that plane are written, and there is only one column for settings, headed by K_N. Such a table *cannot* be read by MAD.

9.3 GETORBIT and PUTORBIT Statements

The two commands

```
GETORBIT,FILENAME=file-name
PUTORBIT,FILENAME=file-name
```

serve the following purposes:

GETORBIT Reads a TFS file file-name containing orbit readings for the orbit position monitors (see Appendix C). This file may have been written by a PUTORBIT command or by an external program. The readings can be used in a subsequent MICADO command to find corrector settings for correcting the orbit positions.

PUTORBIT Finds the closed orbit and writes the readings of all orbit position monitors on a TFS file file-name (see Appendix C). This file may be read again by MAD or an external program.

The format of file-name is described in Appendix B. The default file-name is orbit. The table must have the following columns:

PUNAME Name of the monitor (string).

X Horizontal position (real).

Y Vertical position (real).

STATUS Integer zero for active, nonzero for inactive.

9.4 USEKICK and USEMONITOR, Activate/deactivate elements

To provide more flexibility with orbit and dispersion correction two commands are provided:

```
USEMONITOR,STATUS=flag,MODE=name&
           [,RANGE=range][,CLASS=name][,PATTERN=string]
USEKICK,   STATUS=flag,MODE=name&
           [,RANGE=range][,CLASS=name][,PATTERN=string]
```

The purpose of the two commands is:

USEMONITOR Activates or deactivates a selection of beam position monitors. This command affects elements of types MONITOR, HMONITOR, or VMONITOR.

USEKICK Activates or deactivates a selection of orbit correctors. This command affects elements of types KICKER, HKICKER, or VKICKER.

Both commands have the same attributes:

STATUS If this flag is true (on), the selected elements are activated. Active orbit monitor readings will be considered, and active correctors will be allowed to vary in subsequent correction commands. Inactive elements will be ignored subsequently.

MODE This name may take the value ALL, in which case all correctors or monitors will be set to active or inactive. For the USEKICK command two more values are possible:

USED Changes all correctors which have been set to a non-zero value by a preceding correction.

UNUSED Changes all correctors which have never been set by a preceding correction.

The range selection is described in Section 4.7. One may do two iterations with disjoint sets of correctors as follows:

```
USE,...           ! set working beam line
...              ! define imperfections
CORRECT,NCORR=32  ! use 32 correctors
USEKICK,OFF,USED  ! deactivate used correctors
CORRECT,NCORR=32  ! uses different set of correctors
```

Forcing reuse of the same set is achieved thus:

```
USE,...           ! set working beam line
...              ! define imperfections
CORRECT,NCORR=32  ! use 32 correctors
USEKICK,OFF,UNUSED ! deactivate unused correctors
CORRECT,NCORR=32  ! uses same set of correctors
```

9.5 MICADO Statement

The command MICADO assumes that the closed orbit of the machine is known, and that the table of monitor readings (and optionally of dispersion readings) exists. These can be the result of previous GETORBIT and GETDISP commands.

```
MICADO, ERROR=real, NCORR=integer [,C2LIST] [,M1LIST]
      [,DXWEIGHT=real] [,DYWEIGHT=real] [PLANE=letter]
```

It causes closed orbit correction by the MICADO method [2]. The program assumes that all correctors are available for use, but it limits its choice to the most effective NCORR correctors. The monitors are not used for orbit or dispersion readings, if flagged as inactive in the corresponding TFS files read. The attributes have the following meaning:

ERROR The desired accuracy of the correction (r.m.s. error of the closed orbit, default: 0 m).

NCORR The maximum number of correctors to be used (correctors are selected by the program, default: all).

DXWEIGHT The weight for horizontal dispersion correction. At present this should be left at zero to omit dispersion correction in the horizontal plane.

DYWEIGHT The weight for vertical dispersion correction. If this weight is non-zero, an attempt is made to correct the vertical dispersion.

PLANE If this attribute is X, only the horizontal correction is made; if it is Y, only the vertical correction is made. In all other cases both planes are corrected.

Two attributes affect the printing of tables:

C2LIST Corrector settings after correction,

M1LIST Monitor readings before correction,

Both have the same possible values:

NONE Print summary only for this table (default),

USED Print table of all used correctors or monitors, plus summary,

ALL Print table of all correctors or monitors, plus summary.

Example:

```
MICADO, ERROR=1.E-4, C2LIST, M1LIST
```

9.6 CORRECT Statement

The CORRECT statement makes a complete closed orbit correction by the MICADO method, using the *computed* values at the monitors:

```
CORRECT, NCORR=integer, ERROR=value, ITERATE=integer, &
      C1LIST, C2LIST, M1LIST, M2LIST &
      [,DXWEIGHT=real] [,DYWEIGHT=real] [PLANE=letter]
```

The program always assumes that all correctors and monitors are available for use, but at each iteration it limits its choice to NCORR most effective correctors. The attributes have the following meaning:

NCORR The maximum number of correctors to be used (correctors are selected by the program, default: all). At each iteration the program may select a different set of NCORR correctors, thus using more than NCORR correctors in total.

ERROR The desired accuracy of the correction (r.m.s. error of the closed orbit, default: 0m).

ITERATE The number of iterations to be made on the non-linear problem (default: 1). If ITERATE=0, the monitor and corrector tables are printed, but no correction is made.

DXWEIGHT The weight for horizontal dispersion correction. At present this should be left at zero to omit dispersion correction in the horizontal plane.

DYWEIGHT The weight for vertical dispersion correction. If this weight is non-zero, an attempt is made to correct the vertical dispersion.

PLANE If this attribute is X, only the horizontal correction is made; if it is Y, only the vertical correction is made. In all other cases both planes are corrected.

Four attributes affect the printing of tables:

C1LIST Corrector settings before correction,

C2LIST Corrector settings after correction,

M1LIST Monitor readings before correction,

M2LIST Monitor readings after correction.

All four have the same possible values:

NONE Print summary only for this table (default),

USED Print table of all used correctors or monitors, plus summary,

ALL Print table of all correctors or monitors, plus summary.

Example:

CORRECT,ERROR=1.E-4,NCORR=100,ITERATE=3,C2LIST,M1LIST

Chapter 10

Plotting and Tabulating

Table 10.1: Plotting Commands

Name	Function	Section
SETPLOT	Set plot options	10.1
RESPLOT	Set default plot options	10.1
PLOT	Build a plot	10.2
STRING	Define a table expression	10.5
TABLE	Print selected columns of a table	10.5

10.1 SETPLOT and RESPLOT Commands

The SETPLOT command allows to specify parameters common to all subsequent plots:

```
SETPLOT, FONT=integer,LWIDTH=real,XSIZE=real,YSIZE=real,&  
        ASCALE=real,LSCALE=real,SSCALE=real,TSCALE=real,POST=integer
```

The RESPLOT command serves to reinstall the defaults for the SETPLOT command parameters:

```
RESPLOT
```

It is defined in the command dictionary as

```
RESPLOT: SETPLOT,FONT=1,LWIDTH=1.,XSIZE=0.,YSIZE=0.,&  
        ASCALE=1.,LSCALE=1.,SSCALE=1.,TSCALE=1.
```

To change the plot size, a SETPLOT command specifying both XSIZE and YSIZE must be given before the first PLOT command. These parameters must *both* be entered to be effective. They allow to make long plots on the Versatec plotter. The values (0.,0.) give the default on any device. See GXPLLOT manual for details [19].

XSIZE Horizontal size of plot on the hard copy output device [cm],

YSIZE Vertical size of plot on the hard copy output device [cm].

They may be set to almost the page size for A4 format output, i. e. to 20 and 28 cm for portrait mode, or to 28 and 20 cm for landscape mode.

All other SETPLOT options become effective when set, and remain so until redefined.

FONT Character font, (default: 1, Roman). Other fonts exist, but are normally specific to the GKS-package used (see Appendix of the GKSGRAL manual [1]), and device-dependent. Note that postscript characters will not be plotted correctly on the screen by the Higz plotting system since Higz uses for that the “hardware” character set which cannot be rotated. On a postscript output the postscript fonts will look much better than the (default) vector fonts.

POST This option accepts two values:

- 1 Write a postscript file (one file for all frames),
- 2 Write encapsulated postscript files (one file per frame).

LWIDTH A line width scale factor applying to curves (device-dependent). Accepted values are normally 1, 3, and 5 (default: 1).

ASCALE Scale factor for the size of the plotted characters of the curve annotation,

LSCALE Scale factor for the size of the plotted characters of the axis labels (scale),

SSCALE Scale factor for the size of the plotted symbol at the point positions,

RSCALE Scale factor for the size of the plotted characters of the axis text.

The default is 1.0 for all four scale factors.

10.2 PLOT Command

The PLOT command produces one or several frames (pictures) at a time:

```
PLOT, VAXIS=name, VAXIS1=name, VAXIS2=name, VAXIS3=name, VAXIS4=name, &
    HAXIS=name, BARS=integer, STYLE=integer, SYMBOL=integer, &
    MAXPLOT=integer, SORT=logical, SPLINE=logical, MULTIPLE=logical, &
    FFT=logical, HMIN=real, HMAX=real, VMIN=real, VMAX=real, &
    TABLE=name, TITLE=string, PARAM=name, RANGE=range, DELTAP=real, &
    PARTICLE=integer, TURNS=integer
```

Its parameters are only valid for this one command:

VAXIS Up to five names of dependent (vertical) variables to be plotted *on the same axis (!)*. If more than one vertical axis is required, one may enter up to five variable names per axis via one of the parameters VAXIS1, VAXIS2, VAXIS3, VAXIS4. If VAXIS is specified, the others are ignored. Example:

```
PLOT, HAXIS=S, VAXIS=BETX, BETY
```

gives a plot with BETX and BETY both referring to the same axis, whereas

```
PLOT, HAXIS=S, VAXIS1=BETX, VAXIS2=BETY
```

provides separate axes (with different scales, normally) for the two variables.

HAXIS The name of the independent (horizontal) variable.

BARS Connect each point to the horizontal axis by a vertical line.

STYLE Line style selection for all curves. The default value is 1 (solid line for all curves). Further possible values are: 0 (no connecting line, i.e. nothing at all if no symbol plotted, see below), 2 (dashed), 3 (dotted), 4 (dot-dashed), and others mentioned in the GKSGRAL manual. A special value is 100: when chosen, it provides line styles 1, 2, 3, 4, 1, 2, etc. for successive curves in the same frame.

SYMBOL This option will be mainly used for track plots, and allows to plot symbols at the points which are plotted. These can be one of the following: “hardware” symbols, selected by the values 1 (dot), 2 (+), 3 (*), 4 (o), or 5 (x); the turn number (curve number for tables TWISS and TUNES) in each picture, selected by value 100; an upper case character A...Z (201 to 226), lower case character (301 to 326), or a digit 0...9, selected by 400 to 409. 200 and 300 act similar to 100: 200 makes turns (curves) loop over A...Z, 300 over a...z. The SSCALE factor on the SETPLOT command applies to all of these except the “hardware” symbols which are of fixed size.

MAXPLOT Defines the maximum number of pictures (frames) to be plotted with one command (default: 10). This is a protection against user mistakes mainly (e.g. plotting one frame per s value around the whole machine).

SORT Logical flag, sorts the points in ascending values of the abscissa.

SPLINE Logical flag, connects curve points by a third order natural spline rather than by straight line segments (for α , β , μ and D the correct formulae are used for interpolation if plotted against s).

MULTIPLE Logical flag, only for TRACK table plots: put all particles into one frame.

FFT Plot the result of the Fourier analysis of the picture, rather than the picture itself (not yet implemented).

TABLE Name of the table to be plotted. If omitted, the last generated table will be used.

TITLE Text string to be put on top of the picture (default: run title as read on TITLE command). A minor modification in Version 8.19, yet very useful for a better documentation of MAD pictures, is the capability of *adding* the main job TITLE to the picture when it is different from the local title defined by PLOT, ..., TITLE=string. This enables to include additional information concerning a specific plot, while keeping the generic optics description typically contained in the main TITLE.

PARAM Frame selection parameter: S or DELTAP for the TWISS table, PARTICLE or TURNS for the TRACK table.

RANGE Machine range to be used (see Section 4.7).

DELTAP Values of DELTAP to be used (as on TWISS command).

PARTICLE Particle number.

TURN Turn number.

FILENAME The file name string is stored and used to name output files. Assuming FILENAME="first", a postscript file will have the name first.ps, while encapsulated postscript files will be named first01.eps, first02.eps, etc.

The axis ranges are specified by lower and upper values. If neither value is given, scaling is fully automatic, if both are given, they are used (possibly rounded). If only the lower limit is given as zero, and all corresponding curves lie entirely above or below zero, the scale starts or ends at zero. If only the upper limit is given as zero, the scale is symmetric to zero.

HMIN Lower limit of the abscissa.

HMAX Upper limit of the abscissa.

VMIN Up to four lower limits for each of the four vertical axes.

VMAX Up to four upper limits for each of the four vertical axes.

10.3 Axis and frame selection

All variables from the TWISS table, the TUNES table, and the TRACK table can be plotted on the horizontal or vertical axis. For a list, see Sections 1.5, 1.5.5, and 1.5.6. In addition, the variables RBETX, RBETY, RBXMAX, and RBYMAX can be requested. They stand for the square root of the variables behind the 'R'.

The variables S and DELTAP in the TWISS table, and the variables PARTICLE and TURNS in the TRACK table are used as frame parameters PARAM, either automatically or user-driven. This means that for each value of this variable a new picture is plotted, as described below for the TWISS table.

10.3.1 TWISS table

Depending on the user specification when calculating the table this will normally contain Twiss parameter values for a number of S values (around the ring), and for one or more DELTAP values (see Section 1.5). The table may therefore be regarded as a “matrix” with S as row index, DELTAP as column index, and each “element” consisting of all Twiss values for a given pair (S, DELTAP), both S and DELTAP included in these values.

A subset of the table can be chosen by specifying a RANGE for S, and a list of values for DELTAP. If no S range is given, all S values are selected (!), and similarly for DELTAP: if no values given, all values are used.

Out of this (sub-)matrix the user can now select variables to be plotted. Let us suppose that the user always chooses BETY as ordinate. If S is abscissa, BETY will be plotted against S with DELTAP as “frame parameter” (PARAM), i.e. for each selected value of DELTAP there will be a new plot frame. Similarly for DELTAP as abscissa: in this case, BETY will be plotted against DELTAP for each valid S, giving a new plot frame each time (this might be quite a lot if the user makes a mistake; in order to avoid excessive output, the number of frames per plot command will be limited, with an option MAXPLOT to explicitly changing this limit).

In the case that neither S nor DELTAP is horizontal axis (e.g. plotting BETY against BETX, whatever this may yield), the user has to specify in addition what his frame parameter is. For each value of this frame parameter (S or DELTAP) the values in the corresponding row or column will be plotted in a separate frame. The default is DELTAP. The value of PARAM is ignored if HAXIS=S or DELTAP, i.e. the “other” one is chosen automatically (if HAXIS=DELTA, then S will act as frame parameter).

Examples:

```
PLOT,HAXIS=S,VAXIS=BETY
```

produces n plots (for the n DELTAP values in the table) with one curve each showing the dependence of BETY on S.

```
PLOT,HAXIS=S,VAXIS=BETY,DELTAP=0.:0.01:0.01
```

produces two plots of this type, for DELTAP=0. and 0.01 (provided the DELTAP values exist in the table).

```
PLOT,HAXIS=DELTA,VAXIS=BETY,RANGE=#E
```

produces one plot BETY against DELTAP at the position of the last element in the table.

```
PLOT,HAXIS=DELTA,VAXIS=BETY
```

will produce 10 plots (default value of MAXPLOT) for the first 10 S values in the table.

```
PLOT,HAXIS=BETX,VAXIS=BETY,PARAM=DELTA,DELTAP=0.
```

will produce one plot (at DELTAP=0), the curve points being (BETX,BETY) at all S values. Here, PARAM=DELTA may be omitted since this is the default.

```
PLOT,HAXIS=BETX,VAXIS=BETY,PARAM=S,RANGE=#E
```

will produce one plot at the last S value, with (BETX,BETY) points for all DELTAP values in the table.

