Introduction to MAD-X

G. Sterbini, CERN

Inspired by W. Herr's material

20 January 2020, Archamps

guido.sterbini@cern.ch

THE MAD-X LECTURES

We will have

- 1 h lecture (now).
- 6 h "hand-on" tutorials during the week.
  - Today's tutorials (2× 1 h) will be dedicated to get familiar with the MADX environment, to prepare a very simple input file and to explore a simple lattice cell.
  - Tomorrow's tutorials (2× 1 h) will be devoted to the FODO lattice and transfer lines.
  - On Friday's tutorials (2× 1 h) we will play with chromaticity and the LHC lattice.

Each tutorial is split in two parts of ≈ 20 min each (last 20 minutes for Q&A). Basic knowledge of Linux is assumed but do not hesitate to ask in case: we (Andrea, Guido, Hector and Nuria) are here to help.

MAD-X IN <60M:00S!

Introduction

MAD-X syntax

"Hello World!" example

DISCLAIMER. This material is intended to be an introduction to MAD-X: a large part of the code capabilities are not discussed in details or are not discussed at all! We will use MAD-X to "visualise" the transverse dynamics concepts. The main goal here is to help you to be exposed to the beam dynamics from a new perspective.

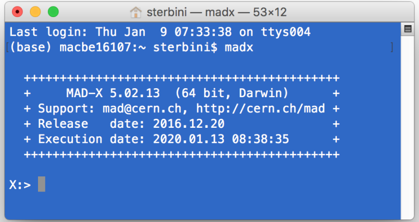
If you want to deepen the subject you can find a lot of material on the web (i.e., here[1])...

- googling "madx", you get the MAD-X homepage.
- To wet your appetite, you can google "MAD-X primer".
- To go in details, you can google "MAD-X manual".

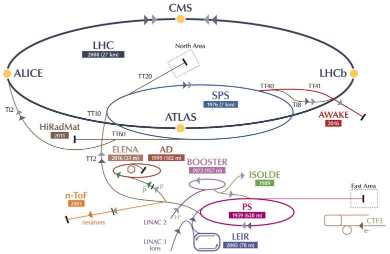
[1] http://madx.web.cern.ch/madx/releases/last-rel/madxuguide.pdf

What is MAD-X?

- A general purpose beam optics and lattice program distributed for free by CERN.
- It is used at CERN since more than 25 years for machine design and simulation (PS, SPS, LHC, linacs. . .).
- MAD-X is written in C/C++/Fortran77/Fortran90 (source code is available under CERN copyright).



A GENERAL PURPOSE BEAM OPTICS CODE



For circular machines, beam lines and linacs…

- Describe/document optics parameters from machine description.
- Design a lattice for getting the desired properties (matching).
- Simulate beam dynamics, machine imperfections and machine operation.

A GENERAL PURPOSE BEAM OPTICS CODE

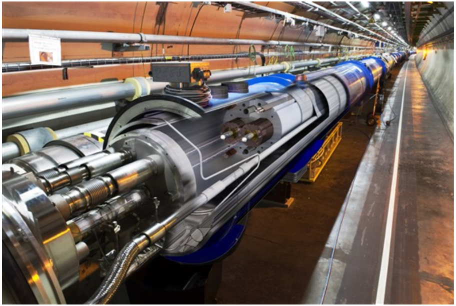
MAD-X is

- multiplatforms (Linux/OSX/WIN...),
- very flexible and possible to extend,
- made for complicated applications, powerful and rather complete,
- mainly designed for large projects (LEP, LHC, CLIC...).

MAD-X is NOT

- a program for teaching,
- (very) easy to use for beginners,
- coming with a graphical user interface.

In large projects (e.g., LHC):

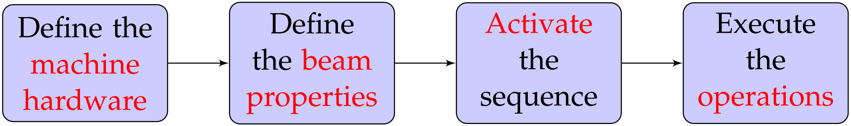


- Must be able to handle machines with $\geq\ 10^4$ elements,
- many simultaneous MAD-X users (LHC: more than 400 around the world): need consistent database,
- if you have many machines: ideally use only one design program.

