# Course on Optics Design

W. Herr, G. Sterbini, CERN

5 September 2017, RHUL

guido.sterbini@cern.ch

# CAS2017 Practical Course on Optics Design

- GOALS:
  - From the lectures to praxis
  - Design a realistic machine optics with various features
- Not a lecture, but following a series of steps (as exercises) applying what was learned in previous lectures
- Done by you in close collaboration with the tutors and your colleagues (cooperative approach is encouraged)
- The MAD-X program is used for this course

# Procedure and basic steps

- First week: work on 4 exercises
  - Design of periodic machine: starting with the geometry (Ex. 1) and the $\beta$-function properties (Ex. 2)
  - Correction of chromaticity (Ex. 3)
  - Design of a dispersion suppression (Ex. 4)
- Second week: work on group projects
  - 5 different projects for 5 groups on:
    - Low-$\beta$ insertion.
    - Closed orbit correction.
    - Tracking of particles.
    - Transfer line.
    - Beam extraction.
  - Each group will present the project on the 14th Sept afternoon.

# The program

| Time | Sunday 3 Sept. | Monday 4 Sept. | Tuesday 5 Sept. | Wednesday 6 Sept. | Thursday 7 Sept. | Friday 8 Sept. | Saturday 9 Sept. | Sunday 10 Sept. | Monday 11 Sept. | Tuesday 12 Sept. | Wednesday 13 Sept. | Thursday 14 Sept. | Friday 15 Sept. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 08:30–09:20 | ARRIVAL DAY | Opening Talks | Introduction to Lattice Cells<br>B. Holzer | Wakefields and Impedances<br>G. Rumolo | Beam Instabilities – Longitudinal<br>K. Li | Beam Instabilities – Transverse<br>G. Rumolo | Electron Cloud and Instabilities<br>K. Li | EXCURSION | Beam-Beam Effects<br>T. Pieloni | NLD Methods and Tools II<br>W. Herr | NLD Methods and Tools III<br>W. Herr | Low Emittance Machines I<br>A. Wolski | DEPARTURE DAY |
| 09:30–10:20 | | Recap. Transverse Beam Dynamics I<br>H. Schmickler | Recap. Longitudinal Beam Dynamics I | Space Charge in Linear Machines<br>M. Ferrario | Space Charge in Circular Machines<br>M. Ferrario | Instabilities in Linacs<br>M. Ferrario | Feedback Systems II | | Timing and Synchronisation<br>A. Gallo | NLD Phenomenology I<br>Y. Papaphilippou | NLD Phenomenology II<br>Y. Papaphilippou | Insertion Devices<br>J. Clarke | |
| 10:30 | | COFFEE | COFFEE | COFFEE | COFFEE | COFFEE | COFFEE | | COFFEE | COFFEE | COFFEE | COFFEE | |
| 11:00–11:50 | | Introduction to RF Measurement Techniques<br>M. Wendt | Introduction Beam Instrumentation and Diagnostics II<br>R. Jones | Recap Longitudinal Beam Dynamics II | Energy Recovery Linacs<br>A. Jankowiak | Feedback Systems I | Discussion on Instabilities | | NLD Methods and Tools I<br>W. Herr | Study | High Brightness Beam Diagnostics<br>A. Cianchi | Low Emittance Machines II<br>A. Wolski | |
| 12:00–12:50 | | Introduction to Beam Instrumentation and Diagnostics I<br>R. Jones | Introduction to Insertions<br>B. Holzer | Introduction to Non-Linear Dynamics<br>Y. Papaphilippou | Landau Damping I<br>V. Kornilov | Landau Damping II<br>V. Kornilov | Advanced Concepts for Beam-Driven Acceleration<br>M. Ferrario | | Beam Cooling<br>M. Steck | Advanced Magnet Technologies<br>L. Quettier | Discussion on Non-Linear Dynamics | Advanced Concepts for Laser-Driven Acceleration<br>S. Hooker | |
| 13:00 | | LUNCH | LUNCH | LUNCH | LUNCH | LUNCH | LUNCH | | LUNCH | LUNCH | LUNCH | LUNCH | |
| 14:30–15:20 | | Recap Transverse Beam Dynamics II<br>H. Schmickler | C1 C2 C3 | C1 C2 C3 | FREE | C1 C2 C3 | C1 C2 C3 | | C1 C2 C3 | C1 C2 C3 | FREE | C1 C2 C3 Presentations | |
| 15:30–16:20 | | Introduction to Optics Design<br>B. Holzer | | | | | | | | | | | |
| 16:30 | | COFFEE | COFFEE | COFFEE | | COFFEE | COFFEE | | COFFEE | COFFEE | | COFFEE | |
| 17:00 | Registration | 1S1M | C1 C2 C3 | C1 C2 C3 | | C1 C2 C3 | C1 C2 C3 | | C1 C2 C3 | C1 C2 C3 | | Closing Talks | |
| 19:30 | Dinner | Dinner | Dinner | Dinner | Dinner | Dinner | Dinner | Dinner | Dinner | Dinner | Dinner | Dinner | |