10.3.2 TRACK table

The TRACK table (see 1.5.6) has the same structure as the TWISS table, with PARTICLE as row index, and TURNS as column index each of which can therefore be chosen as frame selection parameter. The default parameter is PARTICLE. The value of PARAM is ignored if HAXIS=PARTICLE or TURNS, i.e. the "other" one is chosen automatically (if HAXIS=TURNS, then PARTICLE will act as frame parameter). The option MULTIPLE allows to plot the points of all particles in the same frame.

```
PLOT,HAXIS=X,VAXIS=PX,PARTICLE=1,3,5,MULTIPLE
```

will produce one plot of PX versus X for particles 1, 3, and 5.

10.3.3 TUNES table

The TUNES table (see Section 1.5.5) does not have frame-parameters. If PARAM is specified, it will be ignored.

```
PLOT, TABLE=TUNES,HAXIS=DELTAP,VAXIS=QX
```

will produce a plot of the x-tune versus DELTAP.

10.4 Tune Grid

MAD permits to plot an tune grid if the following two conditions are both true:

- A plot of Q_x versus Q_y or of Q_y versus Q_x is requested,
- At least one selection criterion has been entered on the PLOT command.

MAD then plots all resonance lines that are inside the plot window, and which fulfil conditions of the form

$$k_x Q_x + k_y Q_y + k_s Q_s = ps,$$

where k_x , k_y , k_s , and p are integers, and s is the super-periodicity (integer). Furthermore, the user must impose at least one condition of the form

$$F(k_x, k_y, k_s) = n_1 : n_2 : n_3,$$

where F is an arithmetic function k_x , k_y , k_s , and integers only, and only the operators "+", "-", "*", and "/" are allowed. n_1 , n_2 , and n_3 are literal integers, and $n_1 : n_2 : n_3$ stands for a range. It is interpreted like the range in a FORTRAN DO loop, i. e. it runs through the values $n_1, n_1 + n_2, \dots, n_3$.

Examples:

$$k_s = -2 : 2 : 2 \Rightarrow k_s = -2, 0, +2; \quad k_x + k_y = 0 : 0 : 1 \Rightarrow k_x = -k_y.$$

Note that integer arithmetic is used, therefore the condition

$$2 * (k_x + k_y/2) * (k_s - 1)/2 = -3 : 3 : 2$$

will do something weird.

Criteria are specified via the following new attributes of the PLOT command:

PLOT, ..., NTMAX=integer, QCONDi=string, integer1, integer2, integer3

with the meaning:

NTMAX Maximum for $|k_x| + |k_y| + |k_s|$ (default: 20). The absolute upper limits are $|k_x| + |k_y| + |k_s| \leq 20$, and $|k_s| \leq 10$.

QCONDi The function F , encoded as a quoted string, and the three integers n_1, n_2, n_3 for the selection criterion (defaults: " ", 0, 0, 1). The digit i may run from 1 to 10 for up to 10 conditions. When several selection criteria are entered, they must all be true for a line to be plotted.

Example:

```
PLOT, HAXIS=QX, VAXIS=QY, &
    HMIN=21., HMAX=23., VMIN=18., VMAX=20., &
    NTMAX=5, QS=0.2, &
    QCOND1="KX", 1, 1, &
    QCOND2="KS", -3, 3, &
    QCOND3="KX+KY", -10, 10, 10
```

will plot Q_y versus Q_x , and superpose the lines

$$k_x Q_x = ps, k_x = 1; \quad k_s Q_s = ps, k_s = -3, -2, -2, 0, 1, 2, 3;$$

$$k_x Q_x + k_y Q_y = ps, k_x + k_y = -10, 0, 10.$$

The line style reflects the importance of the resonance. The thickness varies from $|k_x| + |k_y| = 1$ (thickest) to $|k_x| + |k_y| = 5$ (thinnest) and remains constant from there on. Lines with $k_s = 0$ are solid, with $|k_s| = 1$ dashed, with $|k_s| = 2$ dot-dashed, and with $|k_s| > 2$ dotted.

10.5 TABLE and STRING Commands

Any table created by MAD can be tabulated on the PRINT file by the command

```
TABLE, NAME=name, COLUMNS=name{, name}, SUM=name{, name}
```

It has the attributes

NAME Name of the table to be tabulated. By default the last table created is used.

COLUMNS Up to 50 names whose values shall be tabulated. These may be any of

- Name of a column in the table to be listed.
- Name of a descriptor in the table to be listed.
- Name of a global variable (parameter, see Section 2.8.1).
- Name of a global STRING expression (see below).
- Name of an expression predefined for the table.

SUM Up to 50 name of columns to be summed. The TABLE command will compute (but not print) and sum up each name for each table row, and print the total after the table. This may be used to compute approximate values for synchrotron integrals.

A STRING expression is defined as follows:

```
label:STRING, "string"
```

where label is the name given to the expression, and string (enclosed in single or double quotes) must contain a valid expression. Operands in such an expression may include:

- Name of a column in the table to be listed.
- Name of a descriptor in the table to be listed.
- Name of a global variable.
- Name of an attribute of a command or definition.

A string may be redefined by repeating the STRING command.

Example:

```
OPTICS, ...
SIGX:STRING,"SQRT(BEAM[EX]*BETX)"
CHROMX:STRING,"SQRT(BETX*BEAM[EX] + (DX*DELTAP)\(\hat{\ }2)\)"
TABLE,NAME=OPTICS,COLUMN=K1L,K2L,BETX,SIGX,SUM=CHROMX
```

10.5.1 Plotting Composite Values

The mechanisms from the preceding subsection can also be used for plotting. One may for example plot the beam size after a NORMAL command by

```
NORMAL, ...
SIGX:STRING,&
"SQRT(BEAM[EX]*(E11\(\hat{\ }2)+E21\(\hat{\ }2)+
BEAM[EY]*E31\(\hat{\ }2)+E42\(\hat{\ }2)+
BEAM[ET]*(E51\(\hat{\ }2)+E61\(\hat{\ }2)))"
SIGY:STRING,&
"SQRT(BEAM[EX]*(E13\(\hat{\ }2)+E23\(\hat{\ }2)+
BEAM[EY]*E33\(\hat{\ }2)+E43\(\hat{\ }2)+
BEAM[ET]*(E53\(\hat{\ }2)+E63\(\hat{\ }2)))"
SIGT:STRING,&
"SQRT(BEAM[EX]*(E15\(\hat{\ }2)+E25\(\hat{\ }2)+
BEAM[EY]*E35\(\hat{\ }2)+E45\(\hat{\ }2)+
BEAM[ET]*(E55\(\hat{\ }2)+E65\(\hat{\ }2)))"
PLOT, TABLE=EIGEN, HAXIS=S, VAXIS=SIGX, SIGY, SIGT
```

If desired, such expressions can be predefined and attached to a given table type. Such predefinitions would be placed in the MAD command dictionary:

```
T\_TWISS:KEYWORD,PR=3,SP=2,&
  GAMX=(S(3),=3\(\backslashslash\)*""),GAMY=(S(3),=3\(\backslashslash\)*""),&
  SIGX=(S(3),=3\(\backslashslash\)*""),SIGY=(S(3),=3\(\backslashslash\)*""),&
T\_TWISS:T\_TWISS,&
  GAMX="(1+ALFX\(\hat{\ }2)/BETX",GAMY="(1+ALFY\(\hat{\ }2)/BETY",&
  SIGX="SQRT(BETX*BEAM[EX])",SIGY="SQRT(BETY*BEAM[EY])"
```

The name of the definition must be the table type, prefixed by T_. Note that the strings in the first definition must be dimensioned at 3, thus allowing to enter information like dimension and axis label information to the second definition.

Chapter 11

HARMON Module

Table 11.1: HARMON Commands

Name	Function	Section
HARMON	Set up HARMON tables	11.1
HCELL	Minimise aberrations	11.4
HCHROMATICITY	Calculate chromaticity	11.2
HFUNCTION	Calculate harmonic functions	11.4
HRESONANCE	Calculate resonance effects	11.3
HTUNE	Tune chromaticity	11.2
HWEIGHT	Set matching weights for resonances	11.4
HVARY	Make a multipole variable	11.4
ENDHARM	Quit HARMON program	11.1

The HARMON program by M. H. Donald [15] can be called directly from MAD.¹ Before calling HARMON, a beam line must be selected by means of a USE command (see Section 5.1). The HARMON command sets up internal tables and defines the emittances to be used for the analysis. The internal tables are released and HARMON mode is terminated by the ENDHARM command. The available commands are listed in Table 11.1.

It is important to note that HARMON *does not* consider the following:

- Coupling effects,
- Excitation errors (EFIELD, EFCOMP),
- Alignment errors (EALIGN).

The integral formalism used does not allow to take these effects into account, and users should be aware of this.

11.1 Activating and Deactivating HARMON

The HARMON module is activated and the relevant tables are initialised by the command

```
HARMON,FX=real,FY=real,FE=real
```

with the attributes:

FX Number of horizontal standard deviations to be taken,

¹Adapted to MAD by D. Schofield (1983) and J.M. Veuillen (1988); rewritten extensively 1989.

FY Number of vertical standard deviations to be taken,

FE Number of longitudinal standard deviations to be taken.

If one of these values is omitted or 0, the program uses 1. The previous attribute OBSERVE no longer exists. If the machine has a symmetry, a second observation point is given at the symmetry point.

The HARMON module is deactivated and its tables are released by the ENDHARM command

ENDHARM

An example is given at the end of this Chapter.

11.2 Chromaticity Calculation or Tuning

The chromaticity is calculated by a command without attributes:

HCHROMATICITY

The chromaticity is tuned to a desired value by the HTUNE command

HTUNE,QX'=real,QY'=real,TOLERANCE=real

The desired values are entered as $dQ/d\delta$ with the attributes

QX' Desired horizontal chromaticity (default: 0),

QY' Desired vertical chromaticity (default: 0).

TOLERANCE Tolerance desired (default: 0, i.e. as good as feasible).

The multipoles to be varied are specified with a HVARY command (see Section 11.4). Example:

HVARY,SF[K2],STEP=0.001

HVARY,SD[K2],STEP=0.001

HVARY,SFK2,STEP=0.0001

HTUNE,QX'=0.0,QY'=0.0

11.3 Resonance Calculations

The resonance effects are computed by the HRESONANCE command

HRESONANCE,ORDER=integer,DISP=logical

The order of the resonance is selected by the attribute

ORDER Order of the resonance. This causes computation of the resonances

$$N_1 Q_x + N_2 Q_y = p_1, \quad N_1 Q_x - N_2 Q_y = p_2,$$

with integer p_1, p_2 , and the condition $N_1 + N_2 = \text{ORDER}$,

DISP If set, the dispersion is included in the computation.

Contrary to previous versions of HARMON, all sextupole components are considered, i.e. all elements of the types RBEND, SBEND, SEXTUPOLE, MULTIPOLE. Example:

HRESONANCE,ORDER=3,DISP

11.4 Distortion Functions

The HVARY command makes a multipole component variable:

```
HVARY,NAME=variable,STEP=real,LOWER=real,UPPER=real
```

The command has the following attributes:

NAME The name of a variable to be varied. It must be the sextupole, octupole or decapole component of an element having one of the types SBEND, RBEND, SEXTUPOLE, or MULTIPOLE.

STEP An initial step size to be used in matching (required),

LOWER Lower limit for the multipole strength,

UPPER Upper limit for the multipole strength.

The units for STEP, LOWER, and UPPER are the same as in the corresponding element definition. Examples:

```
HVARY,NAME=SF1[K2],STEP=0.001,LOWER=-1.0,UPPER=0.0 ! Sextupole
```

```
HVARY,NAME=MF1[K3L],STEP=0.001,LOWER=0.0,UPPER=1.0 ! Multipole
```

The HFUNCTIONS command

```
HFUNCTIONS
```

computes the harmonic functions as defined in [15]. For the chromaticities and dispersions, *but only for these*, the effects of octupoles and decapoles are also considered and may be matched by varying those components. The HCELL command

```
HCELL, TOLERANCE=real, CALLS=integer, QX'=real, QY'=real, &
      QX''=real, QY''=real, QX'''=real, QY'''=real, &
      DQXDEX=real, DQYDEY=real, DQYDEX=real, &
      DX'I=real, DX''I=real, BX'I=real, BY'I=real, RXI=real, RYI=real, &
      DX'S=real, DX''S=real, BX'S=real, BY'S=real, RXS=real, RYS=real
```

adjusts the following quantities to the values entered:

Tolerance Tolerance for the penalty function constructed.

CALLS Limit for the number of calls to the penalty function.

QX' First-order horizontal chromaticity.

QY' First-order vertical chromaticity.

QX'' Second-order horizontal chromaticity.

QY'' Second-order vertical chromaticity.

QX''' Third-order horizontal chromaticity.

QY''' Third-order vertical chromaticity.

DQXDEX Derivative of horizontal tune with horizontal emittance.

DQYDEY Derivative of vertical tune with vertical emittance.

DQYDEX Derivative of vertical tune with horizontal emittance.

DX'I Second-order horizontal dispersion at interaction point.

DX''I Third-order horizontal dispersion at interaction point.

BX'I Variation of β_x with energy at interaction point.

BY'I Variation of β_y with energy at interaction point.

RXI Contributions of horizontal resonances at interaction point.

RYI Contributions of vertical resonances at interaction point.

DX'S Second-order horizontal dispersion at symmetry point.

DX''S Third-order horizontal dispersion at symmetry point.

BX'S Variation of β_x with energy at symmetry point.

BY'S Variation of β_y with energy at symmetry point.

RXS Contributions of horizontal resonances at symmetry point.

RYS Contributions of vertical resonances at symmetry point.

The chromaticities and dispersions of all orders calculated in HARMON contain the effects of multipoles up to the decapole. These multipole strengths may all be adjusted to minimise the distortions. All quantities are fitted in a least-squares sense. The sextupole components to be varied must be specified in a HVARY command. The default matching weights are one for the first-order chromaticities, and zero for all other quantities. Different weights can be set before calling HCELL with the HWEIGHT command

```
HWEIGHT,QX'=real,QY'=real,QX''=real,QY''=real,QX'''=real,QY'''=real,&
      DQXDEX=real,DQYDEY=real,DQYDEX=real,&
      DX'I=real,DX''I=real,BX'I=real,BY'I=real,RXI=real,RYI=real,&
      DX'S=real,DX''S=real,BX'S=real,BY'S=real,RXS=real,RYS=real
```

```
QX' QX'' QX''' QY' QY'' QY''' DQXDEX DQYDEY DQYDEX DX'I DY'I DX'S DY'S BX'I BY'I BX'S BY'S
RXI RYI RXS RYS
```

11.5 Example for a HARMON Sequence

```
USE,      OCT,SUPER=4,SYMM
BEAM,     PARTICLE=ELECTRON,ENERGY=50,&
          EX=0.0645E-6,EY=0.03225E-6,SIGE=1.25E-3
!
HARMON,   FX=10,FY=10,FE=7
!
HCHROMATICITY
HVARY,    NAME=SF1,STEP=0.001 ! Chromaticity sextupoles
HVARY,    NAME=SD1,STEP=0.001
HTUNE,    QX'=0,QY'=0
!
HRESONANCE,ORDER=3
!
HWEIGHT,  QX'''=1.0,QY'''=1.0,QX''''=10.0,QY''''=10.0
HVARY,    SF2,STEP=0.001,LOWER=0.,UPPER=1.
```



```
HVARY,    SD2,STEP=0.001,LOWER=-1.,UPPER=0.
HVARY,    SF3,STEP=0.001,LOWER=0.,UPPER=1.
HVARY,    SD3,STEP=0.001,LOWER=-1.,UPPER=0.
HCELL     ! Chromaticities minimised to all three orders
!
HRESONANCE
ENDHARM
```

Chapter 12

Matching Module

Table 12.1: Matching Commands

Name	Function	Section
CELL	Initialise cell matching	12.1
MATCH	Initialise insertion matching	12.1
WEIGHT	Set matching weights	12.2
CONSTRAINT	Impose matching constraint	12.3
COUPLE	Impose periodicity between two points	12.3
GLOBAL	Constraint on global quantities	12.4
GWEIGHT	Weights for Global Constraints	12.4
VARY	Vary parameter	12.2
FIX	Fix parameter value	12.2
RMATRIX	Constraint on linear matrix	12.3
TMATRIX	Constraint on second-order terms	12.3
LEVEL	Set print level	12.5
LMDIF	Minimisation by gradient method	12.6
MIGRAD	Minimisation by gradient method	12.6
SIMPLEX	Minimisation by simplex method	12.6
ENDMATCH	Leave matching mode	12.1

Before a match operation a beam line must be selected by means of a USE command (see Section 5.1). Matching of this line is then initiated by a CELL or MATCH command. From either of these commands to the corresponding ENDMATCH command MAD recognises the matching commands listed in Table 12.1. For a mathematical description of the minimisation procedures see [23]. In particular one may do the following:

- Define parameter(s) to be varied,
- Couple and/or set parameter values,
- Define constraints,
- Select desired printout detail,
- Match by different methods.