DESCRIBE AN ACCELERATOR IN MAD-X

Goals. . .

- Describe, optimize and simulate a machine with several thousand elements eventually with magnetic elements shared by different beams, like in colliders.



MAD-X LANGUAGE

How does MAD-X get this info? Via text (interpreter).

- It accepts and executes statements, expressions. . . ,
- it can be used interactively (input from command line) or in batch (input from file),
- many features of a programming language (loops, if's,. . .).

All input statements are analysed by a parser and checked.

- E.g. assignments: properties of machine elements, set up of the lattice, definition of beam properties, errors. . .
- E.g. actions: compute lattice functions, optimize and correct the machine. . .

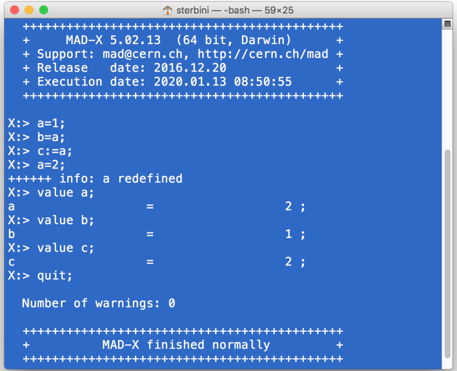
MAD-X INPUT LANGUAGE

- Strong resemblance to "C" language (but NO need for declarations and NOT case sensitive apart in expressions in inverted commas),
- free format, all statements are terminated with **;** (do not forget!),
- comment lines start with: **//** or **!** or is between **/***. . . ***/**,
- Arithmetic expressions, including basic functions (**exp**, **log**, **sin**, **cosh**. . .), built-in random number generators and predefined constants (speed of the light, e, π, m_p, m_e. . .).

In particular it is possible to use deferred assignments

- regular assignment: **a = b**, if **b** changes **a** does not,
- deferred assignment: **a := b**, if **b** changes **a** is updated too.

Example: deferred assignments



We use the **value** command to print the variables content.

DEFINITIONS OF THE LATTICE ELEMENTS

Generic pattern to define an element:

label: *keyword*, properties. . . ;

- For a dipole magnet:
  MBL: SBEND, L=10.0;
- For a quadrupole magnet:
  MQ: QUADRUPOLE, L=3.3;
- For a sextupole magnet:
  MSF: SEXTUPOLE, L=1.0;

In the previous examples we considered only the L property, that is the length in meters of the element.

THE STRENGTH OF THE ELEMENTS

The name of the parameter that define the normalized magnetic strength of the element depends on the element type.

- For dipole (horizontal bending) magnet is k_0:

$$k_0 = \frac{1}{B\rho} B_y \left[\text{in m}^{-1}\right]$$

- For quadrupole magnet is k_1:

$$k_1 = \frac{1}{B\rho} \frac{\partial B_y}{\partial x} \left[\text{in m}^{-2}\right]$$

- For sextupole magnet is k_2:

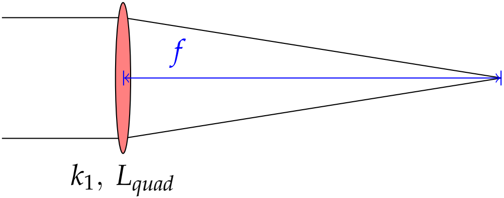
$$k_2 = \frac{1}{B\rho} \frac{\partial^2 B_y}{\partial x^2} \left[\text{in m}^{-3}\right]$$

Interlude

What does k_1 mean? It is related to the quad focal length [2].

$$\frac{1}{k_1 \, L_{quad}} = f \qquad (1)$$

Assuming $k_1 = 10^{-1}$ m^{-2} and $L_{quad} = 10^{-1}$ m the $f = 10^2$ m.



