Introduction to MAD-X

G. Sterbini, CERN
Thanks to W. Herr and B. Holzer

14 January 2014, Archamps

guido.sterbini@cern.ch

THE MAD-X LECTURES

We will have

- 1 h lecture (now).
- 5 h "hand-on" tutorials (today, tomorrow and on Thursday).
  - Today's tutorial (1 h) will be dedicated to check the pc remote connections and to prepare a very simple input file for MADX.
  - Tomorrow's tutorial will be dedicated to explore a FODO cell (1 h) and a FODO lattice (1 h).
  - On Thursday we will study a matching cell (1 h) and we will play with the LHC MADX description (1 h).

Each tutorial is split in two parts of $\approx$ 30 min (last 10 minutes for Q&A). Basic knowledge of Windows and Unix is assumed but do not hesitate to ask in case: we are here to help.

MAD-X IN 60M:00S!

Introduction

MAD-X syntax

Daily life

"Hello World!" example

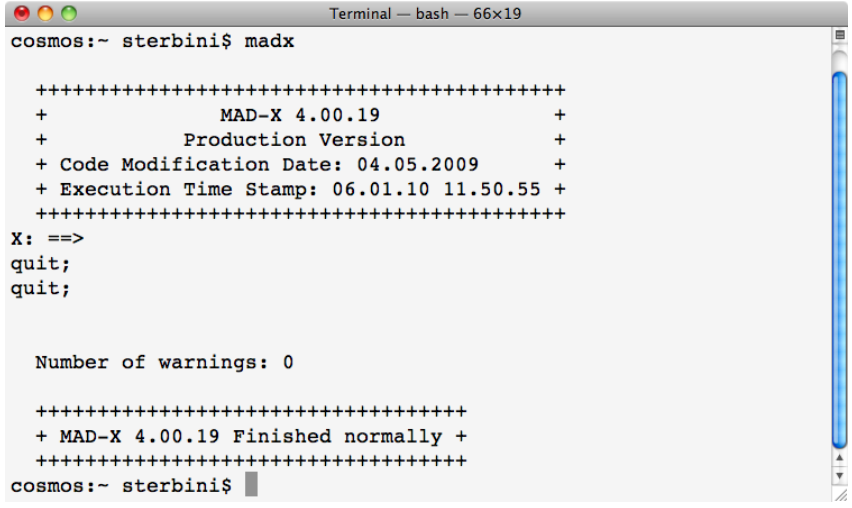
DISCLAIMER. This material is intended to be an introduction to MAD-X: a large part of the code capabilities are not discussed in details or are not discussed at all! We will use MAD-X to "visualise" the transverse dynamics concepts.

If you want to deepen the subject you can find a lot of material on the web (`http://mad.web.cern.ch/mad/madx.old/madx_manual.pdf`)...

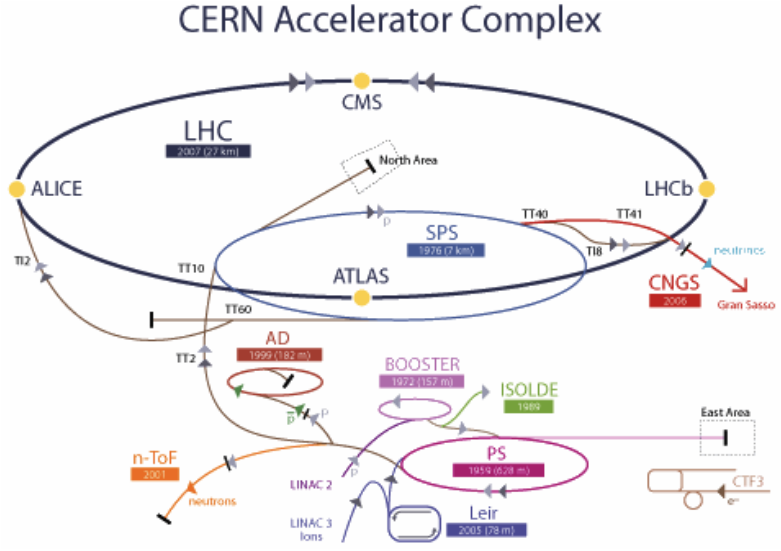
- googling "madx", you get the MAD-X homepage.
- To wet your appetite, you can google "MAD-X primer".
- To go in details, you can google "MAD-X manual".

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 20 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

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

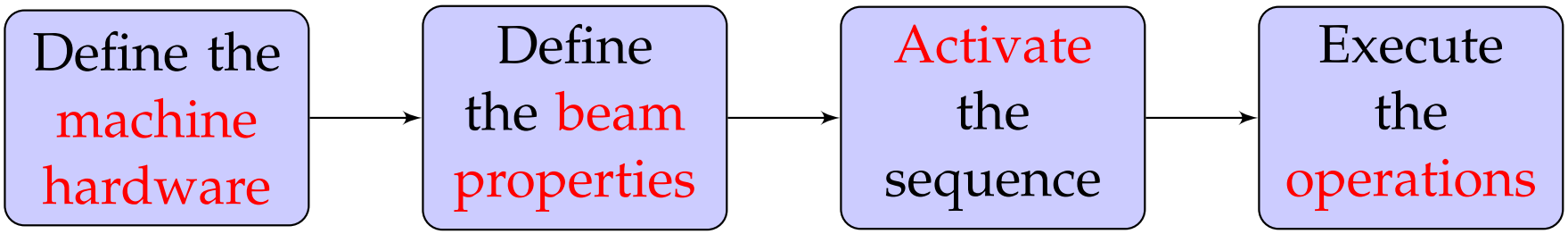
In large projects (e.g., LHC):

- 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...