The matching commands are described in detail below. Some other commands can also be issued during matching.

12.1 Activation and Deactivation of the Matching Module

Before matching a beam line must be selected by means of a USE command (see Section 5.1). The type of match desired is then defined by entering a CELL or MATCH command.

12.1.1 Matching a Periodic Cell

In the first mode, called cell matching, a periodic cell is adjusted. The periodicity is enforced exactly, and constraints are fulfilled in the least squares sense. Cell matching mode is initiated by the CELL command:

```
CELL, DELTAP=real, ORBIT
```

It has two attributes:

DELTAP The value of the momentum error $\Delta p/p_0c$ for which the match should be performed (default: 0).

ORBIT If this flag is true, the closed orbit is also matched.

Examples:

```
! Match a simple periodic cell
USE, PERIOD=OCTANT, RANGE=CELL1
CELL, ORBIT
!
! Match a symmetric and periodic cell with repetitions
USE, HALFCELL, SYMM, SUPER=5
CELL
```

12.1.2 Insertion Matching

In the second mode, called insertion matching, a beam line is matched with given initial values for the optical functions. Constraints may be imposed in other places, i.e. intermediate or end values can be requested. In this case the initial values are assumed as exact, and constraints are fulfilled in the least squares sense. The insertion matching mode is initiated by the MATCH command. In the simplest form the initial values for the optical functions are taken from the periodic solution of another beam line. In this case, all of $(\beta_x, \beta_y, \alpha_x, \alpha_y, D_x, D_y, D_{p_x}, D_{p_y})$ are transmitted to be used as initial values. If there is a constraint on the orbit, the values (x, p_x, y, p_y) of the orbit are also transmitted. Such a condition is entered in the form

```
MATCH, LINE=beam-line, MUX=real, MUY=real, DELTAP=real, ORBIT
```

The initial phases angles may be specified by:

MUX The initial horizontal phase μ_x ,

MUY The initial vertical phase μ_y .

DELTAP The value of the momentum error $\Delta E/p_0c$ for which the match should be performed (default: 0).

ORBIT If this flag is true, the closed orbit is also matched.

Example:

```

CELL1:  LINE=(...)
INSERT: LINE=(...)
        USE,INSERT
        MATCH,LINE=CELL1,MUX=9.345,MUY=9.876,ORBIT

```

This matches the beam line INSERT. Initial conditions are given by the periodic solution for the beam line CELL1.

It is also possible to enter numerical initial values. The MATCH command then has the form

```

MATCH, BETX=real,ALFX=real,MUX=real,&
      BETY=real,ALFY=real,MUY=real,&
      X=real,PX=real,Y=real,PY=real,&
      DX=real,DY=real,DPX=real,DPY=real,&
      DELTAP=real,ORBIT

```

It accepts as attributes the linear lattice functions listed in Section 1.5.3) and the orbit coordinates listed in Section 1.5.1. Omitted initial values are assumed to be zero. This implies that at least BETX and BETY are required to obtain physically meaningful results. Example:

```

USE,    INSERT
MATCH,  BETX=1.6,BETY=0.1

```

This matches the beam line INSERT with given initial values. Values not specified are set to zero.

The end conditions of a previous beam line can finally be transmitted via a SAVEBETA command (see also section 5.7.3):

```

USE,LINE1
SAVEBETA,LABEL=XYZ,PLACE=#E
TWISS
USE,LINE2
MATCH,BETA0=XYZ

```

This example transmits all values (including phases) as initial values. The orbit is used only if the flag ORBIT is entered. Some values may also be changed explicitly:

```

USE,LINE1
SAVEBETA,LABEL=XYZ,PLACE=#E
TWISS
USE,LINE2
MATCH,BETA0=XYZ,MUX=1.234,MUY=4.321

```

will overwrite the phases taken from the previous line.

12.1.3 End of Matching Run

The ENDMATCH command terminates the matching section and deletes all tables related to a tracking run:

```

ENDMATCH

```

12.2 Variable Parameters

A parameter to be varied is specified by the VARY command:

```

VARY,NAME=variable,STEP=real,LOWER=real,UPPER=real

```

It has four attributes:

NAME The name of the parameter or attribute to be varied (see Section 2.4.10),

STEP The approximate initial step size for varying the parameter. If the step is not entered, MAD tries to find a reasonable step, but this may not always work.

LOWER Lower limit for the parameter (optional),

UPPER Upper limit for the parameter (optional).

Examples:

```
VARY,PAR1,STEP=1.0E-4      ! vary global parameter PAR1
VARY,QL11[K1],STEP=1.0E-6  ! vary attribute K1 of the QL11
VARY,Q15[K1],STEP=0.0001,LOWER=0.0,UPPER=0.08 ! vary with limits
```

If the upper limit is smaller than the lower limit, the two limits are interchanged. If the current value is outside the range defined by the limits, it is brought back to range. After a matching operation all varied attributes retain their last value. They are never reset to an old value. Parameters cannot be varied if they depend on other parameters. Example:

```
P1:=10.0-P2
```

Here P2 may be varied, but P1 may not. The reason is that MAD is not able to compute P2 in terms of P1.

The command FIX removes a parameter or attribute from the table of variable parameters:

```
FIX,NAME=variable
```

Example:

```
FIX,NAME=QF[K1]
```

12.3 Constraints

12.3.1 Simple Constraints

Simple constraints are imposed by the CONSTRAINT command. It can take three forms. The first form is

```
CONSTRAINT,[RANGE=range,][CLASS=name,][PATTERN=string,]&
          LINE=beam-line,MUX=real,MUY=real
```

Here all of $(\beta_x, \beta_y, \alpha_x, \alpha_y, D_x, D_y, D_{px}, D_{py})$ are constrained in range to the corresponding values of beam-line. If any of the six components of the closed orbit of the beam line is constrained, the command implies setting of the ORBIT flag. The phase advances (μ_x, μ_y) may also be specified on the constraint. If the match succeeds, the part of the beam line from its beginning to place has the specified phase advance. The optical functions are such that beam-line could be inserted in place without a mismatch. If beam-line was defined with formal arguments, an actual argument list must be provided. Normally place refers to a single observation point (see Section 4.7).

The second form of the CONSTRAINT command is

```
CONSTRAINT,[RANGE=range,][CLASS=name,][PATTERN=string,]&
          BETA0=beta0-name,MUX=real,MUY=real
```

Here all of $(\beta_x, \beta_y, \alpha_x, \alpha_y, D_x, D_y, D_{p_x}, D_{p_y})$ are constrained in range to the corresponding values of a pre-calculated BETA0 module (see 5.7.3). If the ORBIT flag is set, then the six components of the closed orbit of the beam line are also constrained. Note that the default weights for T or PT are zero. The phases (μ_x, μ_y) are *not* taken from the BETA0 block. If they are to be constrained, they must be explicitly specified on the CONSTRAINT command. However, it is permitted to specify the phase values from the BETA0 block:

```
CONSTRAINT, PLACE=... , BETA0=DATA, MUX=DATA[MUX], MUY=DATA[MUY]
```

If the match succeeds, the part of the beam line from its beginning to place has the specified phase advance. Normally place refers to a single observation point (see Section 4.7).

The third form

```
CONSTRAINT, [RANGE=range,] [CLASS=name,] [PATTERN=string,] &
    BETX=real, ALFX=real, MUX=real, &
    BETY=real, ALFY=real, MUY=real, &
    X=real, PX=real, Y=real, PY=real, T=real, PT=real, &
    DX=real, DPX=real, DY=real, DPY=real
```

allows one to enter numerical values for any optical function(s) in place. Place may refer to a multiple observation point, as described in Section 4.7. The constraint then applies to all positions within the set. The linear lattice functions listed in Section 1.5.3 can all be constrained. The matching weights can be set by the WEIGHT command (see below). Upper and lower limits can be requested by replacing the equals sign by a less than sign (<) or by a greater than sign (>). Both lower and upper limits may be requested in the same command. If only lower and upper limits are used in a constraint, the corresponding place may be a range like e.g. a beam line. Example:

```
INSERT: LINE=(...)
CELL1: LINE=(...)
      USE, INSERT
      MATCH, LINE=CELL1
      CONSTRAINT, INSERT, DX<2.5, DX>0.0
```

12.3.2 Sub-Period Constraint

It is sometimes desired to make the linear lattice parameters equal in two places without imposing their actual values. This is requested by the COUPLE command

```
COUPLE, RANGE=range, MUX=real, MUY=real
```

where range refers to an observation range (see Section 4.7). MAD tries to adjust the matched beam line such that the functions $(\beta_x, \beta_y, \alpha_x, \alpha_y, D_x, D_y, D_{p_x}, D_{p_y})$ are equal in the two end points of the range. The orbit cannot be constrained by this command. Optionally the phase advances over the range may be specified by means of the attributes MUX and/or MUY. The matching weights are taken from the latest WEIGHT command (see below). Example:

```
M:      MARKER
INSERT: LINE=(... , M, ... , M, ...)
      USE, INSERT
      MATCH, BETX=2, BETY=1
      COUPLE, M[1]/M[2], MUX=0.5
```

12.3.3 Constraints on Transfer map

Often the constraints are desired on the transfer map rather than on the lattice functions. The two commands

```
RMATRIX,RANGE=range {,RM(i,k)=real} {,WEIGHT(i,k)=real}
TMATRIX,RANGE=range {,TM(i,k,l)=real} {,WEIGHT(i,k,l)=real}
```

are provided for this purpose. They have the following attributes:

RANGE Denotes the beginning and end of the range over which the transfer map is to be fitted.

RM(i,k) A real array of dimension (6,6). As many elements of the linear transfer matrix can be constrained.

TM(i,k,l) A real array of dimension (6,6,6). As many elements of the second-order transfer matrix can be constrained.

WEIGHT A real array of dimension (6,6) or (6,6,6) respectively. If given, this specifies the matching weights for the corresponding matrix elements (default: 1).

12.3.4 Matching Weights

The matching procedures try to fulfil the constraints in a least square sense. A penalty function is constructed which is the sum of the squares of all errors, each multiplied by the specified weight. The larger the weight, the more important a constraint becomes. The weights are taken from a table of current values. These are initially set to default values, and may be reset at any time to different values. Values set in this way remain valid until changed again. The **WEIGHT** command

```
WEIGHT, BETX=real,ALFX=real,MUX=real,&
      BETY=real,ALFY=real,MUY=real,&
      X=real,PX=real,Y=real,PY=real,T=real,PT=real,&
      DX=real,DPX=real,DY=real,DPY=real
```

changes the weights for subsequent constraints. The weights are entered with the same name as the linear lattice function to which the weight applies, as listed in Section 1.5. The default weights are listed in Table 12.2. Note that matching T or PT always requires a **WEIGHT** command, since these quantities have zero default weights. Frequently the matching weights serve to select a restricted set of functions to be matched in a **LINE=line** constraint. One may match the dispersion at a given place to the dispersion of a line as in the following example:

```
WEIGHT,      BETX=0.0,BETX=0.0,ALFX=0.0,ALFY=0.0,MUX=0.0,MUY=0.0
CONSTRAINT, #E,LINE=CELL
```

Table 12.2: Default Matching Weights

name	weight	name	weight	name	weight	name	weight
BETX	1.0	ALFX	10.0	MUX	10.0		
BETY	1.0	ALFY	10.0	MUY	10.0		
X	10.0	PX	100.0	Y	10.0	PY	100.0
T	0.0	PT	0.0				
DX	10.0	DPX	100.0	DY	10.0	DPY	100.0

12.4 GLOBAL: Constraints on Global Quantities

In addition to the conventional matching constraints, corresponding to the commands CONSTRAINT, COUPLE, RMATRIX and TMATRIX, most of the global variables can be constrained by the command GLOBALGWEIGHT, having the following syntax:

```
GWEIGHT, Q1=real, Q2=real, Q1'=real, Q2'=real, &
        Q1''=real, Q2''=real, DQ1DE1=real, DQ1DE2=real, DQ2DE2=real, &
        DYNAPFRAC=real, TURNS=real, &
        WXMIN=real, WXMAX=real, WYMIN=real, WYMAX=real, &
        WXYMIN=real, WXYMAX=real, SMEAR=real, &
        TUNX=real, TUNY=real, DTUNE=real, LYAPUNOV=real, &
        XEND=real, PXEND=real, YEND=real, PYEND=real, TEND=real, PTEND=real, &
        FORMULA=real
GLOBAL, Q1=real, Q2=real, Q1'=real, Q2'=real, &
        Q1''=real, Q2''=real, DQ1DE1=real, DQ1DE2=real, DQ2DE2=real, &
        DYNAPFRAC=real, TURNS=real, &
        WXMIN=real, WXMAX=real, WYMIN=real, WYMAX=real, &
        WXYMIN=real, WXYMAX=real, SMEAR=real, &
        TUNX=real, TUNY=real, DTUNE=real, LYAPUNOV=real, &
        XEND=real, PXEND=real, YEND=real, PYEND=real, TEND=real, PTEND=real, &
        FIXPOINT, FORMULA=real, CALL=name
```

Only one GLOBAL constraint can be active within a matching session, corresponding to the last call to the GLOBAL command. The default value of all the attributes is zero, with the exception of DYNAPFRAC (default: 1.0) and TURNS (default: 64). Global matching weights can be (re)set by the GWEIGHT command, having attributes identical to those of GLOBAL (with the exception of FIXPOINT and CALL). All attributes are optional and have the following meaning:

Q1, Q2, Q1', Q2' Enable a correction of tunes and chromaticities (automatically computed by an internal call to a shortened version of STATIC) in presence of magnetic imperfections or misalignments,

Q1'', Q2'', DQ1DE1, DQ1DE2, DQ2DE2 Enable a correction of nonlinear chromaticities and anharmonicities,

DYNAPFRAC, TURNS Enable an optimisation of dynamic aperture or survival time (automatically computed by an internal call to DYNAP): In the case of TURNS, the option FRACMIN = 1 should appear in the last call to DYNAPSTART. Since these variables take only discrete values, the MAD command SYMPLEX should be used for actual minimisation, instead of LMDIF or MIGRAD,

FIXPOINT Flag enabling a fixed point search corresponding to the number of TURNS specified in the last call to DYNAPSTART. This attribute should be used in conjunction with (some or all) the tracking end-values XEND, PXEND, ..., (automatically computed by an internal call to DYNAP) and generates a constraint to the corresponding starting values XSTART, PXSTART, ...: The user should then VARY the starting values.

FORMULA Enables to constrain to a target numerical value any user-defined expression associated with the variable FORMULA (by a previous statement FORMULA=expression),

CALL Name of any user-defined MAD SUBROUTINE to be called before FORMULA is evaluated (the output is then temporarily re-routed to the file CALL).