[2]thin lens approximation

Example: definitions of elements

- Sextupole magnet:

  ksf = 0.00156;

  MSF: SEXTUPOLE, K2 = ksf, L=1.0;

- Multipole magnet "thin" element:

  MMQ: MULTIPOLE, KNL = $\{k0 \cdot l, k1 \cdot l, k2 \cdot l, k3 \cdot l, \ldots\}$;

- LHC dipole magnet as thick element:

  length = 14.3;

  p = 7000;

  angleLHC = 8.33 * clight * length/p;

  MBL: SBEND, ANGLE = angleLHC;

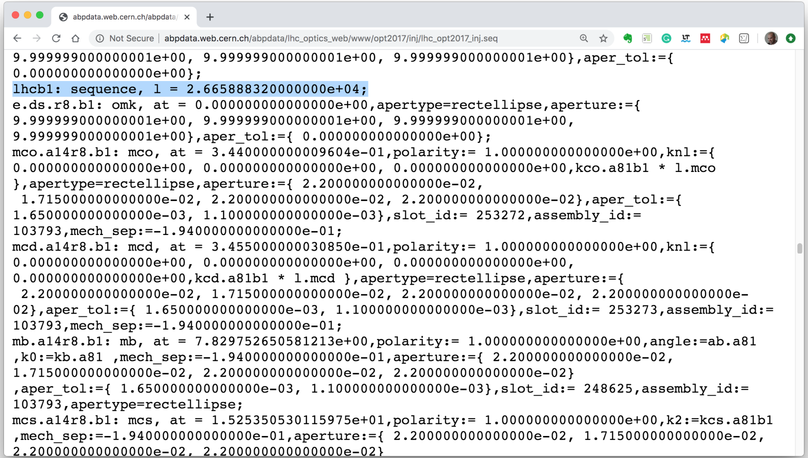
The lattice sequence

A lattice sequence is an ordered collection of machine elements. Each element has a position in the sequence that can be defined wrt the CENTRE, EXIT or ENTRY of the element and wrt the sequence start or the position of an other element:

label: SEQUENCE, REFER=CENTRE, L=length;
...;
...;
...here specify position of all elements...;
...;
...;
ENDSEQUENCE;

Example of sequence: LHC (too tough?)

Check this link!



Beam definition & sequence activation

Generic pattern to define the beam:

label: BEAM, PARTICLE=x, ENERGY[3]=y,...;

e.g., BEAM, PARTICLE=proton, ENERGY=7000;//in GeV

After a sequence has been read, it can be activated:

USE, SEQUENCE=sequence_label;

e.g., USE, SEQUENCE=lhc1;

The USE command expands the specified sequence, inserts the drift spaces and makes it active.

[3] It is the TOTAL energy!

Definition of operations

Once the sequence is activated we can perform operations on it.

- Calculation of Twiss parameters around the machine (very important) in order to know, for stable sequences, their main optical parameters.
  TWISS, SEQUENCE=sequence_label;//periodic solution
  TWISS, SEQUENCE=sequence_label, betx=1;//IC solution

- Production of graphical output of the main optical function (e.g., β-functions):
  PLOT, HAXIS=s, VAXIS=betx,bety;

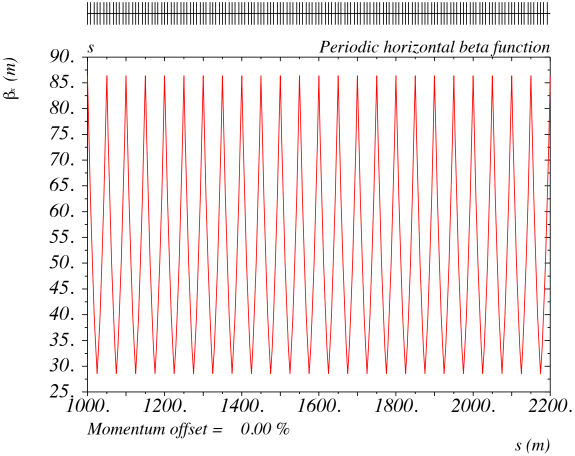
Example

```
TWISS, SEQUENCE=juaseq, FILE=twiss.out;
PLOT, HAXIS=s, VAXIS=betx, bety, COLOUR=100;
```