C1 – Beam Instrumentation and Diagnostics; C2 – RF Measurement Techniques; C3 – Optics Design and Correction

# Available tools

- Individual computer with unix-like operating system. You can access the them with via Window7 machines (see Stewart's talk):
- You can also try to use your laptop if you prefer.
  - download MAD-X from www.cern.ch/madx
- We will use the CAS2017 INDICO site to post all the needed material and, as we progress, the solutions.
- You can access the material also on the following folder available on the UNIX machines /afs/cern.ch/work/s/sterbini/public/RHULCAS

# Do not hesitate to ask and share your doubts

WE ARE HERE TO HELP!

- Stewart BOOGERT
- Werner HERR
- Bernhard HOLZER (1st week)
- Kevin LI (1st week)
- Yannis PAPAPHILIPPOU (2nd week)
- Guido STERBINI

# INTRODUCTION TO MAD-X

MAD-X: Methodical Accelerator Design version 10.

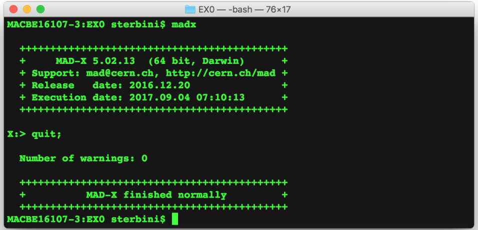
DISCLAIMER. This material is intended to be an introduction to MAD-X: a large part of the code capabilities are not discussed in details or are not discussed at all!

We will use MAD-X to "visualise" the transverse dynamics concepts.

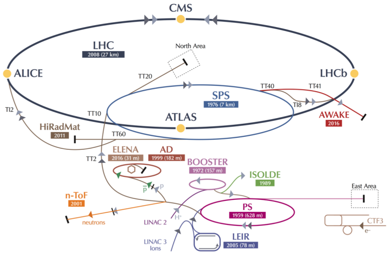
Use the MADX-Primer you received as reference for the MAD-X syntax. In this slides we summarize it to have the basic knowledge to start the hands-on session.

# WHAT IS MAD-X?

- A general purpose beam optics and lattice program distributed for free by CERN.
- It is used at CERN since <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 easy to extend,
- made for complicated applications, powerful and rather complete,
- mainly designed for large projects (LEP, LHC, CLIC...).

MAD-X is NOT

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

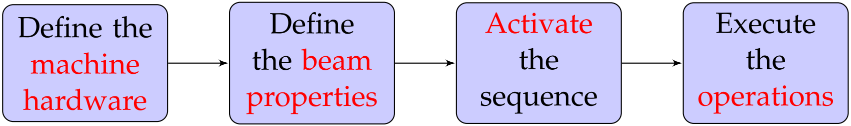
# IN LARGE PROJECTS (E.G., LHC):

- 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, $e$, $\pi$, $m_p$, $m_e$. . . ).

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

EX0 — madx — 67×25

```
MACBE16107-3:EX0 sterbini$ madx

  ++++++++++++++++++++++++++++++++++++++++++++
  +     MAD-X 5.02.13  (64 bit, Darwin)      +
  + Support: mad@cern.ch, http://cern.ch/mad +
  + Release   date: 2016.12.20               +
  + Execution date: 2017.09.04 07:14:23      +
  ++++++++++++++++++++++++++++++++++++++++++++

X:> a=1;
X:> b=a;
X:> c:=a;
X:> a=2;
++++++ info: a redefined
X:> value a;
a                  =                  2 ;
X:> value b;
b                  =                  1 ;
X:> value c;
c                  =                  2 ;
X:>
```

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

# MAD-X CONVENTION

- Not case sensitive
- Elements placed along the reference orbit (variable s)
- Horizontal (assumed bending plane) and vertical variables are x and y
- Describes a local coordinate system moving along s i.e. $x = y = 0$ follows the curvilinear system (reference orbit)

# USING FILE AS INPUT

For a large machine you may need many commands (LHC? >27000)
Better: store your input in different files: e.g. myinput.madx

```
$ madx
X:> call, file= myinput.madx;
```

alternatively, redirection from the file into the parser (LINUX)

```
$ madx< myinput.madx
```

WARNING: the input file has to be plain text file (ASCII)!

