MAD-X – AN UPGRADE FROM MAD8

H. Grote and F. Schmidt, CERN, Geneva, Switzerland

INTRODUCTION

During the central period of the LHC design the accelerator design group at CERN had to face a serious bottleneck concerning our optics design code. Since years we had been using the MAD8 [1] code at CERN and it is still heavily used world wide. This old code has been very well debugged and most of the MAD8 performed rather well. However, due to the lack of essential features for the LHC design, e.g. operating on 2 rings simultaneously, involved Zebra [2] data bank management and the fact that the modules are rather interdependent larger upgrades of the code MAD8 are basically excluded, in particular if one wants to achieve a better physics description of the thick elements and make use of more modern map related tools à la Berz-Forest [3]. On the other hand, the C++ implementation used for the MAD9 program [4] turned out to be too complex for a fast development as it was needed for the LHC design. After a lengthy and frustrating trial period with MAD9 it was decided to suspend the MAD9 development and to start a new project called MAD-X with the following design criteria:

- Core part in C with dynamic memory allocation.
- Truly independent modules with interfaces to the core for data access.
- Make use of existing and debugged modules of MAD8 in Fortran77.
- Retain only those features of MAD8 that are sound and concentrate on those modules that are needed for the LHC design.
- Spread responsibility for development and maintenance of modules between a large group of module keepers inside and outside of our group organized by one code custodian.
- Use E.Forest's PTC [5] as external module to provide map techniques and better physics description of the elements.
- CVS version management.
- Adding powerful constructs to the input language like: WHILE and IF .. ELSE .. ENDIF .

Presently, we are at MAD-X version V1.11 and all modules and features needed for the LHC design are debugged and tested. There is a rather complete documentation on the web together with code, binaries and examples.

In the following we will present the module keepers and describe the documented modules. The major extension of MAD-X by the PTC code will be outlined. A simple example will be given and it will described what kind of documentation can be obtained. Lastly, an outlook will be given about the next steps in the development of MAD-X.

MODULE KEEPERS

Table 1: Module Keepers

Module	Keeper	Tested	Docu.	Exam.
C6T	M.Hayes F.Schmidt	yes	yes	yes
CORORBIT	W.Herr	yes	yes	yes
DYNAP	F.Zimmer– mann	yes	yes	yes
EMIT	H.Grote R.Assmann	no	no	no
ERROR	W.Herr	yes	yes	yes
IBS	D.Brandt	yes	yes	yes
MAKETHIN	M.Hayes H.Burkhardt	yes	yes	yes
MATCH	O.Brüning	yes	yes	yes
PLOT	H.Grote T.D'Amico	yes	yes	yes
SURVEY	A.Verdier	yes	yes	yes
SXF	H.Grote F.Pilat	no	yes	no
THREADER	H.Grote T.Risselada	no	no	no
TWISS	F.Schmidt	yes	yes	yes
THINTRACK	A.Verdier	yes	yes	yes

In Tab. 1 a list is given of all modules with their original and present module keepers depicted in "blue" and "green" respectively. Moreover, we have one first example of an external module, i.e. the F.Pilat from BNL who has agreed to look after the SXF [6] module. The table also shows which modules have been sufficiently tested and for which modules there are documentation and/or examples.

THE DOCUMENTED MAD-X MODULES

C6T

In dynamic aperture studies SixTrack [7] is often used because of its speed and controllability. However, the input files are notoriously difficult to produce by hand. This command may be used to produce SixTrack input files from any MAD-X preparation file.

CORORBIT

This module can be used to correct the closed orbit or a trajectory. The distorted orbit, the model and if required the target orbit are calculated with one or more TWISS commands.

DYNAP

For each previously entered start command, DYNAP tracks two close–by particles over a selected number of turns, from which it obtains the betatron tunes with error, the action smear, and an estimate of the Lyapunov exponent.

ERROR

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 constant or random values. They may be entered after having selected a beam line or sequence by means of a USE command.

MAKETHIN

This module converts a sequence with thick elements into one composed entirely of drifts and thin multipole elements as required e.g. by the default MAX-X tracking.

MATCH

Before a match operation at least one sequence must be selected by means of a USE command. Matching is then initiated by the MATCH command. The matching module can act on more than one sequence simultaneously by specifying more than one sequence when initiating the matching mode. From this command to the corresponding ENDMATCH command MAD accepts various matching commands.