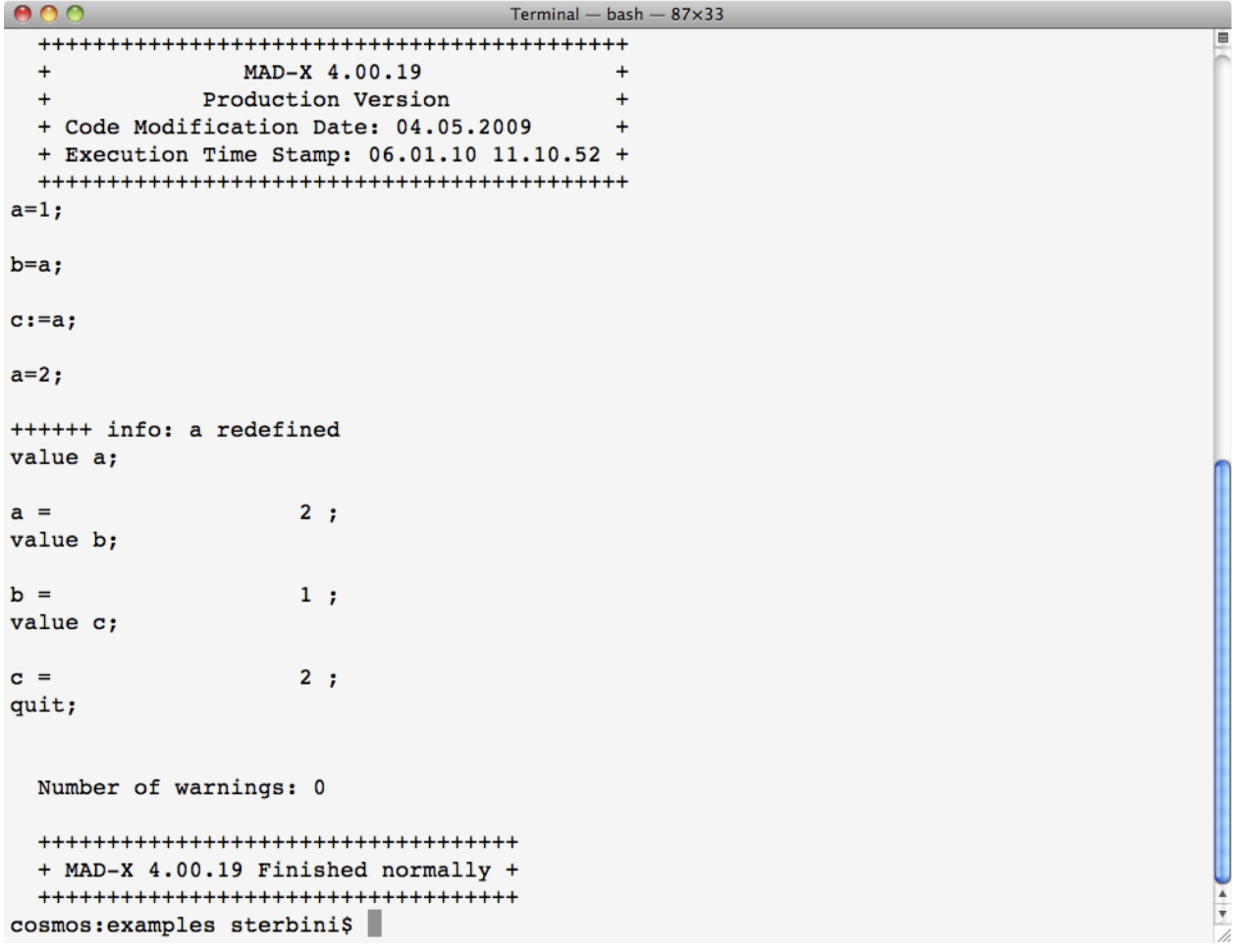
MAD-X INPUT LANGUAGE

- 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:

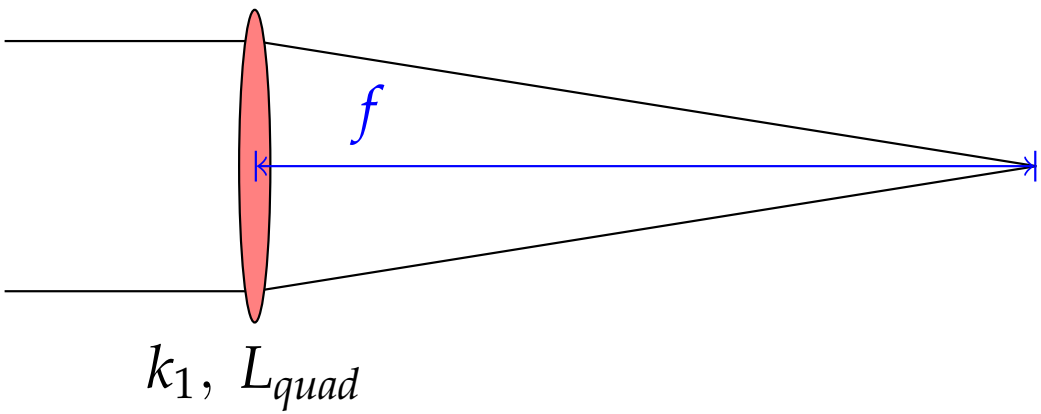
$$k_2 = \frac{1}{B\rho} \frac{\partial^2 B_y}{\partial x^2} \left[\text{in m}^{-3}\right]$$