# USING FILE AS INPUT

sterbini — -bash — 79×34

```
macbe16107:~ sterbini$ echo "stop;">myinput.madx
macbe16107:~ sterbini$ madx

  ++++++++++++++++++++++++++++++++++++++++++++
  +     MAD-X 5.02.13  (64 bit, Darwin)      +
  + Support: mad@cern.ch, http://cern.ch/mad +
  + Release   date: 2016.12.20               +
  + Execution date: 2017.08.24 10:51:13      +
  ++++++++++++++++++++++++++++++++++++++++++++

X:> call, file="myinput.madx";

  Number of warnings: 0

  ++++++++++++++++++++++++++++++++++++++++++++
  +          MAD-X finished normally         +
  ++++++++++++++++++++++++++++++++++++++++++++
macbe16107:~ sterbini$ madx<myinput.madx

  ++++++++++++++++++++++++++++++++++++++++++++
  +     MAD-X 5.02.13  (64 bit, Darwin)      +
  + Support: mad@cern.ch, http://cern.ch/mad +
  + Release   date: 2016.12.20               +
  + Execution date: 2017.08.24 10:51:43      +
  ++++++++++++++++++++++++++++++++++++++++++++
stop;


  Number of warnings: 0

  ++++++++++++++++++++++++++++++++++++++++++++
  +          MAD-X finished normally         +
  ++++++++++++++++++++++++++++++++++++++++++++
macbe16107:~ sterbini$
```

# MAD INPUT STATEMENTS

- Typical assignments:
  - Properties of machine elements
  - Set up of the lattice
  - Definition of beam properties (particle type, energy, emittance ...)
- Typical actions:
  - Compute lattice functions, match optical parameters
  - Assignment of errors and imperfections
  - Correct machines

Recommendation: make use of the examples!

# Definitions of the lattice elements

Generic pattern to define an element:

*label*: *keyword*, attributes...;

- 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 $k_2$:

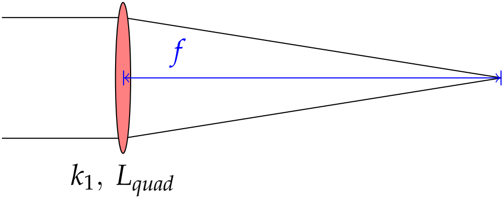
$$k_2 = \frac{1}{B\rho} \frac{\partial^2 B_y}{\partial x^2} \left[\text{in m}^{-3}\right]$$

# Interlude I

What does $k_1$ mean? It is related to the quad focal length [1].

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

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




[1] thin lens approximation

# EXAMPLE: DEFINITIONS OF ELEMENTS

- Sextupole magnet:

  ksf = 0.00156;

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

- Multipole magnet "thin" element [2]:

  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;