A speed-up for closed orbit calculation is possible by first calling TWISS or EMIT before entering the matching module, followed by COGUESS, X=X0, PX=PX0, . . . to set an initial guess for the closed orbit. For example, to maximise the dynamic aperture at constant chromaticities in presence of radiation damping, one can use the following commands:

```

EMIT
COGUESS, X=X0, PX=PX0, Y=Y0, PY=PY0, T=T0, PT=PT0
DYNAPSTART, . . . , DAMP
STATIC
MATCH
GLOBAL, Q1'=Q1', Q2'=Q2', DYNAPFRAC
VARY, . . .
SIMPLEX, CALLS=100
ENDMATCH

```

Note that the chromaticities computed by STATIC (and TWISS) are affected by the presence of RF cavities with non-vanishing RF voltage.

The FORMULA + CALL option is very flexible, although not optimised for speed, and can be used to combine multiple constraints on detuning or dynamic aperture with different initial conditions. If the FORMULA contains several quantities to be minimised at the same time, they should be added quadratically with proper user-defined weights: the resulting expression (minus the target FORMULA value) is multiplied by the global FORMULA weight, possibly redefined by GWEIGHT, FORMULA=real, and then added quadratically to any other matching constraint to arrive at the penalty function to be minimised. The CALL option can not be used within a MAD SUBROUTINE.

12.5 Matching Output Level

The LEVEL command sets the output level for matching:

```
LEVEL, LEVEL=integer
```

Recognised level numbers are:

- 0** Minimum printout: Messages and final values only,
- 1** Normal printout: Messages, initial and final values,
- 2** Like LEVEL=1, plus every tenth new minimum found,
- 3** Like LEVEL=2, plus every new minimum found,
- 4** Like LEVEL=3, plus eigenvalues of covariance matrix (MIGRAD method only).

Example:

```
LEVEL, 2
```

12.6 Matching Methods

12.6.1 LMDIF, Gradient Minimisation

The LMDIF command minimises the sum of squares of the constraint functions using their numerical derivatives:

```
LMDIF, CALLS=integer, TOLERANCE=real
```

It is the fastest minimisation method available in MAD. The command has two attributes:

CALLS The maximum number of calls to the penalty function (default: 1000).

TOLERANCE The desired tolerance for the minimum (default: 10^{-6}).

Example:

```
LMDIF,CALLS=2000,TOLERANCE=1.0E-8
```

12.6.2 MIGRAD, Gradient Minimisation

The MIGRAD command minimises the penalty function using its numerical derivatives of the sum of squares:

```
MIGRAD,CALLS=integer,TOLERANCE=real,STRATEGY=1
```

The command has three attributes:

CALLS The maximum number of calls to the penalty function (default: 1000).

TOLERANCE The desired tolerance for the minimum (default: 10^{-6}).

STRATEGY A code for the strategy to be used (default: 1). The user is referred to the MINUIT manual for explanations [23].

Example:

```
MIGRAD,CALLS=2000,TOLERANCE=1.0E-8
```

12.6.3 SIMPLEX, Minimisation by Simplex Method

The SIMPLEX command minimises the penalty function by the simplex method:

```
SIMPLEX,CALLS=integer,TOLERANCE=real
```

Details are given in the description of the MINUIT program [23]. The command has two attributes:

CALLS The maximum number of calls to the penalty function (default: 1000).

TOLERANCE The desired tolerance for the minimum (default: 10^{-6}).

Example:

```
SIMPLEX,CALLS=2000,TOLERANCE=1.0E-8
```

12.7 Matching Examples

12.7.1 Simple Periodic Beam Line

Match a simple cell with given phase advances:

```
QF:    QUADRUPOLE,...
QD:    QUADRUPOLE,...
CELL1: LINE=(...,QF,...,QD,...)
        USE,CELL1
        CELL
        VARY,NAME=QD[K1],STEP=0.01
        VARY,NAME=QF[K1],STEP=0.01
        CONSTRAINT,PLACE=#E,MUX=0.25,MUY=1/6
        MIGRAD,CALLS=2000
        ENDMATCH
```

12.7.2 Insertion Matching

Match an insertion INSERT to go between two different cells CELL1 and CELL2:

```
CELL1:  LINE=(...)
CELL2:  LINE=(...)
INSERT: LINE=(...)
        USE,INSERT
        MATCH,LINE=CELL1
        VARY,...
        CONSTRAINT,PLACE=#E,LINE=CELL2,MUX=...,MUY=...
        SIMPLEX
        MIGRAD
        ENDMATCH
```

Adjust the beam line INSERT such that it has specified values of the optical functions at its beginning and matches the line CELL7 at its end:

```
CELL7:  LINE=(...)
INSERT: LINE=(...)
        USE,INSERT
        MATCH,BETX=1.6,BETY=0.1
        VARY,...
        CONSTRAINT,PLACE=#E,LINE=CELL7
        SIMPLEX
        MIGRAD
        ENDMATCH
```

12.7.3 A Matching Example for LEP, Version 11

Here the end values are required to be exact. This may be achieved by inverting the beam line:

```
! Define beam lines
CELL:  LINE=(QDH,L23,SD,L22,B6,L21,QF,L23,SF,L22,B6,L24,QDH)
MARK:  MARKER
DISS:  LINE=(QS11,L25,BW,L24,QS12,L25,B4,L24,QS13,L25,B4,&
          L24,QS14,L25,B4,L31,QS15,L25,B4,L32,SF2,L23,QS16)
RFS:   LINE=(L5,QS5,L5,QS6,L5,2*(QS7H,MARK,QS7H,L5,QS8,L5))
LOBS:  LINE=(L1,QS1,L2,QS2,L3,QS3,L4,QS4)
INSS:  LINE=(LOBS,RFS,DISS,L21,B6,L22,QDH)
        USE,(-INSS)
!
! Call matching module and define initial values
MATCH,LINE=CELL
!
! Variable parameters
VARY,QS1[K1],STEP=0.01,LOWER=-0.06,UPPER=0.06
VARY,QS2[K1],STEP=0.01,LOWER=-0.06,UPPER=0.06
VARY,QS3[K1],STEP=0.01,LOWER=-0.06,UPPER=0.06
VARY,QS4[K1],STEP=0.001,LOWER=-0.06,UPPER=0.06
VARY,QS5[K1],STEP=0.01,LOWER=-0.035,UPPER=0.035
VARY,QS6[K1],STEP=0.01,LOWER=-0.035,UPPER=0.035
VARY,QS7[K1],STEP=0.01,LOWER=-0.035,UPPER=0.035
```

```

VARY, QS8[K1], STEP=0.01, LOWER=-0.035, UPPER=0.035
VARY, QS11[K1], STEP=0.01, LOWER=-0.035, UPPER=0.035
VARY, QS12[K1], STEP=0.01, LOWER=-0.035, UPPER=0.035
VARY, QS13[K1], STEP=0.01, LOWER=-0.035, UPPER=0.035
VARY, QS14[K1], STEP=0.01, LOWER=-0.035, UPPER=0.035
VARY, QS15[K1], STEP=0.01, LOWER=-0.035, UPPER=0.035
VARY, QS16[K1], STEP=0.01, LOWER=-0.035, UPPER=0.035
!
! Constraints at the interaction point
WEIGHT, BETX=10.0, BETY=50.0, ALFX=10.0, ALFY=10.0,
MUX=10.0, MUY=10.0, DX=10.0, DPX=100.0
CONSTRAINT, #E, BETX=1.6, BETY=0.1, ALFX=0.0, ALFY=0.0,
MUX=1.771875, MUY=2.0125, DX=0.0, DPX=0.0
!
! Constraints at two intermediate points (MARK occurs twice)
CONSTRAINT, MARK, ALFX=0.0, ALFY=0.0, DX=0.0, DPX=0.0
COUPLE, MARK[1]/MARK[2]
!
! Constraint for limits of dispersion in dispersion suppressor
CONSTRAINT, DISS, DX>0, DX<1.25
!
! Perform match
SIMPLEX, CALLS=1000
MIGRAD, CALLS=24000, TOLERANCE=1.0E-9
!
! End of matching procedure
ENDMATCH

```

12.7.4 Examples for Complex Matching Constraints

All initial conditions and constraint values can depend on variable parameters. This feature permits to set up very complex matching conditions. The first example shows how an insertion can be designed which exchanges the values of the β -functions for the two transverse planes in two places M1 and M2:

```

M1:      MARKER
M2:      MARKER
INSERT:  LINE=(... ,M1,... ,M2,... )
          CONSTRAINT, M1, BETX=BETA1, ALFX=ALFA1, BETY=BETA2, ALFY=ALFA2
          CONSTRAINT, M2, BETX=BETA2, ALFX=ALFA2, BETY=BETA1, ALFY=ALFA1
          VARY, BETA1, STEP=0.01
          VARY, BETA2, STEP=0.01
          VARY, ALFA1, STEP=0.01
          VARY, ALFA2, STEP=0.01

```

The trick is to match the lattice functions in M1 to the four variable parameters

BETA1, ALFA1, BETA2, ALFA2

and in M2 to the same parameters in a different order.

The second example is taken from a spin matching problem.

```

! Match towards interaction point
MULT:=(1+SIN THETA)/SIN THETA ! A constant multiplication factor.

```

```

MATCH,LINE=SRF3M

! Lattice functions imposed at interaction point
WEIGHT,BETX=1,BETY=10,DY=30,DPY=300
CONSTRAINT,LORIPS,BETX=1.75,BETY=0.07,ALFX=0,ALFY=0,DY=0

! ***** Spin matching constraints *****
RMATRIX,DBB[1]/LORIPS,RM(3,4)=0,RM(2,1)=AUX
VARY,AUX,STEP=0.01
RMATRIX,B2UP[1]/LORIPS,RM(3,4)=0,RM(2,1)=MULT*AUX

! Quadrupoles to be varied
VARY,KQSOMP,STEP=0.01,LOWER=-0.2,UPPER=0.2
VARY,KQS1MP,STEP=0.01,LOWER=-0.1,UPPER=0.1
VARY,KQS3MP,STEP=0.01,LOWER=-0.1,UPPER=0.1
VARY,KQS4MP,STEP=0.01,LOWER=-0.1,UPPER=0.1
VARY,KQS5AP,STEP=0.01,LOWER=-0.1,UPPER=0.1
VARY,KQS6AP,STEP=0.01,LOWER=-0.1,UPPER=0.1
VARY,KQS5BP,STEP=0.01,LOWER=-0.1,UPPER=0.1
VARY,KQS6BP,STEP=0.01,LOWER=-0.1,UPPER=0.1

! Initiate matching
SIMPLEX,CALLS=1000,TOLE=1E-3
LMDIF,CALLS=6000,TOLE=1E-12
ENDMATCH

```

Here the interesting point lies in the spin matching constraints. The conditions $RM(3,4)=0$ on the two `RMATRIX` commands ensure the proper phase advances for the vertical phase (0.5 and 1.5 respectively). The conditions on $RM(2,1)$ enforce the condition

$$A_{21} = \frac{(1 + \sin \theta)}{\sin \theta} C_{21}.$$

where A and C are the transfer matrices from the centre of either vertical bend to the interaction point. Since a direct coupling between the values of the transfer matrix in two different points is not possible, a freely variable parameter AUX is introduced. The value of C_{21} is matched to AUX , while the value of A_{21} is matched to an expression in AUX .

Chapter 13

Tracking Module

Table 13.1: Tracking Commands

Name	Function	Section
NOISE	Define noise on parameters	13.2
OBSERVE	Define observation point	13.3
START	Define initial coordinates	13.4
RUN	Start tracking	13.6
TSAVE	Save particle positions	13.5
ENDTRACK	Terminate tracking mode	13.1

Before starting to track, the working beam line must be selected by means of a USE command (see Section 5.1). The energy and emittances should be defined by a BEAM command (see Section 5.3). Tracking is then initiated by the TRACK command. From this command to the corresponding ENDTRACK command MAD accepts the tracking statements listed in Table 13.1. Tracking is done in parallel i.e. the coordinates of all particles are transformed at each beam element as it is reached.

13.1 Activation and Deactivation of the Tracking Module

Tracking through a beam line is initiated by the TRACK command. It has the form

```
TRACK [,ONEPASS] [,DAMP] [,QUANTUM]
```

The emittances and σ 's are evaluated from the latest BEAM command. A BEAM command should not occur while the tracking module is active. The TRACK command has three attributes:

ONEPASS This flag tells MAD not to compute the normalisation transformations and to assume that the machine is stable.

The two other flags have only effect if synchrotron radiation is switched on on the BEAM command:

DAMP This flag causes MAD to consider the systematic component (bias) of energy loss by synchrotron radiation.

QUANTUM If this flag is present together with DAMP, it causes MAD to simulate the quantum effects of synchrotron radiation.

The former attributes RFCAVITY and DELTAP are ignored in this version of MAD. The TRACK command must be preceded by a EMIT command, which allows to adjust the average momentum error. The RF frequencies are set by TRACK according to this momentum error set in EMIT.

The synchrotron tune Q_s computed must be non-zero, if E_t or σ_e is used on the BEAM command. Example for a TRACK command:

```
BEAM, PARTICLE=ELECTRON,ENERGY=50,RADIATE,&
      EX=0.0645E-6,EY=0.03225E-6,SIGE=1.25E-3
EMIT,DELTAP=0.01
TRACK,DAMP,QUANTUM
```

The ENDTRACK command terminates the tracking section:

```
ENDTRACK
```

It deletes all information relevant to the current tracking run.

13.2 NOISE Statement

One may define noise to be applied to magnet excitations with the NOISE statement

```
NOISE,VARIABLE=variable,AMPLITUDE(i)=real,FREQUENCY=real,PHASE=real
```

The NOISE statement must be entered after the TRACK, but before the RUN command. It has the effect to apply several sinusoidal variations to the specified parameter. The command has four attributes.

VARIABLE The variable to be affected.

AMPLITUDE(i) The value of the i^{th} noise amplitude A_i .

FREQUENCY(i) The value of the i^{th} noise frequency f_i .

PHASE(i) The value of the i^{th} noise phase ϕ_i .

Before each turn is tracked, the noise is re-evaluated as

$$\Delta A = \sum_{i=1}^N A_i \cos(2\pi(f_i t + \phi_i)).$$

where t is the real time.

13.3 OBSERVE Statement

The statement OBSERVE

```
OBSERVE,PLACE=place,TABLE=table
```

defines a new observation point where a track table is to be generated. It has the attributes:

PLACE The observation point. Only the first occurrence in the machine is used.

TABLE The name to be given to the table.

The OBSERVE statement must be entered after the TRACK, but before the RUN command.

13.4 START Statement

The START command defines the initial coordinates of the particles to be tracked. There may be many START statements, one for each particle. Particles are always started with coordinates relative to the computed closed orbit for the defined energy error. The command format is:

```
START, X=real,PX=real,Y=real,PY=real,T=real,DELTAP=real,&
      FX=real,PHIX=real,FY=real,PHIY=real,FT=real,PHIT=real
```

The START statement must be entered after the TRACK, but before the RUN command.

For each canonical pair two specifications are possible. First, they may be specified as displacements with respect to the closed orbit: X, PX, Y, PY, T, DELTAP, as defined in Section 1.5.1. In this case the initial conditions are:

$$\begin{aligned} x &= X, & p_x &= PX, \\ y &= Y, & p_y &= PY, \\ \Delta t &= T, & \Delta E/cp_0 &= DELTAP. \end{aligned}$$

It is also possible to enter rotations in normalised phase space. Initial conditions Z in normalised phase space are related to the closed orbit and normalised via the eigenvectors:

$$\begin{aligned} Z = Z_{co} &+ \sqrt{E_x}FX(\Re V_k \cos PHIX + \Im V_k \sin PHIX) \\ &+ \sqrt{E_y}FY(\Re V_k \cos PHIY + \Im V_k \sin PHIY) \\ &+ \sqrt{E_t}FT(\Re V_k \cos PHIT + \Im V_k \sin PHIT) \end{aligned}$$

where Z_{co} is the closed orbit vector, $\Re V_k$ and $\Im V_l$ are the real and imaginary parts of the k^{th} eigenvectors. The other variables have the meaning:

FX The normalised (betatron) amplitude for mode 1,

PHIX The (betatron) phase $PHIX = \phi_x/2\pi$ for mode 1,

FY The normalised (betatron) amplitude for mode 2,

PHIY The (betatron) phase $PHIY = \phi_y/2\pi$ for mode 2,

FT The normalised (synchrotron) amplitude for mode 3,

PHIT The (synchrotron) phase $PHIT = \phi_t/2\pi$ for mode 3.