INTERLUDE

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

$$\frac{1}{k_1 \; L_{quad}} = f \tag{1}$$

Assuming $k_1 = 10^{-1}$ m 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:

  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?)

Terminal — vim — 114×36

```
640 MSS : SEXTUPOLE, L := l.MSS, Kmax := Kmax_MSS, Kmin := Kmin_MSS, Calib := Kmax_MSS / Imax_MSS;
641 //---------------------- SOLENOID       -------------------------------------------
642 MBAS2 : SOLENOID, L := l.MBAS2;
643 MBCS2 : SOLENOID, L := l.MBCS2;
644 MBLS2 : SOLENOID, L := l.MBLS2;
645 //---------------------- VCORRECTOR     -------------------------------------------
646 MCBCV : VCORRECTOR, L := l.MCBCV, Kmax := Kmax_MCBCV, Kmin := Kmin_MCBCV, Calib := Kmax_MCBCV / Imax_MCBCV;
647 MCBV : VCORRECTOR, L := l.MCBV, Kmax := Kmax_MCBV, Kmin := Kmin_MCBV, Calib := Kmax_MCBV / Imax_MCBV;
648 MCBWV : VCORRECTOR, L := l.MCBWV, Kmax := Kmax_MCBWV, Kmin := Kmin_MCBWV, Calib := Kmax_MCBWV / Imax_MCBWV;
649 MCBXV : VCORRECTOR, L := l.MCBXV, Kmax := Kmax_MCBXV, Kmin := Kmin_MCBXV, Calib := Kmax_MCBXV / Imax_MCBXV;
650 MCBYV : VCORRECTOR, L := l.MCBYV, Kmax := Kmax_MCBYV, Kmin := Kmin_MCBYV, Calib := Kmax_MCBYV / Imax_MCBYV;
651 //---------------------- VKICKER        -------------------------------------------
652 MBAW : VKICKER, L := l.MBAW, Kmax := Kmax_MBAW, Kmin := Kmin_MBAW, Calib := Kmax_MBAW / Imax_MBAW;
653 MBWMD : VKICKER, L := l.MBWMD, Kmax := Kmax_MBWMD, Kmin := Kmin_MBWMD, Calib := Kmax_MBWMD / Imax_MBWMD;
654 MBXWT : VKICKER, L := l.MBXWT, Kmax := Kmax_MBXWT, Kmin := Kmin_MBXWT, Calib := Kmax_MBXWT / Imax_MBXWT;
655
656 //------------------------ LHC SEQUENCE                  -------------------------
657 LHCB1 : SEQUENCE, refer = CENTRE, L = LHCLENGTH;
658 IP1:OMK,                  at= pIP1+IP1OFS.B1*DS;
659   MBAS2.1R1:MBAS2,        at= 1.5+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 2209454,
660   TAS.1R1:TAS,            at= 20.015+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 102103,
661   BPMSW.1R1.B1:BPMSW,     at= 21.475+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 104594,
662   MQXA.1R1:MQXA,          at= 26.15+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 282126, assembly_id= 102104,
663   MCBXH.1R1:MCBXH,        at= 29.842+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 282213, assembly_id= 102104,
664   MCBXV.1R1:MCBXV,        at= 29.842+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 282212, assembly_id= 102104,
665   BPMS.2R1.B1:BPMS,       at= 31.529+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 241889, assembly_id= 102105,
666   MQXB.A2R1:MQXB,         at= 34.8+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 241890, assembly_id= 102105,
667   MCBXH.2R1:MCBXH,        at= 38.019+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 249450, assembly_id= 102105,
668   MCBXV.2R1:MCBXV,        at= 38.019+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 249451, assembly_id= 102105,
669   MQXB.B2R1:MQXB,         at= 41.3+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 241892, assembly_id= 102105,
670   TASB.3R1:TASB,          at= 45.342+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 241893, assembly_id= 102106,
671   MQSX.3R1:MQSX,          at= 46.608+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 282127, assembly_id= 102106,
672   MQXA.3R1:MQXA,          at= 50.15+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 241895, assembly_id= 102106,
673   MCBXH.3R1:MCBXH,        at= 53.814+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 249456, assembly_id= 102106,
674   MCBXV.3R1:MCBXV,        at= 53.814+(0-IP1OFS.B1)*DS, mech_sep= 0, slot_id= 249457, assembly_id= 102106,
```

BEAM DEFINITION & SEQUENCE ACTIVATION

Generic pattern to define the beam:

label: BEAM, PARTICLE=x, ENERGY[2]=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.