[2] We are going to use extensively it during the course

# 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?)

sterbini — sterbini@lxplus030:/afs/cern.ch/eng/lhc/optics/runII/2017 — ssh lxplus.cern.ch — 128×34

```
MCBWV : VCORRECTOR, L := l.MCBWV, Kmax := Kmax_MCBWV, Kmin := Kmin_MCBWV, Calib := Kmax_MCBWV / Imax_MCBWV;
MCBXV : VCORRECTOR, Lrad := l.MCBXV, Kmax := Kmax_MCBXV, Kmin := Kmin_MCBXV, Calib := Kmax_MCBXV / Imax_MCBXV;
MCBYV : VCORRECTOR, L := l.MCBYV, Calib := Kmax_MCBYV_4.5K / Imax_MCBYV_4.5K;
//---------------------- VKICKER        --------------------------------------------
MBAW : VKICKER, L := l.MBAW, Kmax := Kmax_MBAW, Kmin := Kmin_MBAW, Calib := Kmax_MBAW / Imax_MBAW;
MBWMD : VKICKER, L := l.MBWMD, Kmax := Kmax_MBWMD, Kmin := Kmin_MBWMD, Calib := Kmax_MBWMD / Imax_MBWMD;
MBXWT : VKICKER, L := l.MBXWT, Kmax := Kmax_MBXWT, Kmin := Kmin_MBXWT, Calib := Kmax_MBXWT / Imax_MBXWT;

/*********************************************************************************/
/*                        LHC SEQUENCE                                           */
/*********************************************************************************/

LHCB1 : SEQUENCE, refer = CENTRE, L = LHCLENGTH;
IP1:OMK,                           at= pIP1+IP1OFS.B1*DS;
  MBAS2.1R1:MBAS2,                 at= 1.5+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 2209454,
  TAS.1R1:TAS,                     at= 19.95+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 102103,
  BPMSW.1R1.B1:BPMSW002,           at= 21.564+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 6080259, assembly_id= 6080224,
  BPMSW.1R1.B1_DOROS:BPMSW002,     at= 21.564+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 10429420, assembly_id= 6080224,
  BPMWK.1R1:BPMWK,                 at= 21.62+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 6080224,
  BPMWF.A1R1.B1:BPMWF,             at= 21.724+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 6080267, assembly_id= 6080224,
  MQXA.1R1:MQXA,                   at= 26.15+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 282126, assembly_id= 102104,
  MCBXH.1R1:MCBXH,                 at= 29.842+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 282213, assembly_id= 102104,
  MCBXV.1R1:MCBXV,                 at= 29.842+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 282212, assembly_id= 102104,
  BPMS.2R1.B1:BPMS,                at= 31.529+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 241889, assembly_id= 102105,
  MQXB.A2R1:MQXB,                  at= 34.8+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 241890, assembly_id= 102105,
  MCBXH.2R1:MCBXH,                 at= 38.019+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 249450, assembly_id= 102105,
  MCBXV.2R1:MCBXV,                 at= 38.019+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 249451, assembly_id= 102105,
  MQXB.B2R1:MQXB,                  at= 41.3+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 241892, assembly_id= 102105,
  TASB.3R1:TASB,                   at= 45.342+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 241893, assembly_id= 102106,
  MQSX.3R1:MQSX,                   at= 46.608+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 282127, assembly_id= 102106,
  MQXA.3R1:MQXA,                   at= 50.15+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 241895, assembly_id= 102106,
  MCBXH.3R1:MCBXH,                 at= 53.814+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 249456, assembly_id= 102106,
  MCBXV.3R1:MCBXV,                 at= 53.814+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 249457, assembly_id= 102106,
                                                                                    772,0-1        2%
```

# The WHILE instruction

Very useful for periodic sequences!

```
...
qf: multipole,knl=0,kqf*lq;
qd: multipole,knl=0,kqd*lq;
ncells=10;
...
cas1: sequence, refer=centre, l=circum;
...
n=1;
WHILE (n<ncells) {
        qf: qf, at=(n-1)*lcell;
        qd: qd, at=(n-1)*lcell+0.50*lcell;
        n = n + 1;
}
...
```