The eigenvectors are computed in the TRACK command. The emittances are calculated in the latest EMIT command preceding it, or taken from the latest BEAM command, whichever comes last.

13.5 TSAVE Statement

The TSAVE command

```
TSAVE,FILENAME=string
```

saves the latest particle positions on the file named by string. These will normally be the positions of surviving particles after the last RUN command. If no RUN command has been seen yet, the positions are taken from any START commands seen. The positions are written as START commands, which may be read by a subsequent tracking run.

13.6 RUN Statement

This command starts or continues the actual tracking:

```
RUN,METHOD=TRANSPORT|LIE3|LIE4 [,TABLE=name]&
    TURNS=integer,FPRINT=integer,FFILE=integer
```

The RUN command initialises tracking and uses the values in the last particle bank for initial conditions. It optionally builds a table of particle positions, suited for plotting or further analysis; it may also write a portable binary file. Note that the CONTINUE command has been deleted from Version 8.2. RUN has the attributes:

METHOD The name of the tracking method to be used. It may be one of:

TRANSPORT The TRANSPORT method [9], using transfer matrices of order two (default),
 LIE3 The Lie-algebra method [16] to order 2 in the canonical variables,
 LIE4 The Lie-algebra method [16] to order 3 in the canonical variables.

TABLE The name to be given to the tracking table. If a table with the same name already exists, it is discarded. If TABLE is omitted, no table is built. The table is not saved automatically; it can be saved on a file by the ARCHIVE command.

TURNS The number of turns (integer) to be tracked (default: 1).

FPRINT The print frequency, an integer (default: no printing). Printing always occurs before the first and after the last turn. If the value of FPRINT is greater than zero, printing also occurs after every FPRINT turns. This print-out includes normalised values, invariants, and phases (for their definitions see 13.4). Printing in other positions may be requested by a SELECT, TRACK command.

FFILE The output frequency for the binary file, an integer (default: no printing). If the value of FFILE is greater than zero, output occurs after every FFILE turns. The binary file is written in EPIO format, and has not been frozen yet. The output will be used by an off-line analysis program.

Example:

```
RUN,TURNS=1000,FPRINT=10
```

This RUN command tracks 1000 turns and prints every 10th turn.

13.7 DYNAPSTART and DYNAP: Dynamic Aperture search

The new command DYNAPSTART has attributes common to the MAD commands START and RUN. Its purpose is to define initial conditions XSTART, PXSTART, YSTART, PYSTART, TSTART, PTSTART and tracking options for *a single particle*. Tracking is then invoked by a subsequent call to DYNAP, which performs an automatic search of dynamic aperture and may compute several auxiliary quantities according to the additional attributes of DYNAPSTART. The dynamic aperture is expressed as a fraction DYNAPFRAC of the initial betatron amplitudes. The syntax of this new command is

```
DYNAPSTART, X=real, PX=real, Y=real, PY=real, T=real, DELTAP=real, &
    FX=real, PHIX=real, FY=real, PHIY=real, FT=real, PHIT=real, &
    METHOD=TRANSPORT|LIE3|LIE4, TABLE=name, DAMP, &
    TURNS=integer, STEP=real, CHKBELow=real, FRACMIN=real, &
    FASTUNE, LYAPUNOV, CONTINUE
```

All the attributes are optional and have the following meaning:

X, PX, Y, PY, T, DELTAP initial particle displacements w.r.t. the closed orbit,
 FX, PHIX, FY, PHIY, FT, PHIT normalised amplitudes and phases (as in START),
 METHOD name of the tracking method to be used (default: TRANSPORT),
 TABLE name to be given to the tracking table (if LYAPUNOV is also selected, this table contains the difference between the phase space coordinates of two companion particles),
 DAMP flag to perform tracking with radiation damping,
 TURNS number of turns to be tracked (default: 64),
 step fractional step for downward search of dynamic aperture (default: 0.05),
 CHKBELow particle stability is checked below DYNAPFRAC down by a fraction CHKBELow (default: 0),
 FRACMIN the downward search of dynamic aperture is stopped if DYNAPFRAC becomes smaller than FRACMIN (default: 0.1); in this case the variable TURNS is set to the number of turns the particle survived,
 FASTUNE flag to compute the tunes TUNX and TUNY, together with the quadratic sum DTUNE of their variations over half the number of TURNS, by interpolated fast Fourier transform and Hanning filter (if TURNS \geq 64), using the method discussed in Ref. [4]: this option works more efficiently when the number of TURNS is a power of two and requires a non-vanishing oscillation amplitude in both betatron planes,
 LYAPUNOV flag to perform tracking of two companion particles with horizontal separation DO (the default separation $DO = 10^{-7}$ m can be changed by specifying LYAPUNOV=real): the Lyapunov exponent is then estimated by a fit of the phase space distance between the two companion particles versus the number of turns,
 CONTINUE flag to ADD the new initial conditions to the final conditions XEND, PXEND, YEND, PYEND, TEND, PTEND at the end of the last tracking invoked by DYNAP (this option can be useful to simulate acceleration).

The closed orbit and normal mode eigenvectors (used to define initial conditions through normalised amplitudes and phases) are computed only when required: this improves the speed of dynamic aperture search for variable parameters. During matching, the orbit should be recomputed using the option CELL, ORBIT (see Section 12.1. After a call to DYNAPSTART, the following variables are defined:

XSTART, PXSTART, YSTART, PYSTART, TSTART, PTSTART,
 TURNS, STEP, CHKBELow, FRACMIN, DO,
 WXSTART, WYSTART, WXYSTART,

where WXSTART and WYSTART are the two betatron invariants in metres (i.e., not normalised to the emittances) and WXYSTART is their sum. The initial conditions XSTART, PXSTART, ..., can be set directly without using DYNAPSTART: they can also be varied during matching for fixed point search through GLOBAL constraints (see Section 12.4). Note that the variable containing the initial momentum deviation is PTSTART, although the corresponding attribute in DYNAPSTART is called DELTAP.

After a call to DYNAP (which has no attributes), the following variables are computed:

DYNAPFRAC, TURNS,
 XEND, PXEND, YEND, PYEND, TEND, PTEND,
 WXMIN, WXMAX, WYMIN, WYMAX, WXYMIN, WXYMAX, SMEAR,
 TUNX, TUNY, DTUNE, LYAPUNOV.

The fractional dynamic aperture DYNAPFRAC (when smaller than unity) expresses the maximum stable betatron amplitude, with prescribed synchrotron initial conditions and with a relative error equal to STEP, along the *direction* corresponding to the initial betatron conditions assigned by DYNAPSTART. The minimum and maximum betatron invariants in each plane Wxmin, Wxmax, Wymmin, Wymax and the extrema of their sums Wxymin, WxyMAX are automatically determined during tracking (they are expressed again in metres) together with the “smear”, defined by $SMEAR = 2(WXYMAX - WXYMIN)/(WXYMAX + WXYMIN)$. In the absence of radiation damping, these quantities carry additional information (usually not exploited in tracking studies) about the dependence of the dynamic aperture on the initial betatron phases; their calculation requires a negligible fraction of the tracking CPU time.

Particle stability over the number of TURNS defined by DYNAPSTART is first tested by tracking with the specified initial conditions. If the particle is not lost (and CHKBELow is zero), the value of DYNAPFRAC is set to unity. If the particle is lost, the initial *betatron* displacements XSTART, PXSTART, YSTART and PYSTART (but not the synchrotron ones) are progressively reduced by a factor $(1 - STEP)$ until the particle survives: at each iteration, the value of DYNAPFRAC is also multiplied by the same reduction factor. If DYNAPFRAC becomes smaller than FRACMIN, this downward search is stopped and the variable TURNS contains the particle survival time: therefore setting $FRACMIN = 1$ is useful to produce survival plots.

If CHKBELow is different from zero, particle stability over the specified number of TURNS is also checked below DYNAPFRAC, in order to detect possible “stability holes”. Using the same fractional STEP, tracking is repeated for a number n of iterations such that $(1 - STEP)^n \leq 1 - CHKBELow$: if the particle is lost at any intermediate step, the whole process is restarted. For a coarse-grained, fast dynamic aperture search one can use a STEP of 5% and a value of CHKBELow of 10%, corresponding to two steps below DYNAPFRAC. For a fixed value of CHKBELow, smaller values of STEP (say 1%) give rise to a finer scan in phase space and thus to a better relative precision on the final dynamic aperture, with the same safety margin for no “stability holes” below DYNAPFRAC. It should be clearly understood, however, that *no finite algorithm* can guarantee particle stability below *any* initial oscillation amplitude for a given number of turns. The only case when this would be mathematically possible (autonomous systems with only one degree of freedom, such as a pendulum) is of no practical interest in accelerator physics. Our choice of dynamic aperture search strategy is a compromise between speed requirements (tracking “from above” is faster, since unstable particles are quickly lost) and uniform relative precision on the maximum stable amplitude: more efficient strategies, based on the search of hyperbolic fixed points, are conceivable for some limited classes of dynamical systems [18, 6], but they are not yet available for general use in accelerators.

The Lyapunov exponent is estimated by a fit of the normalised phase space distance D/D_0 between two companion particles versus the turn number N : if the option TABLE is also selected in DYNAPSTART, a tracking table containing the difference between the phase space coordinates of the two particles is generated and their normalised distance can be plotted using predefined STRING variables (see next section). The tracking interval is subdivided into four subsequent periods of length TURNS/4 and the first sub-period, corresponding to fast transients, is discarded: if the slope of $\ln(D/D_0)$ versus $\ln(N)$ in each of the three subsequent periods is equal to unity to within 10%, the variable lyapunov is set to zero (i.e., the particle should be considered as stable). Otherwise the value of LYAPUNOV is the maximum of the slopes (over each of the three subsequent intervals) of $\ln(D/D_0)$ versus N : in this case, the trajectories of the two companion particles in at least one sub-period are exponentially divergent, corresponding to chaotic behaviour. The exponential separation is accompanied by larger and larger differences in the betatron phases of the two companion particles, which periodically come back very close to each other when their relative phases become multiples of 2π . The choice of three subsequent intervals is adopted to reduce the probability that the distance between the two companion particles be decreasing, rather than increasing, over all the sub-periods. This choice, corresponding to an optimum number of TURNS around 1000, can in principle be modified by changing the Fortran subroutine FITLYAP. The Lyapunov exponent LYAPUNOV and the detuning à la Laskar DTUNE [4, 26]

are early indicators of particle chaoticity and can be very useful to speed up the investigations of long-term stability.

13.8 A Tracking Example

```
L: LINE=(OCT,-OCT)
USE,L
! Misalignments and closed orbit correction may be done here.
BEAM,EX=0.1E-6,EY=0.05E-6,ET=1.0E-3,ENERGY=60.0
TRACK
  START,X=0.001,Y=0.001,DELTAP=0.001 ! absolute X and Y
  START,FX=10.0,FY=10.0,DELTAP=-0.001 ! particle at 10 sigma
  RUN,TURNS=1024,FPRINT=1
  PLOT,HAXIS=X,VAXIS=PX,MULTIPLE
ENDTRACK
```

Plotting can be requested as described in section 1.5.6. The position of the PLOT command is irrelevant, as long as it occurs after the RUN command, and before any command which overwrites the track table generated.

Chapter 14

Subroutines and Procedures

Table 14.1: Subroutine Commands

Name	Meaning	Section
SUBROUTINE	Begin subroutine definition	14.1
ENDSUBROUTINE	End subroutine definition	14.1
CALLSUBROUTINE	Call predefined subroutine	14.2
DO	Repeated execution of commands	14.3
ENDDO	End of DO group	14.3

The input language provides features for gathering strings of statements, which may be treated as a simple command called a *subroutine*. There are commands to define a subroutine in terms of other commands, and to execute subroutines.

These language features allow the user to assemble a personal library of commonly used commands. Individual commands and subroutines may be called from this library whenever they are wanted, and simple labels may be substituted for relatively complicated subroutines or commands. The personal statement library can be appended to the MAD dictionary, which makes the statements available automatically when the program is started. Alternatively it can be placed in a user file and accessed on demand via the CALL and RETURN commands. Good candidates for a subroutine library are command sequences for the HARMON, Matching, or Tracking Module.

14.1 SUBROUTINE and ENDSUBROUTINE

The SUBROUTINE and ENDSUBROUTINE statements delimit a group of related statements which together form the body of the subroutine. A subroutine definition, followed by a call, has the format

```
label:  SUBROUTINE      ! begin of subroutine "label"
        ...             ! included statements
        ENDSUBROUTINE  ! end of subroutine "label"
        ...             ! other commands
        label          ! call to subroutine "label"
```

The subroutine is given the name label. The statements included are checked and stored in memory, but not executed until the subroutine is called. Example:

```
S: SUBROUTINE ! begin of definition for S
    USE,SSC
    SURVEY
    TWISS
```

```
ENDSUBROUTINE ! end of definition for S
...
S             ! here S is executed the first time
```

The size of a subroutine is limited only by the available memory space. A labelled command within a subroutine may be called by its label from anywhere in the program. A subroutine cannot contain another SUBROUTINE, DO or ENDDO statement. However, within a subroutine, nested calls to other subroutines are permitted. Also, references to DO statements which have already been accepted by the program are permitted.

14.2 CALLSUBROUTINE

The CALLSUBROUTINE command causes execution of a group of commands previously assembled as a subroutine or procedure. It is included in the language for formal purposes only, to make the call explicit.

```
CALLSUBROUTINE,label
```

A statement consisting of label only is equivalent to CALLSUBROUTINE,label. The following are two equivalent calls to a subroutine S:

```
S
CALLSUBROUTINE,S
```

14.3 DO and ENDDO

The DO statement allows grouping of one or more commands and offers an option to repeat their execution. Both subroutines and commands may be included. The list of commands follows the DO statement, and ends with an ENDDO:

```
DO,TIMES = integer]
    {command}
ENDDO
```

The commands SUBROUTINE, ENDSUBROUTINE, or DO must not be placed between DO and ENDDO. Example:

```
QUAD3:QUAD,L=1.5,K1=0.01
DO,TIMES=2
    TWISS,DELTAP=-0.01:+0.01:0.002
    SET, QUAD3[K1], QUAD3[K1]+0.001
ENDDO
```

This example shows how a SET statement may change parameters of the computations after each iteration. The length of a DO list is not limited.

Chapter 15

Known Defects of Version 8

15.1 Definitions

MAD uses full 6×6 matrices to allow coupling effects to be treated, and the canonical variable set $x, p_x/p_0, y, p_y/p_0, -c\Delta t, \Delta E/p_0c$, as opposed to other programs most of which use the set $x, x', y, y', -\Delta s, \delta$. Like the program MARYLIE [16], MAD uses the relative energy error $\Delta E/p_0c$ which is related to the relative momentum error δ by $\Delta E/p_0c = \beta\Delta p/p_0 = \beta\delta$.

As from Version 8.13, MAD uses an additional *constant* momentum error $\delta_s = \Delta p/p_0$ in all optical calculations. The transfer maps contain the *exact* dependence upon this value; therefore the tunes for large deviations can be computed with high accuracy as opposed to previous versions.