[2] 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;

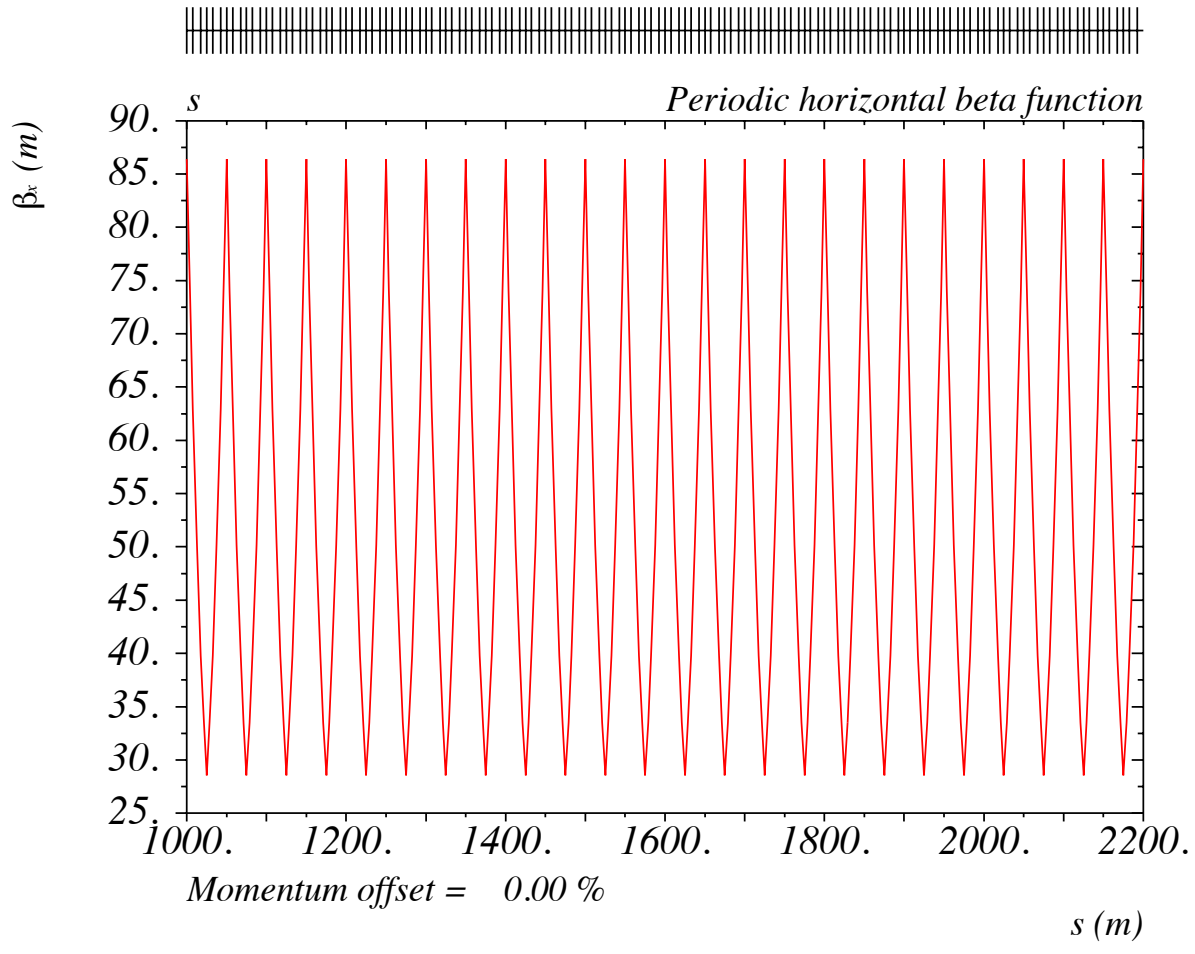
Example

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.

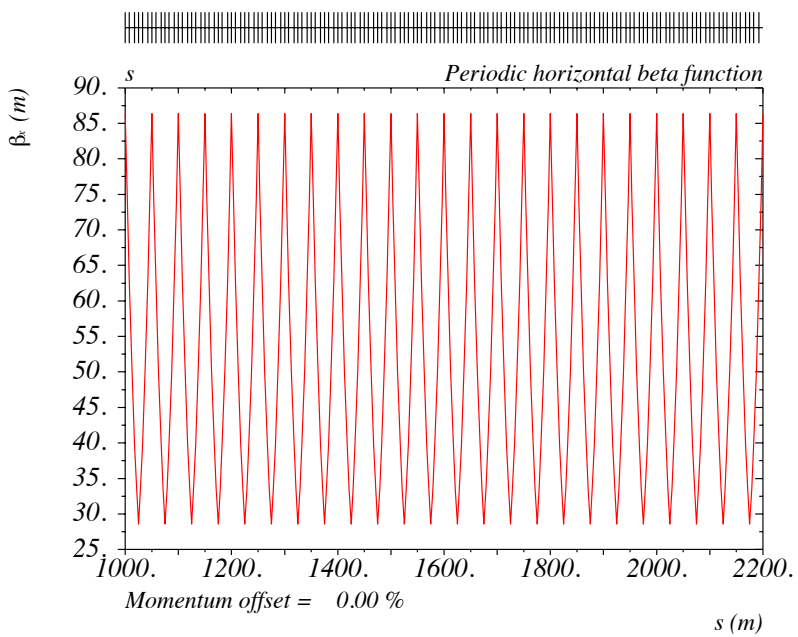
Example:

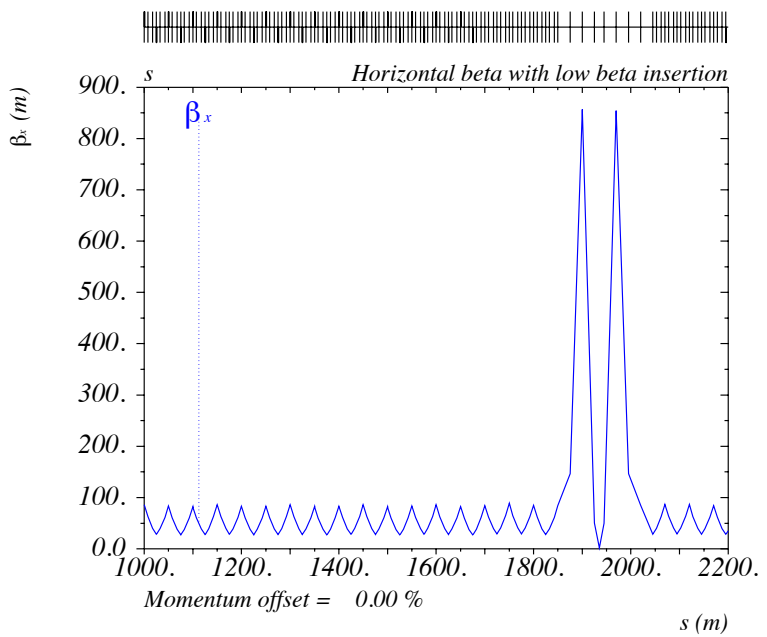
```
MATCH, SEQUENCE=sequence_name;
   GLOBAL, Q1=26.58;//H-tune
   GLOBAL, Q2=26.62;//V-tune
   VARY, NAME= kqf, STEP=0.00001;
   VARY, NAME = kqd, STEP=0.00001;
   LMDIF, CALLS=50, TOLERANCE=1e-6;//method adopted
ENDMATCH;
```

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., Mathematica™, 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.

SOME USEFUL TIPS FOR THE TUTORIALS (WIN)

Additional software needed...

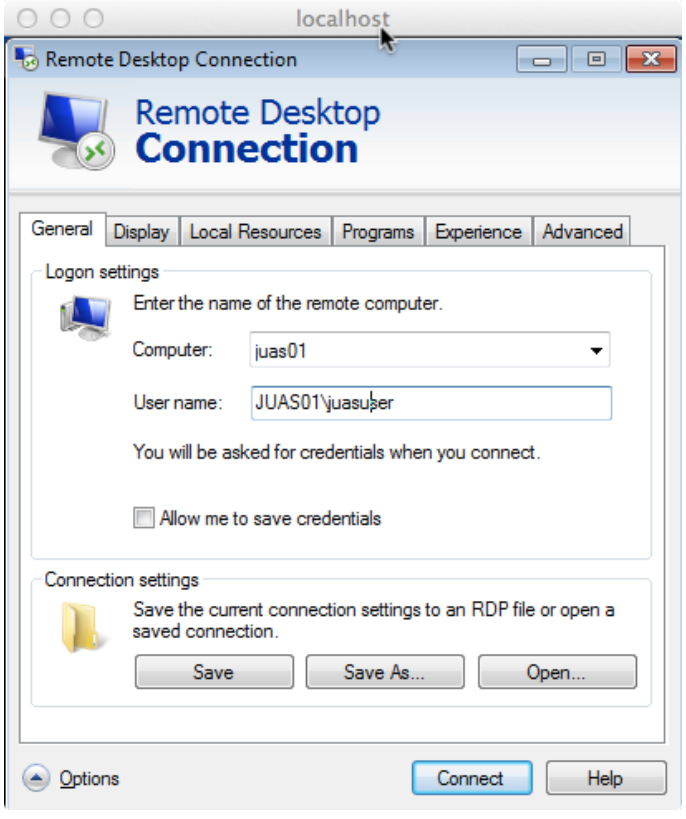
- Most of the input/output of MADX is in ASCII files: you can read/write/edit them using Notepad++. The graphical output of MAD is on PS format: you can use Evince to open it.

Tricks for beginners:

- a working MAD-X environment has been set up for you
- to make all happy, we will use a unix-lixe shell (Cygwin) in a WIN OS.
- once you have the input (e.g., whatever.inp), execute it by:
  - 1. "CALL, FILE=whatever.inp;" at the MADX prompt,
  - 2. madx<whatever.inp" at the shell (preferred).

ACCESS REMOTE MACHINE JUASXX

- Computer: JUASXX
- User: JUASXX\ juasuser
- PWD: Juas2012User
- Cygwin (UNIX shell in WIN). From the shell with the command "open" you open the Win Explorer (edit inp, read out)
- Use your assigned PC!