# 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., $\beta$-functions):
  PLOT, HAXIS=s, VAXIS=betx,bety;

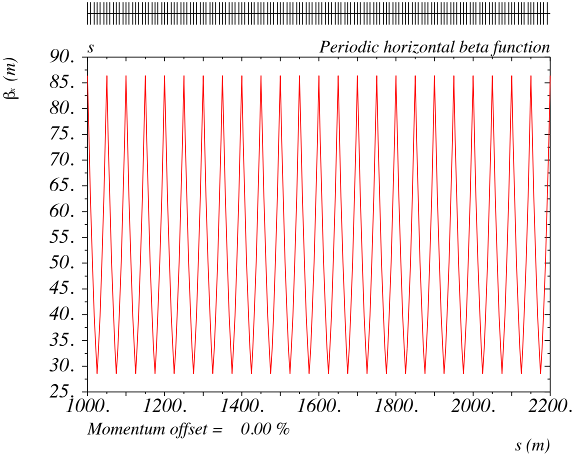
Example

```
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.

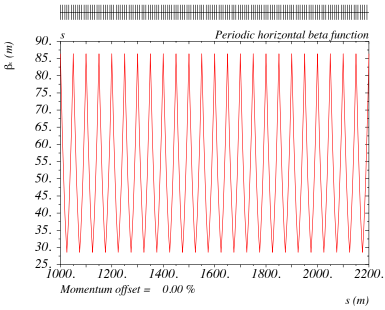
Example:

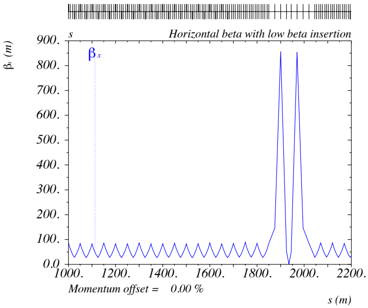
```
MATCH, SEQUENCE=sequence_name;
  GLOBAL, Q1=26.58;//H-tune
  GLOBAL, Q2=26.62;//V-tune
  VARY, NAME= kqf, STEP=0.00001;
  VARY, NAME = kqd, STEP=0.00001;
  LMDIF, CALLS=50, TOLERANCE=1e-6;//method adopted
ENDMATCH;
```

# 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, Matlab™...),
- 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.

# Suggested approach

- THINK about the physics behind the exercise and refer to the lectures.
- SKETCH your machine and its element.
- EVALUATE on paper what you expect.
- SIMULATE your machine in MAD-X.
- INTERPRET and EXPLORE the results.

# "Hello World!" input file

EX0 — vi helloWorld.madx — 94×40

```
/****Definition of elements****/
qfType:QUADRUPOLE, L=1.5, K1:=kf;
qdType:QUADRUPOLE, L=1.5, K1:=kd;

/****Definition of the sequence****/
fodo:SEQUENCE, REFER=exit, L=10;
qf: qfType, at=5;
qd: qdType, at=10;
ENDSEQUENCE;

/****Definition of the strength****/
kf=+0.2985;
kd:=-kf;

/****Definition of the beam****/
beam;

/****Activation of the sequence****/
use, sequence=fodo;

/****Operations****/
twiss, file=beforeMatching.twiss;
plot, HAXIS=s, VAXIS=betx, bety, title='Before matching', colour=100, interpolate=true;

/****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;

twiss, file=afterMatching.twiss;

plot, HAXIS=s, VAXIS=betx, bety, title='after matching', colour=100,  interpolate=true;

/****Best Regards****/
QUIT;
```

# “Hello World!” output (1)

EX0 — -bash — 93×30

```
twiss, file=beforeMatching.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
      0.4877944671        -8.26503645        208.1244543                  0

             dxrms             xcomax             xcorms                 q2
                 0                  0                  0       0.4877944671

               dq2            betymax              dymax              dyrms
       -8.26503645        208.1244543                  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
```

# "Hello World!" output (2)

EX0 — -bash — 106×35

```
START LMDIF:

Initial Penalty Function =   0.11309242E+02


call:       4  Penalty function =   0.59659299E+01
call:       7  Penalty function =   0.27181868E+01
call:      10  Penalty function =   0.39842148E+00
call:      13  Penalty function =   0.23236533E-02
call:      16  Penalty function =   0.66509381E-07
call:      19  Penalty function =   0.57925759E-16
 ++++++++++ LMDIF ended: converged successfully
call:      19  Penalty function =   0.57925759E-16
ENDMATCH;


MATCH SUMMARY

Node_Name                  Constraint   Type  Target Value       Final Value        Penalty
--------------------------------------------------------------------------------------------------
Global constraint:         q1           4     2.50000000E-01     2.50000000E-01     1.46427285E-17
Global constraint:         q2           4     2.50000000E-01     2.50000001E-01     4.32830308E-17


Final Penalty Function =   5.79257593e-17





Variable                 Final Value  Initial Value Lower Limit  Upper Limit
--------------------------------------------------------------------------------
kf                        2.11022e-01  2.98500e-01 -1.00000e+20  1.00000e+20
kd                       -2.11022e-01 -2.98500e-01 -1.00000e+20  1.00000e+20
```

# "Hello World!" output (3)

EX0 — -bash — 106×35

```
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
      0.2500000004      -0.3176945857        14.60761372                  0

             dxrms             xcomax             xcorms                 q2
                 0                  0                  0       0.2500000007

               dq2            betymax              dymax              dyrms
     -0.3176945859        14.60761371                  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, title='after matching', colour=100,  interpolate=true;

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