The choice of canonical variables in MAD still leads to slightly different definitions of the lattice functions. In MAD the Courant-Snyder invariants [13] take the form $W_x = \gamma_x x^2 - 2\alpha_x x p_x + \beta_x p_x^2$. Comparison to the original form $W_x = \gamma_x x^2 - 2\alpha_x x x' + \beta_x x'^2$ shows that the orbit functions cannot be the same. A more detailed analysis, using $x' = p_x/(1 + \delta)$, shows that all formulas can be made consistent by defining the MAD orbit functions as

$$\beta_{xM} = \beta_x(1 + \delta), \quad \alpha_{xM} = \alpha_x, \quad \gamma_{xM} = \gamma_x/(1 + \delta).$$

For constant δ_s along the beam line and $\delta = 0$, the lattice functions are the same. In a machine where δ varies along the circumference, e.g. in a linear accelerator or in an e^+e^- storage ring, the definition of the Courant-Snyder invariants must be generalised. The MAD invariants have the advantage that they remain invariants along the beam line even for variable δ .

With the new method this problem occurs in TWISS only for non-constant δ .

15.1.1 Treatment of $\partial^2 Q / \partial \delta^2$ in TWISS

It has been noted in [29] that MAD returned tunes which are too low for non-zero Δp . The difference was found to be quadratic in Δp with a negative coefficient. This problem has been eliminated thanks to the new treatment of momentum errors from Version 8.13 onwards.

Appendix A

Format of the SURVEY, STRUCTURE, and TWISS Files

All output generated by the TAPE options of the commands SURVEY or TWISS has the same general layout. The output is a coded file with default name SURVEY or TWISS. It consists of a header record, a set of position records, and a trailer record.

A.1 Common Output

In all cases the header record has two lines. It has the format

(5A8,I8,L8,I8/A80)

and contains:

PROGVRSN The current version number of MAD,

DATAVRSN This field may contain the following strings:

SURVEY Output of the SURVEY command follows,

STRUCTURE Output of the STRUCTURE command follows,

TWISS Output of the TWISS command (linear lattice functions), follows

CHROM Output for the CHROM option of the TWISS command follows,

DATE The date of the MAD run. This is the same date as on the printed output.

TIME The time of the MAD run. This is the same time as on the printed output.

JOBNAME The job name for the MAD run, if available from the operating system.

SUPER The number of superperiods.

SYMM The symmetry flag. This is true, if only one half of a symmetric super-period was computed.

NPOS The number of position records that follow. This is larger by one than the number of elements.

TITLE The page header as defined by the latest TITLE command.

There are NPOS position records. Each of them contains four or five lines. The first two lines have the format

(A4,A16,F12.6,3E16.9/5E16.9)

and contain the following information about the preceding element:

KEYWORD The element keyword,

NAME The element name,

Note that the TYPE parameter no longer occurs in this file. The remaining values are written according to Table A.1. The first position record refers to the position preceding the first element. Its content is

KEYWORD=blank, NAME='INITIAL'

All other values are zero. The remaining lines of the position records differ for the three cases. The

Table A.1: Element Data in Position Records

Keyword									
DRIFT	L	0	0	0	0	0	0	0	0
RBEND	L	ANGLE	K1	K2	TILT	E1	E2	H1	H2
SBEND	L	ANGLE	K1	K2	TILT	E1	E2	H1	H2
QUADRUPOLE	L	0	K1	0	TILT	0	0	0	0
SEXTUPOLE	L	0	0	K2	TILT	0	0	0	0
OCTUPOLE	L	0	0	0	TILT	K3	0	0	0
MULTIPOLE	0	KOL	K1L	K2L	T0	K3L	T1	T2	T3
SOLENOID	L	0	0	0	0	KS	0	0	0
RFCAVITY	L	0	0	0	0	FREQ	VOLT	LAG	0
ELSEPARATOR	L	0	0	0	TILT	EFIELD	0	0	0
KICKER	L	0	0	0	HKICK	VKICK	0	0	0
HKICKER	L	0	0	0	HKICK	0	0	0	0
VKICKER	L	0	0	0	0	VKICK	0	0	0
SROT	0	0	0	0	0	ANGLE	0	0	0
YROT	0	0	0	0	0	ANGLE	0	0	0
MONITOR	L	0	0	0	0	0	0	0	0
HMONITOR	L	0	0	0	0	0	0	0	0
VMONITOR	L	0	0	0	0	0	0	0	0
MARKER	0	0	0	0	0	0	0	0	0

magnetic fields do not contain any field errors.

For the STRUCTURE command the output contains the field errors, and the format for thin multipoles has been changed. The first line (format (A4,A16,F12.6,1P,2E16.9,I16)) contains the element type and name, the length (zero), the horizontal and vertical deflections, and highest order written. The following line(s) contain the multipole components (normal and skewed) for first to maximum order. The STRUCTURE command writes no further data beyond the element table.

A.2 SURVEY Output

When DATAVRSN=SURVEY, the third and fourth lines of the position records have the format

(4E16.9/3E16.9)

and contain the global coordinates and the cumulative length in the order

X	Y	Z	SUML
THETA	PHI	PSI	

The trailer record has two lines in the format

(3E16.9/3E16.9)

and contains the coordinates of the machine centre, the minimum and maximum radius and the machine circumference in the order

X	Y	Z
RMIN	RMAX	C

A.3 TWISS Output

When DATAVRSN=TWISS, the third to fifth lines of the position records have the format

(5E16.9/5E16.9/5E16.9)

and contain the quantities

ALFX	BETX	MUX	DX	DPX
ALFY	BETY	MUY	DY	DPY
X	PX	Y	PY	SUML

in this order. The trailer record has three lines in the format

(3E16.9/5E16.9/5E16.9)

and contains

DELTAP	GAMTR	C		
COSMUX	QX	QX'	BXMAX	DXMAX
COSMUY	QY	QY'	BYMAX	DYMAX

in this order. If the COUPLE option was given, the output has the same format, but the quantities given refer to the two possible modes (1,2) instead of referring to the two planes (x, y) .

A.4 CHROM Output

When DATAVRSN=CHROM, the third to fifth lines of the position records have the format

(5E16.9/5E16.9/5E16.9)

and contain

WX	PHIX	DMUX	DDX	DDPX
WY	PHIY	DMUY	DDY	DDPY
X	PX	Y	PY	SUML

in this order. There is no trailer record in this case.

Appendix B

Format for File Names

The MAD program normally uses alphanumeric names for files. This Appendix describes the valid formats for file names, as used in different computer operating systems.

B.1 IBM VM/CMS System

B.1.1 Standard Streams

The standard streams are defined within MAD with the names listed in Table B.1. At CERN, the

Table B.1: Standard Files Used by MAD, IBM-VM/CMS Version

Purpose	unit	file name
Command dictionary input	4	DICT
Normal input	5	
Input lines and error messages	6	
Plot output (GKS metafile)	8	MAD METAFILE
Plot output (HIGZ metafile)	8	MAD PS
Normal output	14	PRINT
Dynamic tables	15	TABLE

EXEC file MAD8 EXEC stored on the MAD disk performs the following default initialisations:

```
FILEDEF DICT      DISK MAD8 DICT *
FILEDEF 5         DISK TERMINAL
FILEDEF 6         DISK TERMINAL
FILEDEF PRINT     DISK problem LISTING scratch
FILEDEF DUMP      DISK problem DUMP scratch
FILEDEF METAFILE  DISK MADPLOT METAFILE scratch
```

Problem is the name of the input file (or PROBLEM in interactive mode). Scratch is the mode of a scratch disk created by the EXEC file. The behaviour of MAD can be adapted to the user's needs by changing the EXEC file.

B.1.2 User-Defined Files

All other file names must be chosen according to the VM/CMS conventions. Within MAD a file name has one to three parts separated by dots

filename.filetype.filemode

```
filename.filetype
      filetype
```

All letters are converted to uppercase. If all three parts are present, the file name is used as entered. Example:

```
CALL,FILENAME='LEPTWO.DATA.D'
```

If two parts are present, they are taken as filename and filetype. The filemode defaults to * for input files, or to the contents of the REXX variable SCRATCH for output files. SCRATCH The CERN EXEC MAD8 EXEC sets this variable to the mode of a scratch disk. Examples:

```
CALL,  FILENAME='TEST.MAD'
TWISS,  TAPE='LEP.TWISS'
```

Assume that the mode of the scratch disk is F. CALL requires an input file and TWISS an output file, so these commands have the same effect as:

```
CALL,  FILENAME='TEST MAD *'
TWISS,  TAPE='LEP TWISS F'
```

If only one part appears, it is used as filetype. The filemode has the same default as above, and filename defaults to the contents of the REXX variable PROBLEM. The CERN EXEC MAD8 EXEC sets this variable to the filename of the DATA file, or if MAD runs interactively, to the value PROBLEM. Example:

```
CALL,  FILENAME='DATA'
CALL                                ! FILENAME='SAVE' is default
TWISS,  TAPE                                ! FILENAME='TWISS' is default
TWISS,  TAPE='LATTICE'
```

Assume that the filename of the DATA file is LEP and the mode of the scratch disk is F. The above examples have then the same effect as:

```
CALL,  FILENAME='LEP DATA *'
CALL,  FILENAME='LEP SAVE *'
TWISS,  TAPE='LEP TWISS F'
TWISS,  TAPE='LEP LATTICE F'
```

B.2 UNIX and UNICOS Systems

B.2.1 Standard Stream Names

The standard streams are defined within MAD with the names listed in Table B.2. Note that on the Cray XMP the table file is not used; dynamic tables reside in SSD. File names are converted to upper case, unless they are enclosed in quotes. Without special action they refer to files in the user's working directory. They can be assigned other names by the ln command entered at the keyboard or issued in a shell script. Such a link is desired in particular for the command dictionary:

```
ln /usr/madman/mad.dict dict
```

This link makes the command dictionary available under the assumption that it resides in the file /usr/madman/mad.dict. It is recommended to create a subdirectory for each problem, and to store the data file in this subdirectory. MAD should then be run while this directory is current, so as to create all output files in the same directory. This helps keeping related files together.

Table B.2: Standard Files Used by MAD, UNIX Version

Purpose	unit	file name
Command dictionary input	4	dict
Normal input	5	stdin
Input lines and error messages	6	stdout
Plot output (GKS metafile)	8	mad.metafile
Plot output (HIGZ metafile)	8	mad.ps
Normal output	14	print
Dynamic tables	15	table

B.2.2 User-Defined Files

Other files can be assigned any name accepted by the UNIX system, with the restriction that MAD accepts at most 40 characters, and that it converts file names to upper case unless enclosed in quotes. Examples:

```
CALL,    FILENAME='ssc.data'
TWISS,   TAPE='ssc/twiss.data'
```

The first file is read in the user's working directory, and the TWISS output is written on the file `twiss.data` in the subdirectory `ssc`.

B.3 VAX VMS System

B.3.1 Standard Stream Names

The standard streams are defined within MAD with the names listed in Table B.3. Without special

Table B.3: Standard Files Used by MAD, VAX-VMS Version

Purpose	unit	file name
Command dictionary input	4	DICT
Normal input	5	
Input lines and error messages	6	
Plot output (GKS metafile)	8	METAFILE
Plot output (HIGZ metafile)	8	MAD.PS
Normal output	14	PRINT
Dynamic tables	15	TABLE

action, these file names refer to files in the user's working file directory. They can be assigned other names by an ASSIGN command entered at the keyboard or issued in a DCL file. This is desirable in particular for the command dictionary:

```
ASSIGN DISK\%SI:[PUBSI]MAD8.DICT DICT
```

This command makes the dictionary available on the CERN central VAX system (VXCERN). The other standard streams should be assigned by a sequence like

```
ASSIGN DISK\%gg:[user]mad.print PRINT
ASSIGN DISK\%gg:[user]mad.metafile METAFILE
```

B.3.2 User-Defined Files

Other files can be assigned any name accepted by the VMS system, with the restriction that MAD accepts at most 40 characters. All letters are converted to upper case. Examples:

```
CALL,    FILENAME='SSC.DATA'  
TWISS,   TAPE='DISK\$_IZ[MADMAN]SSC.TWISS'  
SURVEY,  TAPE='DISK\$_IZ[MADMAN.SSC]SURVEY.DATA'
```

The first file is read in the user's working directory; the TWISS output is written on a file SSC.TWISS in the current directory, and the SURVEY output is written on a file SURVEY.DATA in the subdirectory SSC.

Appendix C

Format of TFS Files

TFS files (Table File System) are used in the LEP control system. Their use is documented in Reference [14]. The MAD program knows only coded TFS files. This Appendix describes the special formats used.

C.1 Descriptor Lines

MAD writes the following descriptors on all tables:

COMMENT The current title string from the most recent TITLE command.

ORIGIN The version of MAD used.

DATE The date of the MAD run.

TIME The wall clock time of the MAD run.

TYPE The type of the table:

 TWISS Lattice function table.

 OPTICS Output of the OPTICS command.

 TRACK Track table.

 TUNES Table of tunes and chromaticities versus $\Delta E/p_0c$.

The track tables contain the following additional descriptors:

BETX Horizontal β_x ,

ALFX Horizontal α_x ,

BETY Vertical β_y ,

ALFY Vertical α_y ,

X Horizontal orbit position,

PX Horizontal orbit slope,

Y Vertical orbit position,

PY Vertical orbit slope,

DX Horizontal dispersion,

DPX Horizontal dispersion slope,

DY Vertical dispersion,

DPY Vertical dispersion slope,

The format of descriptor lines is

('@ ',A16,' ',A4,...)

Where the A16 format is used for the descriptor name, the A4 format for its format, and the remaining information uses the format listed below for columns.

C.2 Column Formats

The column formats used are listed in Table C.1. Control lines begin with the TFS control character,

Table C.1: Column Formats used in TFS Tables

C format	Meaning	FORTRAN format
%hd	Short integer	(I8)
%lf	Long float	(G20.12)
%f	Short float	(G14.6)
%ks	String of length k	(''',A,'''')
	Undefined value	('~')

followed by a blank. Data lines begin with two blanks. Columns are also separated by one blank character. The column width is chosen such as to accommodate the large of the column name and the data values of the column. Thus an integer column uses at least 8 characters to accommodate the (I8) format, but if its name has more than 8 characters, it becomes wider.

Bibliography

- [1] *The Graphical Kernel System (GKS)*. ISO, Geneva, July 1985. International Standard ISO 7942.
- [2] B. Autin and Y. Marti. *Closed Orbit Correction of Alternating Gradient Machines using a small Number of Magnets*. CERN/ISR-MA/73-17, CERN, 1973.
- [3] D.P. Barber, K. Heinemann, H. Mais and G. Ripken, *A Fokker–Planck Treatment of Stochastic Particle Motion within the Framework of a Fully Coupled 6-dimensional Formalism for Electron-Positron Storage Rings including Classical Spin Motion in Linear Approximation*, DESY report 91-146, 1991.
- [4] R. Bartolini, A. Bazzani, M. Giovannozzi, W. Scandale and E. Todesco, *Tune evaluation in simulations and experiments*, CERN SL/95-84 (AP) (1995).
- [5] J. D. Bjorken and S. K. Mtingwa. *Particle Accelerators* **13**, pg. 115.
- [6] E.M. Boltt and J.D. Meiss, *Targeting chaotic orbits to the Moon through recurrence*, Phys. Lett. **A204**, 373 (1995).
- [7] P. Bramham and H. Henke. private communication and LEP Note LEP-70/107, CERN.
- [8] Karl L. Brown. *A First-and Second-Order Matrix Theory for the Design of Beam Transport Systems and Charged Particle Spectrometers*. SLAC 75, Revision 3, SLAC, 1972.
- [9] Karl L. Brown, D. C. Carey, Ch. Iselin, and F. Rothacker. *TRANSPORT — A Computer Program for Designing Charged Particle Beam Transport Systems*. CERN 73-16, revised as CERN 80-4, CERN, 1980.
- [10] A. Chao. *Evaluation of beam distribution parameters in an electron storage ring*. Journal of Applied Physics, 50:595–598, 1979.
- [11] A. W. Chao and M. J. Lee. *SPEAR II Touschek lifetime*. SPEAR-181, SLAC, October 1974.
- [12] M. Conte and M. Martini. *Particle Accelerators* **17**, 1 (1985).
- [13] E. D. Courant and H. S. Snyder. *Theory of the alternating gradient synchrotron*. Annals of Physics, 3:1–48, 1958.
- [14] Ph. Defert, Ph. Hofmann, and R. Keyser. *The Table File System, the C Interfaces*. LAW Note 9, CERN, 1989.
- [15] M. Donald and D. Schofield. *A User’s Guide to the HARMON Program*. LEP Note 420, CERN, 1982.
- [16] A. Dragt. *Lectures on Nonlinear Orbit Dynamics, 1981 Summer School on High Energy Particle Accelerators, Fermi National Accelerator Laboratory, July 1981*. American Institute of Physics, 1982.