"Hello World!" input file

Terminal — vim — 105×36

```
 1 /****Definition of elements****/
 2 qfType:QUADRUPOLE, L=1.5, K1:=kf;
 3 qdType:QUADRUPOLE, L=1.5, K1:=kd;
 4
 5 /****Definition of the sequence****/
 6 fodo:SEQUENCE, REFER=exit, L=10;
 7 qf: qfType, at=5;
 8 qd: qdType, at=10;
 9 ENDSEQUENCE;
10
11 /****Definition of the strength****/
12 kf=+0.2985;
13 kd=-0.2985;
14
15 /****Definition of the beam****/
16 beam, particle=proton, energy=7001;
17
18 /****Activation of the sequence****/
19 use, sequence=fodo;
20
21 /****Operations****/
22 twiss;
23 plot, HAXIS=s, VAXIS=betx, bety;
24
25 /****Matching****/
26 MATCH, sequence=fodo;
27   GLOBAL, Q1=.25;
28   GLOBAL, Q2=.25;
29   VARY, NAME=kf, STEP=0.00001;
30   VARY, NAME=kd, STEP=0.00001;
31   LMDIF, CALLS=50, TOLERANCE=1e-8;
32 ENDMATCH;
33
34 /****Best Regards****/
35 QUIT;
"fodo.mad" 35L, 689C
```

"HELLO WORLD!" OUTPUT (1)

Terminal — bash — 105×36

```
cosmos:examples sterbini$ madx<fodo.mad

  ++++++++++++++++++++++++++++++++++++++++++++
  +              MAD-X 4.00.19               +
  +            Production Version            +
  + Code Modification Date: 04.05.2009       +
  + Execution Time Stamp: 07.01.10 12.04.00 +
  ++++++++++++++++++++++++++++++++++++++++++++
/****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=-0.2985;
```

"HELLO WORLD!" OUTPUT (2)

Terminal — bash — 105×36

```
/****Definition of the beam****/

beam, particle=proton, energy=7001;



/****Activation of the sequence****/

use, sequence=fodo;



/****Operations****/

twiss;

enter Twiss module
++++++ info: Zero value of SIGT replaced by 1.
++++++ info: Zero value of SIGE replaced by 1/1000.

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   -3.30872245e-24   -5.497558139e+11

                q1                dq1            betxmax               dxmax
      0.4877944671       -8.265035446        208.1244543                   0

             dxrms             xcomax             xcorms                  q2
                 0                  0                  0       0.4877944671

               dq2            betymax              dymax              dyrms
      -8.265035446        208.1244543                  0                  0
```

"HELLO WORLD!" OUTPUT (3)

Terminal — bash — 105×36

```
twiss;

enter Twiss module
++++++ info: Zero value of SIGT replaced by 1.
++++++ info: Zero value of SIGE replaced by 1/1000.

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   -3.30872245e-24   -5.497558139e+11

                q1                dq1            betxmax               dxmax
      0.4877944671       -8.265035446        208.1244543                   0

             dxrms             xcomax             xcorms                  q2
                 0                  0                  0       0.4877944671

               dq2            betymax              dymax              dyrms
      -8.265035446        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
plot, HAXIS=s, VAXIS=betx, bety;

++++++ info: Zero value of SIGT replaced by 1.
++++++ info: Zero value of SIGE replaced by 1/1000.

 GXPLOT-X11  1.50 initialized

 plot number =            1
```

"HELLO WORLD!" OUTPUT (4)

Terminal — bash — 105×36

```
/****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=100, TOLERANCE=1e-7;

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

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
 ++++++++++ LMDIF ended: converged successfully
call:      16   Penalty function =   0.66509381E-07
ENDMATCH;
```

"HELLO WORLD!" OUTPUT (5)

Terminal — bash — 105×36

```
MATCH SUMMARY

Node_Name                    Constraint   Type  Target Value       Final Value        Penalty
-------------------------------------------------------------------------------------------
Global constraint:           q1           4     2.50000000E-01     2.50018141E-01     3.29107276E-08
Global constraint:           q2           4     2.50000000E-01     2.50018330E-01     3.35986532E-08


Final Penalty Function =   6.65093808e-08




Variable                  Final Value  Initial Value Lower Limit  Upper Limit
-------------------------------------------------------------------------------
kf                         2.11034e-01  2.98500e-01 -1.00000e+20  1.00000e+20
kd                        -2.11034e-01 -2.98500e-01 -1.00000e+20  1.00000e+20

END MATCH SUMMARY

VARIABLE "TAR" SET TO   6.65093808e-08


/****Best Regards****/

QUIT;


  Number of warnings: 0

  ++++++++++++++++++++++++++++++++++
  + MAD-X 4.00.19 Finished normally +
  ++++++++++++++++++++++++++++++++++
```

TUTORIAL 1: FIRST PART

My first MADX job.

1. Connect to the remote machine assigned to your pc station (JuasXX).
2. Open a Cygwin terminal and make and move to a new folder.
3. Open an editor and write your first MADX input file (just 1 line like “stop;” or “exit;” or “quit;”).
4. Run it. If all is fine, nothing interesting should happen.

TUTORIAL 1: SECOND PART

My first accelerator.

1. Make a very simple machine with 2 quads (focusing and defocusing). Each quad is L=0.1 m long and has a focal length of f=20 m (K1$\times$L=1/f in thin lens approximation).
2. Build a sequence of 4 m putting the center of the quads at 1 and 3 m.
3. Define a proton beam at $E_{tot} = 2$ GeV. Activate the sequence, try to find the periodic solution and plot the β-functions. If you found $\beta_{max} \approx 43$ m you succeeded. Try with $E_{tot} = 0.7$ GeV.
4. Using the plot you obtained can you estimate the phase advance of the cell?