EXAMPLE OF THE TWISS FILE

```
* NAME          S           BETX          BETY
$ %s            %le         %le           %le
 "QF"           1.5425      107.5443191   19.4745051
 "QD"           33.5425     19.5134888    107.4973054
 "QF"           65.5425     107.5443191   19.4745051
 "QD"           97.5425     19.5134888    107.4973054
 "QF"           129.5425    107.5443191   19.4745051
 "QD"           161.5425    19.5134888    107.4973054
 "QF"           193.5425    107.5443191   19.4745051
 "QD"           225.5425    19.5134888    107.4973054
 "QF"           257.5425    107.5443191   19.4745051
 "QD"           289.5425    19.5134888    107.4973054
 "QF"           321.5425    107.5443191   19.4745051
 "QD"           353.5425    19.5134888    107.4973054
 "QF"           385.5425    107.5443191   19.4745051
 "QD"           417.5425    19.5134888    107.4973054
 "QF"           449.5425    107.5443191   19.4745051
 "QD"           481.5425    19.5134888    107.4973054
 "QF"           513.5425    107.5443191   19.4745051
 "QD"           545.5425    19.5134888    107.4973054
 "QF"           577.5425    107.5443191   19.4745051
 "QD"           609.5425    19.5134888    107.4973054
  ....
  ....
```

EXAMPLE OF THE GRAPHICAL OUTPUT (PS FORMAT)



Matching global parameters

It is possible to modify the optical parameters of the machine using the MATCHING module of MAD-X.

- Adjust magnetic strengths to get desired properties (e.g., tune Q, chromaticity dQ),
- Define the properties to match and the **parameters** to vary.

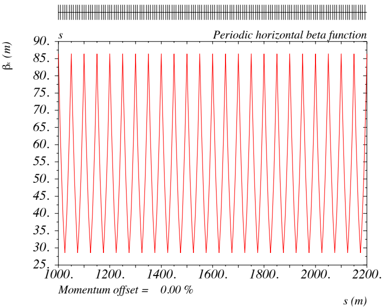
Example:

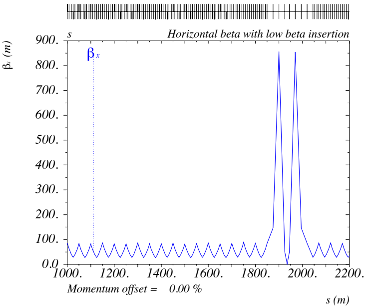
```
MATCH, SEQUENCE=sequence_name;
  GLOBAL, Q1=26.58;//H-tune
  GLOBAL, Q2=26.62;//V-tune
  VARY, NAME= kqf, STEP=0.00001;
  VARY, NAME = kqd, STEP=0.00001;
  LMDIF, CALLS=50, TOLERANCE=1e-6;//method adopted
ENDMATCH;
```

Other types of matching I

Local matching and performance matching:

- Local optical functions (insertions, local optics change),
- any user defined variable.





OTHER TYPES OF MATCHING II

Local matching and performance matching:

- Local optical functions (insertions, local optics change),
- any user defined variable.

Example:

```
MATCH, SEQUENCE=sequence_name;
   CONSTRAINT, range=#e, BETX=50;
   CONSTRAINT, range=#e, ALFX=-2;
    VARY, NAME= kqf, STEP=0.00001;
   VARY, NAME = kqd, STEP=0.00001;
   JACOBIAN, CALLS=50, TOLERANCE=1e-6;
ENDMATCH;
```

General considerations on MAD-X syntax

Input language seems heavy, but:

- can be interfaced to data base and to other programs (e.g., Python),
- programs exist to generate the input interactively,
- allows web based applications,
- allows interface to operating system.

MAD-X can estimate the machine performance by:

- studying of long term stability with multipolar component,
- taking into account the tolerances for machine elements,
- simulating operation of the machine (imperfections,. . .).

Do we use MAD-X for everything? NO!

MAD-X is an optics program (single particle dynamics).