PLOT

Values contained in MAD-X tables can be plotted in the form column versus column, with up to four differently scaled vertical axes; furthermore, if the horizontal axis is the position ”s” of the elements in a sequence, then the symbolic machine can be plotted on the top of the figure. The ”environment” (line thickness, annotation size, PostScript format) can be set with the setplot command.

SURVEY

The SURVEY command computes the coordinates of all machine elements in a global reference system. These coordinates can be used for installation. In order to produce coordinates in a particular system, the initial coordinates and angles can be specified. The computation results are always written to an internal table but they can also be written to an external file.

SXF

Read and writes SXF format[6] from or to the currently USEd sequence with all alignment and field errors.

TWISS

The TWISS command causes computation of the closed orbit and of the coupled Courant and Snyder [8] linear lattice functions, and optionally of the chromatic functions, either as the periodic solution or starting with initial values of the lattice functions. It operates on the working beam line defined in the latest USE command: i.e. either a SEQUENCE=”sequence_name” or a LINE=”line_name” on the TWISS command. Moreover, one can restrict the TWISS calculation to a desired RANGE.

THINTRACK

Particle trajectories can be tracked either for single passage (option onepass in the command), or for many turns (default option). In all cases the tracking is performed element per element. Only thin elements are allowed, which guarantees the symplecticity of the coordinate transformation. Any lattice can be converted into a “thin element” lattice by invoking the MAKETHIN command.

PTC

E.Forest’s Polymorphic Tracking Code (PTC[5]) is a kick code or symplectic integrator and therefore ideally suited to describe all elements symplecticly and to arbitrary exactness. The degree of exactness is determined by the user and the speed of his computer. The code is written in an object oriented fashion using Fortran90. Therefore, it becomes much easier to describe arbitrarily complex accelerator structures. The other main advantage is that the code is inherently based on map formalism [3] and a linking with MAD-X will provide all the sophisticated tools, e.g. Normal Form. There is already an experimental MAD-X version that calculates successfully the fully 6d coupled lattice functions like in MAD8 but using PTC tools. MAD-X version 2.0 will include PTC and then many additional modules are conceivable that are based on this tool.

DOCUMENTATION

The MAD-X website [9] is accessible either via a “google” search for “CERN MAD-X” or directly via: “http://frs.home.cern.ch/frs/Xdoc/mad-X.html”. On this website you can find a “News” link which shows the changes between versions, the documentation based on “html” files and derived from them a “ps” and a “pdf” version, a “Keyword and Subject Index”, a link to “Source and binaries” and one link to “Examples” for all modules and a facility to report bugs found in MAD-X by its users. Lastly, it is planned to provide a MAD-X mailing list “Subscribe” button.

EXAMPLE

As an example a simple FODO lattice has been used: it shows how elements are defined and located in a sequence. Notice the two symbols "=" and ":=", the former evaluates the variable with the present value while in the second form this evaluation is deferred to the moment when it is needed and with the momentary value. The example shows how a "WHILE" loop is used to build the structure of "ncell" cells. After applying the "BEAM" command, "USE"ing the sequence and "SETPLOT"ing the plot parameters "TWISS" is executed twice. The first "TWISS" is used to produce Fig. 1 with the "PLOT" command. The second time "TWISS" is done for a number of momentum deviations

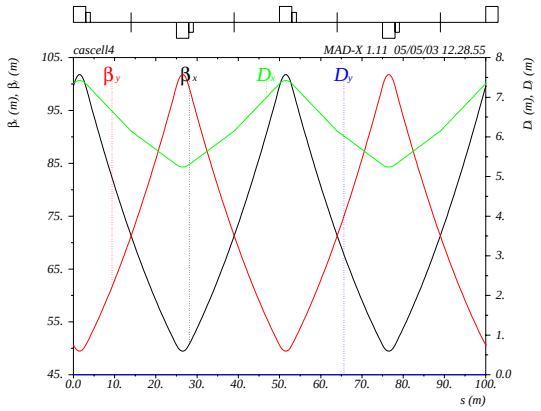

Figure 1: Lattice function for a simple FODO structure.