Tutorial 2: First Part

Build a simplified LHC FODO cell.

- The LHC FODO cell is long 106.9 m. Each quad is ≈ 5.3 m. Build the cell in MADX putting the start of the first quad at the start of the sequence.
- Define the beam (proton at $E_{tot} = 7$ TeV), activate the sequence and try to twiss it powering the quads to obtain $\Delta\mu \approx 90$ deg phase advance in the cell using the thin lens approximation (use Fig. 1). What is the actual phase advance computed by MADX?

TUTORIAL 2: FIRST PART

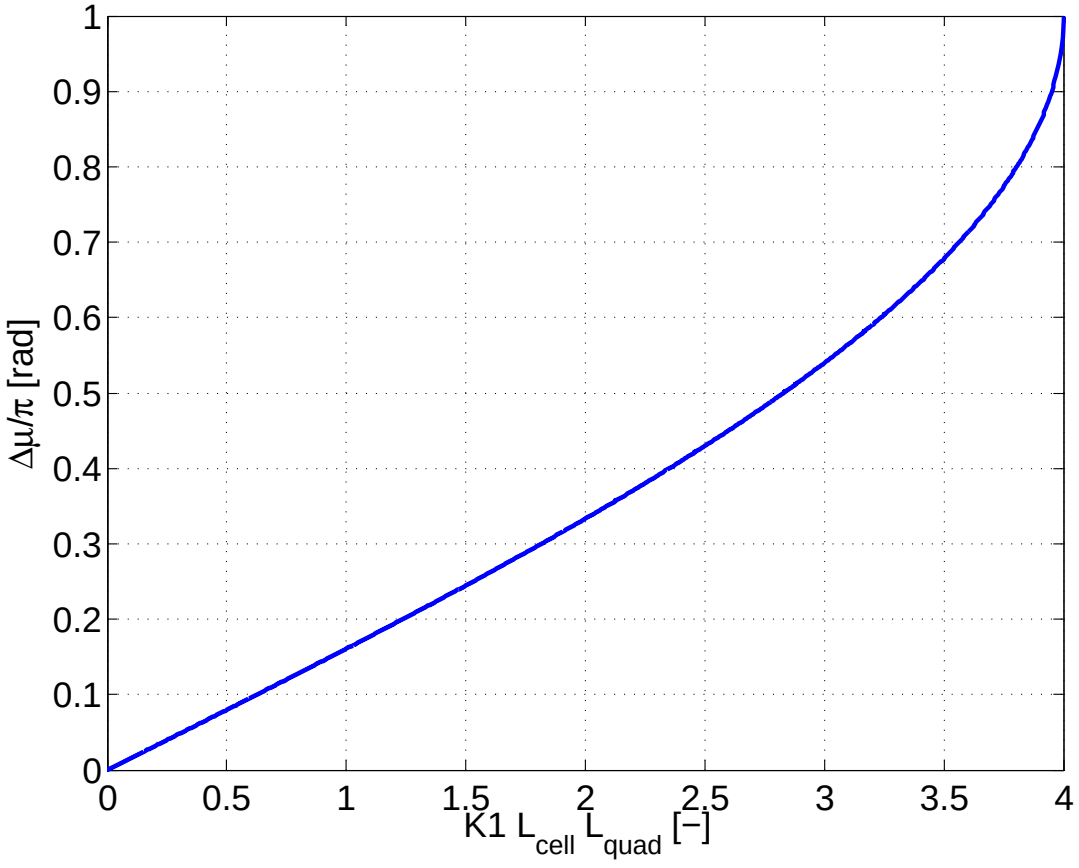


Figure 1: Phase advance versus quad strength, cell length and quad length. Thin lens approximation of a FODO.

Tutorial 2: Second Part

Build a simplified LHC FODO cell.

- What is the β_{max}? Compare with the thin lens approximation (Fig. 2). And the maximum beam σ (assume ϵ_n=3 mrad mm, $E_{tot} = 7$ TeV).
- If you increase the focusing strength of the quadrupole, what is the effect of it on the β_{max}, β_{min} and on the $\Delta\mu$?