MAD-X has limitations where

- multi particle and multi bunch simulations are required,
- machine is not static, i.e., beam changes its own environment (space charge, instabilities, beam-beam effects. . .),
- requires self-consistent treatment, computation of fields and forces,
- execution speed is an issue,
- for detailed studies dedicated programs are needed, but often with I/O interface to MAD-X.

"Hello World!" input file

LectureExample — vi fodo.mad — 147×49

```
! ## Definition of elements
! Define two quadrupoles (note the deferred assignments).
qf_type: quadrupole, l=1.5, k1:=kf;
qd_type: quadrupole, l=1.5, k1:=kd;

! ## Definition of the sequence
! A short fodo of 10 m.
fodo:sequence, refer=exit, l=10;
qf: qf_type, at=5;
qd: qd_type, at=10;
endsequence;

! ## Definition of the strength
kf=+0.25;
kd:=-kf;

! ## Definition of the beam
beam, particle=proton, energy=7000;

! ## Activation of the sequence
use, sequence=fodo;

! ## Operations
! A simple twiss and plot
select, flag=twiss, column=name,s,betx, bety, alfx,alfy;
twiss, file=before_matching.twiss;
plot, haxis=s, vaxis=betx, bety, colour=100, noversion=true, title='before matching';

! ## Matching
match, sequence=fodo;
  global, q1=.25;
  global, q2=.25;
  vary, name=kf, step=0.00001;
  vary, name=kd, step=0.00001;
  lmdif, calls=50, tolerance=1e-8;
endmatch;

! ## Operations
twiss, file=after_matching.twiss;
plot, haxis=s, vaxis=betx, bety, colour=100, noversion=true, title='after matching', interpolate=true;

! ## Output
value, table(summ,Q1);
value, table(twiss,qf, betx);

! ## Conversion ps2pdf
! This command assumes that in your system the command ps2pdf is available
system, 'ps2pdf madx.ps';
"fodo.mad" 51L, 1234C
```

"Hello World!" output (1)

LectureExample — -bash — 147×49

```
(base) MACBE16107-4:LectureExample sterbini$ madx fodo.mad

  ++++++++++++++++++++++++++++++++++++++++++++
  +     MAD-X 5.02.13  (64 bit, Darwin)      +
  + Support: mad@cern.ch, http://cern.ch/mad +
  + Release   date: 2016.12.20               +
  + Execution date: 2020.01.19 11:29:14      +
  ++++++++++++++++++++++++++++++++++++++++++++
! ## Definition of elements

! Define two quadrupoles (note the deferred assignments).

qf_type: quadrupole, l=1.5, k1:=kf;

qd_type: quadrupole, l=1.5, k1:=kd;


! ## Definition of the sequence

! A short fodo of 10 m.

fodo:sequence, refer=exit, l=10;

qf: qf_type, at=5;

qd: qd_type, at=10;

endsequence;


! ## Definition of the strength

kf=+0.25;

kd:=-kf;


! ## Definition of the beam

beam, particle=proton, energy=7000;


! ## Activation of the sequence

use, sequence=fodo;
```

“Hello World!” output (2)

LectureExample — -bash — 147×49

```
! ## Activation of the sequence

use, sequence=fodo;



! ## Operations

! A simple twiss and plot

select, flag=twiss, column=name,s,betx, bety, alfx,alfy;

twiss, file=before_matching.twiss;

enter Twiss module

iteration:   1 error:   0.000000E+00 deltap:   0.000000E+00
orbit:   0.000000E+00  0.000000E+00  0.000000E+00  0.000000E+00  0.000000E+00  0.000000E+00

++++++ table: summ

            length             orbit5               alfa            gammatr
                10                 -0                  0                  0

                q1                dq1            betxmax              dxmax
      0.3159191546      -0.4863193631        16.65487108                  0

             dxrms             xcomax             xcorms                 q2
                 0                  0                  0       0.3159191546

               dq2            betymax              dymax              dyrms
     -0.4863193631        16.65487108                  0                  0

            ycomax             ycorms             deltap            synch_1
                 0                  0                  0                  0

           synch_2            synch_3            synch_4            synch_5
                 0                  0                  0                  0

            nflips
                 0
plot, haxis=s, vaxis=betx, bety, colour=100, noversion=true, title='before matching';

Plot - default table plotted: twiss

 GXPLOT-X11  1.50 initialized

 plot number =            1
```