- [17] D. A. Edwards and L. C. Teng. *Parametrisation of linear coupled motion in periodic systems*. IEEE Trans. on Nucl. Sc., 20:885, 1973.
- [18] M. Giovannozzi, *Analysis of the stability domain of planar symplectic maps using invariant manifolds*, CERN/PS 96-05 (PA) (1996).
- [19] H. Grote. *GXPLOT User's Guide and Reference Manual*. LEP TH Note 57, CERN, 1988.
- [20] LEP Design Group. *Design Study of a 22 to 130 GeV e^+e^- Colliding Beam Machine (LEP)*. CERN/ISR-LEP/79-33, CERN, 1979.
- [21] M. Hanney, J. M. Jowett, and E. Keil. *BEAMPARAM — A program for computing beam dynamics and performance of e^+e^- storage rings*. CERN/LEP-TH/88-2, CERN, 1988.
- [22] R. H. Helm, M. J. Lee, P. L. Morton, and M. Sands. *Evaluation of synchrotron radiation integrals*. IEEE Trans. Nucl. Sc., NS-20, 1973.
- [23] F. James. *MINUIT, A package of programs to minimise a function of n variables, compute the covariance matrix, and find the true errors*. program library code D507, CERN, 1978.
- [24] E. Keil. *Synchrotron radiation from a large electron-positron storage ring*. CERN/ISR-LTD/76-23, CERN, 1976.
- [25] D. E. Knuth. *The Art of Computer Programming*. Volume 2, Addison-Wesley, second edition, 1981. Semi-numerical Algorithms.
- [26] J. Laskar, C. Froeschlé and A. Celletti, *The measure of chaos by the numerical analysis of the fundamental frequencies. Application to the standard mapping*, Physica D**56**, 253 (1992).
- [27] H. Mais and G. Ripken, *Theory of Coupled Synchro-Betatron Oscillations*. DESY internal Report, DESY M-82-05, 1982.
- [28] M. Meddahi, *Chromaticity correction for the $108^\circ/60^\circ$ lattice*, CERN SL/Note 96-19 (AP) (1996).
- [29] J. Milutinovic and S. Ruggiero. *Comparison of Accelerator Codes for a RHIC Lattice*. AD/AP/TN-9, BNL, 1988.
- [30] B. W. Montague. *Linear Optics for Improved Chromaticity Correction*. LEP Note 165, CERN, 1979.
- [31] Gerhard Ripken, *Untersuchungen zur Strahlführung und Stabilität der Teilchenbewegung in Beschleunigern und Storage-Ringen unter strenger Berücksichtigung einer Kopplung der Betatronschwingungen*. DESY internal Report R1-70/4, 1970.
- [32] F. Ruggiero, *Dynamic Aperture for LEP 2 with various optics and tunes*, Proc. Sixth Workshop on LEP Performance, Chamonix, 1996, ed. J. Poole (CERN SL/96-05 (DI), 1996), pp. 132–136.
- [33] L. C. Teng. *Concerning n-Dimensional Coupled Motion*. FN 229, FNAL, 1971.
- [34] U. Völkel. *Particle loss by Touschek effect in a storage ring*. DESY 67-5, DESY, 1967.
- [35] R. P. Walker. *Calculation of the Touschek lifetime in electron storage rings*. 1987. Also SERC Daresbury Laboratory preprint, DL/SCI/P542A.
- [36] P. B. Wilson. *Proc. 8th Int. Conf. on High-Energy Accelerators*. Stanford, 1974.
- [37] A. Wrulich and H. Meyer. *Life time due to the beam-beam bremsstrahlung effect*. PET-75-2, DESY, 1975.

Index

- !, 12
- *, 9, 15
- +, 15
- , 15
- ., 9
- /, 15
- ;, 12
- [, 9
- [a-ex-z...], 9
- [xyz], 9
- #E, 49
- #F, 49
- #L, 49
- #S, 49
- &, 12
- |, 12
- ^ , 15

- A, 63
- abbreviations, 14
- ABS(X), 15
- action commands, 51
- ADD, 84, 87
- ALFA, 8
- ALFX, 7, 58, 59, 64, 108, 132
- ALFY, 7, 58, 59, 64, 108, 132
- alignment errors, 80
- ALL, 88–90
- AMPLITUDE(i), 116
- ANGLE, 27, 34
- angle, 10
 - rotation, 2
- ANTI-PROTON, 53
- aperture, 33
- arc length, 6
- ARCHIVE, 75, 78
- arguments, 41
- ASCALE, 92
- ASIN(X), 15
- ASSIGN, 75, 76
- assignment
 - parameter, 20
- AT, 44

- attribute, 12, 13, 17
 - deferred, 17
 - element, 26
 - integer, 15
 - logical, 15
 - name, 13, 14, 18
 - real, 15
 - string, 14
 - value, 13
- attribute, 26
- axes
 - global, 3
 - local, 3
- azimuth, 2

- bar, 12
- BARS, 92
- BCURRENT, 54
- BEAM, 51, 53, 115–117
- beam
 - current, 54, 71
 - defaults, 54
- beam interaction, 35
- beam line, 39
 - arguments, 41
 - reference, 47
 - reflection, 40
 - repetition, 40
 - sequence, 42
 - simple, 39
 - used, 51
- beam monitor, 33
- beam monitor errors, 80
- beam parameters, 69–74
- beam size, 73
- BEAMBEAM, 25, 35
- BEAMPARAM, 69
- bend
 - angle, 4
 - radius, 4
- bending magnet, 4, 26
- bending magnet errors, 83
- BETA0, 59, 60, 64, 107

- BETATRON, 63, 64
- BETRF, 32, 73
- BETX, 7, 58, 59, 64, 92, 94, 105, 108, 132
- BETX, BETY, 95
- BETY, 7, 58, 59, 64, 92, 94, 105, 108, 132
- blanks, 12
- BMPM, 69
- BMPM, 69–74
- braces, 12
- BROT, 83
- BUCKET, 70, 73
- bunch, 54
 - current, 54
 - length, 54
 - number, 54
- BUNCHED, 54
- BX'I, 101
- BX'S, 101
- BXMAX, 8
- BY, 44, 45
- BY'I, 101
- BY'S, 101
- BYMAX, 8

- C1LIST, 90
- C2LIST, 89, 90
- CALL, 75, 76, 109
- CALLS, 100, 111
- CALLSUBROUTINE, 122, 123
- canonical momentum, 5, 9
- canonical variables, 5
- cavity, 3, 32
- CELL, 103, 104
- CELL, ORBIT, 119
- CENTRE, 11, 42, 64
- CHARGE, 35, 53
- charge, 10, 53
- CHKBELow, 119, 120
- CHROM, 57, 58, 125, 127
- chromatic functions, 7, 57, 127
- chromaticity, 58, 99
 - tuning, 99
- CLASS, 44, 49, 65
- class, 36
- CLEAR, 49
- CLIGHT, 17
- CLORB, 70
- closed orbit, 2, 74
- COFACT, 19
- COGUESS, 51, 53
- collimator, 33

- COLUMNS, 64, 96
- COMMAND, 19
- command, 12
 - attribute, 12, 13, 17
 - continuation, 12
 - dictionary, 75
 - format, 12
 - keyword, 13
 - label, 12, 13
 - option, 18, 20
- commands
 - actions, 51
 - utility, 12
- COMMENT, 132
- comments, 12
- concatenation, 37
- CONJUGATE, 63, 64
- CONSTANT, 16
- constant, 16
- CONSTRAINT, 49, 103, 106, 107
- constraint, 17
 - periodic, 107
 - simple, 106
 - transfer map, 108
- continuation, 12
- CONTINUE, 118, 119
- conventions, 1
- coordinate transformation, 34
- coordinates
 - global, 2, 126
 - initial, 117
 - local, 1
- CORRECT, 86, 87, 89
- correction, accuracy, 89, 90
- correction, iteration, 90
- corrector, 4, 31
- corrector errors, 84
- corrector number, 89, 90
- corrector tables, 89, 90
- COS(X), 15
- COUPLE, 57–59, 103, 107
- coupling, 57, 69, 71
- current, 10, 54, 71
- CYCLE, 43

- D, 120
- DO, 119, 120
- DAMP, 115, 119
- damping, 69
- data
 - pool, 78, 79

- streams, 75–79, 125–133
- DATAVRSN, 125
- DATE, 125, 132
- DBL, 82
- DBLN, 83
- DBLR, 83
- DBLS, 83
- DDPX, 8, 59, 64
- DDPY, 8, 59, 64
- DDX, 8, 59, 64
- DDY, 8, 59, 64
- DEBUG, 19
- default beam, 54
- defects, 124
- deferred expression, 17
- DEFINE, 19
- definition
 - element, 25
 - line, 39
 - parameter, 20
- DEGRAD, 17
- DELQ, 70
- DELTAP, 6, 8, 9, 57–59, 63, 64, 68, 93–95, 104, 117, 119
- DICT, 75
- dictionary, 75
- dipole, 4, 26
- dipole errors, 83
- DISP, 99
- displacement, 3
- DISTANCE, 9
- distortion functions, 100
- DKL(i), 83
- DKLN(i), 83
- DKLR(i), 83
- DKLS(i), 83
- DMUX, 8, 59, 64
- DMUY, 8, 59, 64
- DO, 122, 123
- DOUBLE, 18
- DP, 65
- DPHI, 81
- DPSI, 81
- DPX, 7, 59, 64, 108, 133
- DPY, 7, 59, 64, 108, 133
- DQ1DE1, 64, 109
- DQ1DE2, 64, 109
- DQ1DE3, 64
- DQ2DE2, 64, 109
- DQ2DE3, 64
- DQ3DE3, 64
- DQXDEX, 100, 101
- DQYDEX, 100, 101
- DQYDEY, 100, 101
- DRIFT, 25, 26, 126
- drift, 3
- DS, 81
- DTHETA, 81
- DTUNE, 109, 119, 120
- dump pool, 79
- DX, 7, 59, 64, 81, 87, 108, 132
- DX' I, 101
- DX' S, 101
- DX' I, 101
- DX' S, 101
- DXMAX, 8
- DXWEIGHT, 89, 90
- DY, 7, 59, 64, 81, 87, 108, 133
- DY' I, 101
- DY' S, 101
- DYMAX, 8
- DYNAMIC, 51, 63
- dynamic table, 78
- dynamic tables, 75
- DYNAP, 118, 119
- DYNAPFRAC, 109, 118–120
- DYNAPFRAC, TURNS, 109
- DYNAPSTART, 118–120
- DYWEIGHT, 89, 90
- E, 32
- E , 17
- E1, 27, 65
- E2, 27, 65
- EALIGN, 49, 80
- ECCN1, 61
- ECCN2, 61
- ECHO, 12, 14, 18, 75, 76
- ECOLLIMATOR, 25, 33
- edit
 - elements, 36
- EFCOMP, 49, 80, 82
- EFIELD, 49, 65, 80, 82
- EIGEN, 51, 61
- electric field, 10, 32
- ELECTRON, 53
- electron beam parameters, 69–74
- electrostatic separator, 3, 32
- ELEMENT, 44, 45
- element, 25
 - attribute, 17, 26

- class, 36
- data, 126
- edit, 36
- format, 25
- keyword, 26
- label, 26
- name, 26
- redefinition, 36
- sequence, 52
- type, 26
- elevation angle, 3
- ELSEPARATOR, 25, 32, 126
- EMASS , 17
- EMIT, 51, 61, 115, 117
- emittance, 10
 - coupling, 69
 - horizontal, 53
 - longitudinal, 54
 - normalised, 54
 - vertical, 54
- ENDDO, 122, 123
- ENEDIT, 43
- ENDHARM, 98, 99
- ENDMATCH, 103, 105
- ENDSEQUENCE, 42
- ENDSUBROUTINE, 122, 123
- ENDTRACK, 115, 116
- ENERGY, 53, 54
- energy, 10, 53
 - error, 2
 - spread, 54
- energy error, 6, 8, 9, 57
- ENTRY, 11, 42
- ENVELOPE, 51, 62
- EOPT, 80, 84
- EPRINT, 49, 80, 84
- ERROR, 55, 89, 90
- error
 - alignment, 80
 - corrector, 84
 - definitions, 80–85
 - dipole, 83
 - energy, 6, 8, 9, 57
 - field, 81
 - momentum, 6, 8, 9, 57
 - monitor, 80
 - multipole, 84
 - options, 84
 - printing, 84
 - quadrupole, 84
 - random, 80
 - saving, 85
 - sextupole, 84
- error messages, 75
- ESAVE, 49, 80
- ET, 54
- EVARY, 70
- EX, 53, 54
- examples
 - constraint, 113
 - EALIGN, 66
 - HARMON, 101
 - matching, 111–114
 - OPTICS, 66
 - place, 66
 - PRINT, 66
 - range, 66
 - spin matching, 113
 - SPLIT, 66
 - track, 121
 - transfer map constraint, 113
- EXCITE, 75, 77
- EXDATA, 70
- EXIT, 11, 42
- EXN, 54
- EXP(X), 15
- EXPONENT, 63
- EXPRESS, 19
- expression, 15, 17, 20, 21
- EY, 54
- EYN, 54
- .FALSE., 15
- FASTUNE, 119
- FE, 99
- FFILE, 118
- FFT, 93
- field, 10
 - components, 5
 - electric, 32
 - expansion, 5
 - fringe integral, 27
 - magnetic, 5
- field errors, 81
- file names, 128
- FILENAME, 10, 64, 68, 76–79, 85–87, 93
- files, 75–79, 125–133
- files, user-defined, 128, 130, 131
- FINT, 27
- FIRST, 55
- fit point, 49

- FIX, 103, 106
- FIXED, 63
- FIXPOINT, 109
- FLAG, 55
- flag, 15
- flat machine, 69
- FONT, 91
- formal arguments, 41
- format
 - command, 12
 - element, 25
 - label, 14
 - name, 14
- FORMULA, 109
- FPRINT, 118
- FRACMIN, 119, 120
- FRACTION, 56
- FREQ, 65
- frequency, 10
- FREQUENCY(i), 116
- fringe field, 27
- FROM, 44
- FT, 117
- FULL, 49
- function, 15
- functions
 - chromatic, 7, 57, 127
 - distortion, 100
 - lattice, 7, 57, 91, 127
- FX, 70, 73, 98, 117
- FX, PHIX, FY, PHIY, FT, PHIT, 119
- FY, 70, 73, 99, 117

- GAMMA, 53
- GAMMATR, 8
- GAMX, 7
- GAMY, 7
- garbage removal, 78
- GAUSS(), 16
- geometry, 56, 126
- GETDISP, 86, 89
- GETKICK, 86, 87
- GETORBIT, 86, 87, 89
- GKSGRAL, 92
- GLOBAL, 103, 109, 119
- global
 - coordinates, 2
 - reference, 2
- global axes, 3
- gradient minimisation, 110, 111
- grep, 9

- GWEIGHT, 103, 109
- GXPLOT, 91

- H1, 28, 65
- H2, 28, 65
- HAMILTON, 63
- HARMON, 32, 65, 98
- HARMON, 98–102
- HAXIS, 92, 94, 95
- HCELL, 98, 100, 101
- HCHROMATICITY, 98, 99
- HELP, 12–14
- HFUNCTION, 98
- HFUNCTIONS, 100
- HGAP, 28
- high order modes, 10
- HKICK, 31, 65
- HKICKER, 25, 31, 88, 126
- HMAX, 93
- HMIN, 93
- HMONITOR, 25, 33, 88, 126
- horizontal invariant, 6
- horizontal phase, 6
- horizontal position, 5, 6, 9
- HRESONANCE, 98, 99
- HTUNE, 98, 99
- HVARY, 98–101
- HWEIGHT, 98, 101