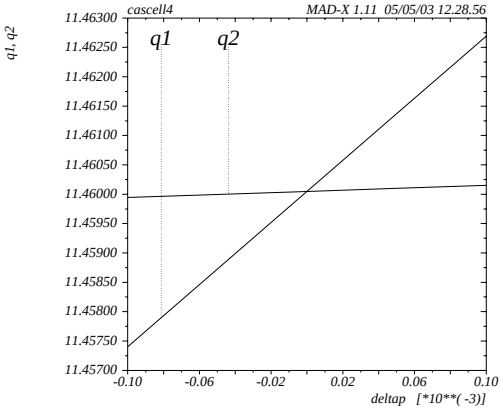

Figure 2: Mode I and II Tunes versus momen. deviation.

```
// element definitions;
lsex = 1.1; mq: quadrupole, l=3.0;
mb3:multipole, lrad:= 5.0,
knl:={.031415927}; qf: mq, k1:=kqf;
qd: mq,k1:=kqd; mscbh:
sextupole, l=lsex, k2:=ksf;
mscbv: sextupole, l=lsex, k2:=ksd; kqf
= .9795014e-2; kqd=-.9795014e-2;ksf=.01; ksd=-.01;
circum=5000.0;ncell=100;
lcell=circum/ncell;lcell2=lcell/2.;lquad=3.00;
lquad2=lquad/2.;lsex=1.1;
// sequence declaration;
cascell4: sequence, refer=centre, l=circum;
start_machine: marker, at = 0;
n = 1;
while (n < ncell+1) {
  qf:qf,at=(n-1)*lcell+0.00*lcell+lquad2;
  mscbh:mscbh,at=(n-1)*lcell+0.00*lcell+lquad+lsex/2.0;
  mb3:mb3,at=(n-1)*lcell+0.25*lcell+lquad2;
  qd:qd,at=(n-1)*lcell + 0.50*lcell+lquad2;
  mscbv:mscbv,at=(n-1)*lcell+0.50*lcell+lquad+lsex/2.0;
  mb3:mb3,at=(n-1)*lcell+0.75*lcell+lquad2;
  n=n+1;
}
end_machine: marker at=circum;
endsequence;
// using sequence
eg:=100;bg:=eg/pmass;
en:=3.75e-06;epsx:=en/bg;epsy:=en/bg;
beam, sequence=cascell4,
  particle=proton,energy=eg,sigt=0.077,
  sige=1.1e-4,npart=1.05e11,exn=4*en,eyn=4*en,
  kbunch=10,et=0.002,bv=1,ex=epsx,ey=epsy;
setplot,post=2,ascale=1.5,lscale=1.5,
  rscale=1.5,sscale=1.5,lwidth=3;
use,period=cascell4;
twiss;
plot,haxis=s,hmin=0.,hmax=100.,spline,vaxis1=betx,
  bety,vaxis2=dx,dy,colour=100;
twiss,deltap=-0.0001:0.0001:0.00005;
plot,table=summ,haxis=deltap,vaxis=q1,q2;
stop;
```

OUTLOOK

At version V1.11 MAD-X is now advanced enough to be presented as CERN's official replacement for MAD8. MAD-X has the most useful features of MAD8 but those features that were flawed and difficult to improve have been removed. On the other hand, MAD-X is now so much better organized into independent modules which are nicely interfaced with the "C" core that external modules can easily be added and maintained. The proper integration of MAD-X with the very versatile map based tool PTC is imminent. A release of MAD-X for the 32-bit Windows platform is in preparation. For fall 2003 a review is planned to discuss how MAD-X should be developed further to bring best service to the accelerator community.

REFERENCES

[1] H.Grote and F.C. Iselin, CERN/SL/90–13(AP) (Rev. 5), http://hansg.home.cern.ch/hansg/mad/mad8/mad8.html.

[2] R. Brun, M. Goossens and J. Zoll, Program library Q100, CERN, 1993.

[3] M. Berz, É. Forest and J. Irwin, Part. Accel., 1989, Vol. 24.

[4] F.C. Iselin, J. Jowett, J. Pacin and A. Adelmann, "MAD Version 9", in the proceedings of EPAC2000;

[5] E. Forest, E. McIntosh and F. Schmidt, KEK Report 2002–3, CERN–SL–2002–044 (AP).

[6] H. Grote, et al., RHIC/AP/155.

[7] F. Schmidt, CERN SL/94–56 (AP) (1994) (Rev.11.2001, http://cern.ch/Frank.Schmidt/Documentation/doc.html.

[8] E.D. Courant and H.S. Snyder, Ann. of Phys., 3, 1958.

[9] H. Grote and F. Schmidt, http://frs.home.cern.ch/frs/Xdoc/mad-X.html.