"Hello World!" output (3)

LectureExample — -bash — 147×49

```
! ## Matching

match, sequence=fodo;

START MATCHING

number of sequences: 1
sequence name: fodo
  global, q1=.25;

  global, q2=.25;

  vary, name=kf, step=0.00001;

  vary, name=kd, step=0.00001;

  lmdif, calls=50, tolerance=1e-8;

number of variables:    2
user given constraints: 1
total constraints:      2

START LMDIF:

Initial Penalty Function =   0.86906699E+00


call:       4   Penalty function =   0.12041476E-01
call:       7   Penalty function =   0.18270348E-05
call:      10   Penalty function =   0.40829956E-13
 ++++++++++ LMDIF ended: converged successfully
call:      10   Penalty function =   0.40829956E-13
endmatch;


MATCH SUMMARY

Node_Name                  Constraint   Type  Target Value       Final Value        Penalty
--------------------------------------------------------------------------------------------------
Global constraint:         q1           4     2.50000000E-01     2.50000014E-01     1.83678689E-14
Global constraint:         q2           4     2.50000000E-01     2.50000015E-01     2.24620874E-14


Final Penalty Function =   4.08299562e-14
```

“Hello World!” output (4)

LectureExample — -bash — 147×49

```
Variable                 Final Value  Initial Value Lower Limit  Upper Limit
--------------------------------------------------------------------------------
kf                        2.11022e-01  2.50000e-01 -1.00000e+20  1.00000e+20
kd                       -2.11022e-01 -2.50000e-01 -1.00000e+20  1.00000e+20

END MATCH SUMMARY

VARIABLE "TAR" SET TO   4.08299562e-14


! ## Operations

twiss, file=after_matching.twiss;

enter Twiss module

iteration:   1 error:   0.000000E+00 deltap:   0.000000E+00
orbit:   0.000000E+00  0.000000E+00  0.000000E+00  0.000000E+00  0.000000E+00  0.000000E+00

++++++ table: summ

            length             orbit5               alfa            gammatr
                10                 -0                  0                  0

                q1                dq1            betxmax              dxmax
      0.2500000136      -0.3176945739        14.60761389                  0

             dxrms             xcomax             xcorms                 q2
                 0                  0                  0        0.250000015

               dq2            betymax              dymax              dyrms
     -0.3176945752        14.60761386                  0                  0

            ycomax             ycorms             deltap            synch_1
                 0                  0                  0                  0

           synch_2            synch_3            synch_4            synch_5
                 0                  0                  0                  0

            nflips
                 0
plot, haxis=s, vaxis=betx, bety, colour=100, noversion=true, title='after matching', interpolate=true;

Plot - default table plotted: twiss
 plot number =             2
```

"Hello World!" output (5)

LectureExample — -bash — 147×49

```
          dxrms              xcomax              xcorms                  q2
              0                   0                   0         0.250000015

            dq2             betymax               dymax               dyrms
  -0.3176945752         14.60761386                   0                   0

         ycomax              ycorms              deltap             synch_1
              0                   0                   0                   0

        synch_2             synch_3             synch_4             synch_5
              0                   0                   0                   0

         nflips
              0
plot, haxis=s, vaxis=betx, bety, colour=100, noversion=true, title='after matching', interpolate=true;

Plot - default table plotted: twiss
 plot number =            2


! ## Output

value, table(summ,Q1);

table( summ q1 )   =       0.2500000136 ;
value, table(twiss,qf, betx);

table( twiss qf  betx ) =       14.60761389 ;


! ## Conversion ps2pdf

! This command assumes that in your system the command ps2pdf is available

system, 'ps2pdf madx.ps';



! ## Exit

quit;


  Number of warnings: 0

  ++++++++++++++++++++++++++++++++++++++++++++
  +          MAD-X finished normally         +
  ++++++++++++++++++++++++++++++++++++++++++++
(base) MACBE16107-4:LectureExample sterbini$
```