- I4I2, 70
- IBM, 128
- IBS, 51, 60
- identifier format, 14
- impedance, 10
- INCREMENT, 75, 77
- INFO, 19
- initial conditions, 117
- initial values, lattice, 58, 59
- INPUT, 75
- input, 75
 - lines, 12
- insertion matching, 104
- INSTALL, 44, 49
- INSTRUMENT, 25, 33
- integer, 15
- INTER, 18
- INTERACT, 70, 71
- intra-beam scattering, 60
- INVARIANT, 63
- invariant
 - horizontal, 6

- longitudinal, 6
 - vertical, 6
- ITERATE, 90
- JOBNAME, 125
- K_N_H, 87
- K_N_V, 87
- KOL, 65
- K1, 27, 29
- K2, 28, 29
- K3, 29, 30
- KAPPA, 69, 70
- KBUNCH, 54
- KEYWORD, 19, 65, 126
- keyword, 13
 - element, 26
 - format, 14
 - help, 13
- KHM, 70
- KICK, 31
- KICKER, 25, 31, 88, 126
- KnL, 30, 65
- KS, 31, 65
- KSOL, 65
- KSnL, 65
- L, 26, 27, 29–33, 65
- LABEL, 11, 60, 63
- label, 12, 13
 - element, 26
 - format, 14
 - line, 39
 - list, 47
 - sequence, 42
 - show, 14
- label, 12
- LAG, 32, 65, 73
- lag
 - time, 9
- lag (time), 6
- lattice functions, 7, 57, 91, 127
- lattice parameters, 59
- length, 10
- LEVEL, 103, 110
- LIE, 56
- LIE3, 118
- LIE4, 118
- limits
 - matching, 106
- LINE, 19, 39, 58, 62, 64
- line, 39
 - label, 39
 - reference, 47
- lines, 39–50
- LIST, 47
- list
 - label, 47
 - replacement, 47
 - restrictions, 48
- LMDIF, 103, 110
- local
 - origin, 2
 - reference, 2
- local axes, 3
- local coordinates, 1
- LOG(X), 15
- LOGDIST, 9
- logical, 15
- LOGTURNS, 9
- longitudinal invariant, 6
- LOWER, 100, 106
- lower case, 12
- LRAD, 30
- LSCALE, 92
- luminosity, 70, 71
- LUMP, 25, 37
- LWIDTH, 92
- LYAPUNOV, 9, 109, 119, 120
- lyapunov, 120
- LYAPUNOV=real, 119
- M1LIST, 89, 90
- M2LIST, 90
- machine parameters, 71
- magnetic field, 5
- main beam line, 51
- MAKESEQUENCE, 11
- MAP, 63
- MARKER, 4, 25, 26, 126
- MASS, 53
- mass, 10, 53
- MATCH, 103–105
- matching, 103–114
 - limits, 106
 - methods, 110, 111
 - output, 110
 - weight, 108
- MATRIX, 25, 36
- MAX(X,Y), 16
- MAXPLOT, 93, 94
- METHOD, 118, 119

- method (tracking), 118
- MICADO, 86, 87, 89
- MIDARC, 70, 73
- MIGRAD, 103, 110, 111
- MIN(X,Y), 16
- misalignment errors, 80
- MODE, 88
- momentum, 6, 10, 53
 - canonical, 5, 9
 - error, 2
 - normalised, 6
- momentum error, 6, 8, 9, 57
- MONITOR, 25, 33, 88, 126
- monitor, 4, 33
- monitor errors, 80
- monitor tables, 89, 90
- MOVE, 44, 49
- MREX, 80, 81
- MREY, 80, 81
- MULTIPLE, 93, 95
- MULTIPOLE, 25, 30, 81, 82, 99, 100, 126
- multipole, 10
- multipole coefficients, 5
- multipole errors, 84
- MUX, 7, 57–59, 64, 104, 107, 108
- MUY, 7, 57–59, 64, 104, 107, 108

- N, 63
- NAME, 10, 64, 77, 96, 100, 106, 126
- name, 14
 - format, 14
 - show, 14
- NCORR, 89, 90
- NINT, 70
- NOISE, 115, 116
- noise, 116
- NONE, 89, 90
- NONLINEAR, 63, 64
- NORMAL, 63, 97
- normalised emittance, 54
- normalised variables, 6
- NPART, 54
- NPOS, 125
- NTMAX, 96

- observation point, 49
- observation points, 50
- OBSERVE, 115, 116
- OCTUPOLE, 25, 30, 126
- octupole, 3
- ONEPASS, 115

- operand, 16
- operating system command, 23
- operator, 15
- OPTICS, 55, 56, 64, 132
- OPTION, 12
- option
 - command, 20
- ORBIT, 63, 104–107
- orbit
 - closed, 2, 74
 - reference, 1
- orbit correction, 86–90
- orbit corrector, 31
- ORDER, 11, 82, 85, 99
- orientation, 3
- ORIGIN, 132
- output, 75
- output frequency, 118

- PACKMEMORY, 75, 78
- page header, 18
- PARAM, 93–95
- parameter, 20–21
 - label, 16
 - name, 18, 20
 - RF, 72
 - value, 21
- PARTICLE, 53, 54, 93–95
- particle
 - bank, 118
 - charge, 53
 - energy, 53
 - mass, 53
 - momentum, 53
- PATTERN, 9, 49, 76
- PC, 53
- PERIOD, 52
- periodic solution, 58
- PG, 32
- phase, 57, 107
 - horizontal, 6
 - vertical, 6
- phase advance, 107
- PHASE(i), 116
- PHI, 3
- PHIO, 57
- PHIT, 6, 117
- PHIX, 6, 8, 59, 64, 117
- PHIY, 6, 8, 59, 64, 117
- physical element, 25
- physical units, 9

- PI , 17
- PLACE, 60, 63, 116
- place examples, 66
- PLANE, 87, 89, 90
- PLOT, 91, 92, 95
- plot, 75, 91–97
- PMASS , 17
- pool, 78
 - dump, 79
 - reload, 79
- POOLDUMP, 75, 79
- POOLLOAD, 75
- position
 - horizontal, 5, 6, 9
 - vertical, 5, 6, 9
- POSITRON, 53
- POST, 92
- potential, 5
- power, 10
- power supply noise, 116
- PRINT, 49, 51, 55, 75, 76
- print frequency, 118
- PROBLEM, 129
- procedure, 122
- PROGVRSN, 125
- PROTON, 53
- PSI, 3
- PSIO, 57
- PT, 6, 59, 64, 107, 108
- PTO, 58, 61
- PTEND, 109, 119
- PTN, 6
- PTSTART, 118, 119
- PUNAME, 86, 88
- PUTDISP, 86
- PUTKICK, 86, 87
- PUTORBIT, 86, 87
- PX, 5, 9, 59, 64, 95, 108, 117, 132
- PXO, 58, 61
- PXEND, 109, 119
- PXN, 6
- PXSTART, 118–120
- PY, 5, 9, 59, 64, 108, 117, 132
- PYO, 58, 61
- PYEND, 109, 119
- PYN, 6
- PYSTART, 118, 120
- Q1, 61, 64, 109
- Q1', 64, 109
- Q1'', 64, 109
- Q1'', Q2'', DQ1DE1, DQ1DE2, DQ2DE2, 109
- Q1, Q2, Q1', Q2', 109
- Q2, 61, 64, 109
- Q2', 64, 109
- Q2'', 64, 109
- Q3, 61
- QCONDi, 96
- QS, 116
- QUADRUPOLE, 25, 29, 126
- quadrupole, 3, 10
- quadrupole errors, 84
- QUANTUM, 115
- QX, 8, 58
- QX', 58, 99–101
- QX'', 100, 101
- QX''', 100, 101
- QY, 8, 58
- QY', 58, 99–101
- QY'', 100, 101
- QY''', 100, 101
- RADDEG, 17
- RADIATE, 54
- radiation, 54, 73
- RADIUS, 82, 85
- RADLOSS, 65
- random
 - generator, 16
 - seed, 84
- random error, 80
- random values, 17
- RANF(), 16
- RANGE, 49, 52, 70, 73, 81, 93, 94, 107, 108
- range, 49, 81
- range examples, 50, 66
- RBEND, 25, 26, 99, 100, 126
- RBETX, 94
- RBETY, 94
- RBXMAX, 94
- RBymax, 94
- RCOLLIMATOR, 25, 33
- real, 15
- redefinition, 36
- REFER, 11, 42, 55
- reference
 - beam line, 47
 - orbit, 1
 - system, 1, 4
- REFLECT, 44
- reflection, 40
- relation, 20

- relativistic, 6
- reload pool, 79
- REMOVE, 44, 49
- repetition, 40
- REPLACE, 45, 49
- replacement list, 47
- RESET, 18
- RESONANCE, 63
- resonances, 99, 100
- RESPLOT, 91
- restrictions on lists, 48
- RETRIEVE, 64, 75, 78
- RETURN, 75, 76
- REXX, 129
- RF, 10
- RF cavity, 3
- RF parameters, 72
- RFCAVITY, 25, 32, 63, 126
- Rij, 62
- RM(i, k), 36, 108
- RMATRIX, 103, 108
- roll angle, 3
- ROT(i), 83
- rotation, 4, 34
- rotation angle, 2
- RSIZE, 92
- RUN, 9, 115, 117, 118
- RXI, 101
- RXS, 101
- RYI, 101
- RYS, 101

- S, 6, 65, 93–95
- SAVE, 57–59, 62, 75, 76
- SAVEBETA, 60, 105
- SAVESIGMA, 63
- SBEND, 25, 26, 99, 100, 126
- scattering (intra-beam), 60
- SCRATCH, 129
- SECOND, 55
- SECTORMAP, 51, 68
- SEED, 84
- seed (random generator), 84
- SELECT, 49, 51, 55, 56, 68
- SELECT, FLAG=ERROR, 49, 80, 82
- SELECT, FLAG=SEQEDIT, 44
- SELECTED, 44, 45
- selection examples, 50
- selection point, 49
- separator, 3, 32
- SEQEDIT, 43, 55
- SEQUENCE, 39, 42, 43
- sequence, 43, 52
 - label, 42
- SET, 12, 20
- SETPLOT, 91, 93
- SEXTUPOLE, 25, 29, 99, 100, 126
- sextupole, 3
- sextupole errors, 84
- SHOW, 12, 14
- SHUNT, 32
- SIGE, 54
- SIGMA0, 62
- SIGT, 54
- SIGX, 7, 35
- SIGxx, 62
- SIGY, 7, 35
- simple beam line, 39
- SIMPLEX, 103, 111
- SIN(X), 15
- SINGLE, 70, 73
- SMEAR, 109, 119, 120
- SOLENOID, 25, 31, 126
- solenoid, 3
- SORT, 93
- SPLINE, 93
- SPLIT, 49, 56
- SQRT(X), 15
- SROT, 25, 34, 126
- SSCALE, 92, 93
- START, 115, 117–119
- statement, 12
- STATIC, 51, 63
- STATUS, 75, 78, 87, 88
- STEP, 100, 106, 120
- step, 119
- step size, 106
- STOP, 12, 18
- STR_NAME, 87
- STRATEGY, 111
- streams, 75–79, 125–133
- STRING, 91, 96, 120
- string, 14
- STRUCTURE, 125, 126
- STYLE, 92
- sub-line, 40
- SUBROUTINE, 122, 123
- subroutines, 122–123
- substitution list, 47
- SUM, 96
- SUPER, 52, 58, 59, 125

- super-period, 52
- SURVEY, 51, 56, 125, 126
- SYMBOL, 93
- symbolic constant, 16, 17
- SYMM, 52, 58, 59, 125
- SYMPL, 19
- synchrotron
 - integrals, 71
 - radiation, 54, 73
- synchrotron tune, 116
- SYNRAD, 70
- SYSTEM, 12, 23

- T, 6, 9, 59, 63, 64, 107, 108, 117
- TO, 58, 61
- TABLE, 60, 78, 91, 93, 96, 116, 118–120
- table
 - TFS, 87
 - track, 9, 93, 95, 118
 - tunes, 95
 - twiss, 93, 94
- tables, 75
- TAN(X), 15
- TAPE, 57–59, 125
- TAUQ, 70
- TELL, 19
- TEND, 109, 119
- TERMINAL, 76
- TFILL, 32
- TFS, 133
- TFS files, 132
- TFS table, 87
- TGAUSS(X), 16
- THETA, 2
- THETAO, 56
- TILT, 28–32, 65, 84
- tilt angle, 3
- TILT1, 61
- TILT2, 61
- TIME, 125, 132
- time lag, 6, 9
- TITLE, 12, 18, 93, 125
- TM(i, k, l), 36, 108
- TMATRIX, 103, 108
- TN, 6
- Tn, 30
- TO, 44
- TOLERANCE, 53, 99, 111
- Tolerance, 100
- total energy, 53
- TOUSCH, 70

- TRACE, 18
- TRACK, 9, 55, 115, 117, 132
- track table, 9, 93, 95, 118
- tracking, 115–121
 - method, 118
- transfer map constraints, 108
- TRANSPORT, 118, 119
- .TRUE., 15
- TSAVE, 115, 117
- TSTART, 118
- tune, 58
- TUNES, 8, 58, 132
- tunes table, 95
- TUNX, 109, 119
- TUNY, 109, 119
- TURNS, 93–95, 109, 118–120
- TWISS, 7, 8, 51, 55, 57–60, 93, 125, 127, 132
- twiss table, 93, 94
- TWISS3, 51, 62
- TWOPI, 17
- TYPE, 26, 65, 81, 126, 132

- UNICOS, 129
- units, 5, 9
- UNIX, 129
- UNIX, 23
- UNUSED, 88
- UPPER, 100, 106
- UPPER CASE, 12
- USE, 43, 51, 55, 86, 115
- USED, 88–90
- USEKICK, 49, 86, 88
- USEMONITOR, 49, 86, 88
- user files, 128, 130, 131
- USER0(), 16
- USER1(X), 16
- USER2(X, Y), 16
- utility commands, 12

- VALUE, 12, 21, 77
- VARIABLE, 116
- variable, 18, 105
- variables, 5
 - canonical, 5
 - normalised, 6
- VARY, 103, 105
- VAX VMS, 130
- VAXIS, 92
- VAXIS1, 92
- VAXIS2, 92
- VAXIS3, 92

VAXIS4, 92
vector potential, 5
VERIFY, 19
vertical bar, 12
vertical invariant, 6
vertical phase, 6
vertical position, 5, 6, 9
VKICK, 31, 65
VKICKER, 25, 31, 88, 126
VM/CMS, 23, 128
VMAX, 93
VMIN, 93
VMONITOR, 25, 33, 88, 126
VOLT, 32, 65
voltage, 10

WARN, 19
WEIGHT, 103, 107, 108
weight (matching), 108
wild-card, 9, 76
working beam line, 51
WT, 6
WX, 6, 7, 59, 64
WXMAX, 109, 119, 120
WXMIN, 109, 119, 120
WXSTART, 119
WXYMAX, 109, 119, 120
WXYMIN, 109, 119, 120
WXYSTART, 119
WY, 6, 8, 59, 64
WYMAX, 109, 119, 120
WYMIN, 109, 119, 120
WYSTART, 119

X, 2, 5, 9, 59, 64, 88–90, 95, 108, 117, 132
X, PX, Y, PY, T, DELTAP, 119
XO, 56, 58, 61
XEND, 109, 119
XIX, 8
XIY, 8
XMA, 35
XMAX, 8
XN, 6
XRMS, 8
XSIZE, 33, 91
XSTART, 118–120

Y, 2, 5, 9, 59, 64, 88–90, 108, 117, 132
YO, 56, 58, 61
YEND, 109, 119
YMA, 35
YMAX, 8
YN, 6
YRMS, 8
YROT, 25, 34, 126
YSIZE, 33, 91
YSTART, 118, 120

Z, 2
ZO, 56