TUTORIAL 2: SECOND PART

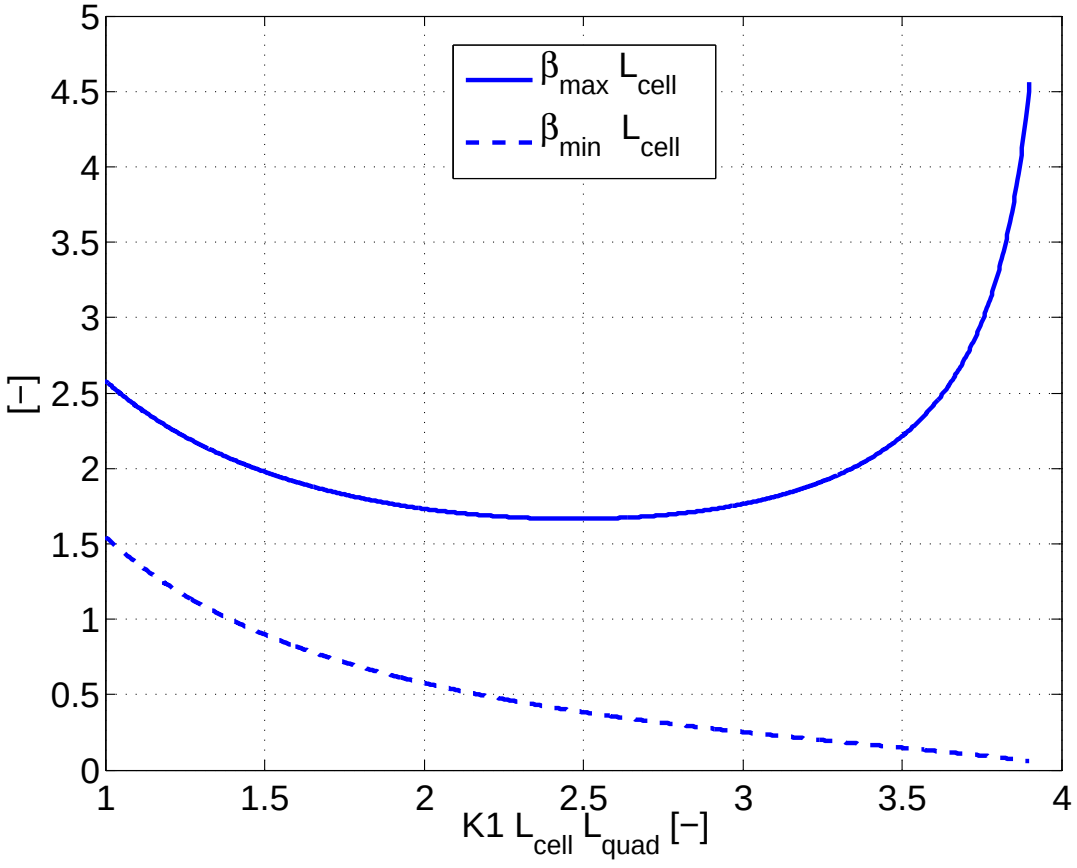


Figure 2: β-functions versus quad strength, cell length and quad length. Thin lens approximation of a FODO.

TUTORIAL 3: FIRST PART

The LHC FODO lattice

- Consider now that in the cell of Tutorial 2 there are 4 sector dipoles of 14.3 m. In the lattice there are a total of 736 dipoles with equal bending angles. Install the four dipoles in the FODO cell. Do the dipoles (weak focusing) affect on the β_{max} and the dispersion? Compute the relative variation on the β_{max} on the two planes.
- LHC has 8 octants, each one of 23 FODO cells. What is the phase advance contribution due to the octants in LHC?

TUTORIAL 3: SECOND PART

Changing the machine working point.

- Change the beam to $E_{tot} = 3.5$ TeV. What is the new tune of the machine? Why?
- Suppose you want to set a tune of (60.2, 67.2), match the FODO to get it. What is the maximum tune that you can reach with 23 cells/octant and 8 octants? (HINT: what it the maximum phase advance per FODO cell in thin approximation?...)

TUTORIAL 4: FIRST PART

Periodic solution and IC solution

- Build a transfer line of 10 m with 4 quads of L=0.4 m (centered at 2, 4, 6, and 8 m). With K1 respectively of 0.1, 0.1 , 0.1 , 0.1 m^{-2}. Can you find a periodic solution?
- Can you find a IC solution starting from $(\beta_x, \alpha_x, \beta_y, \alpha_y) = (1, 0, 2, 0)$?
- What is the final optical condition $(\beta_x^{end}, \alpha_x^{end}, \beta_y^{end}, \alpha_y^{end})$?

TUTORIAL 4: SECOND PART

Periodic solution and IC solution

- Starting from $(\beta_x, \alpha_x, \beta_y, \alpha_y) = (1, 0, 2, 0)$ match the line to $(\beta_x, \alpha_x, \beta_y, \alpha_y) = (2, 0, 1, 0)$ at the end.
- Starting from $(\beta_x, \alpha_x, \beta_y, \alpha_y) = (1, 0, 2, 0)$ and the gradient obtained with the previous matching, match to $(\beta_x^{end}, \alpha_x^{end}, \beta_y^{end}, \alpha_y^{end})$. Can you find back K1 respectively of 0.1, 0.1 , 0.1 , 0.1 m^{-2}?
- consider that the quadrupoles have an excitation current factor of 100 A/m^2 and an excitation magnetic factor of 100 T/m/A and aperture of 40 mm diameter. Compute the magnetic field at the poles of the four quads after matching (HINT: assume linear regime and use a dimensional approach).

Tutorial 5: First Part

LHC and MADX

- Run the MADX scripts.
- What is the LHC length? What is the s-position of IP1 and IP5? and the β-functions there?
- What are the beam1 and beam2 tunes at injections?

Tutorial 5: Second Part

LHC and MADX

- Are the two beams colliding in IP1 at injection?
- Is the crossing of the two beams vertical or horizontal in IP1 at collision?
- What are the beta function at the IPs at collision energy? Why do we inject with a higher β-function at the IPs?