Introduction to MAD-X

N. Fuster Martínez, IFIC (CSIC-UV)

Material based on Guido Sterbini, Öznur Mete and Bruce Yee courses

22nd of January 2024

GOAL

The goal of this course and workshop is to **work on** the understanding of **transverse beam dynamics concepts in magnetic lattices by taking a hands-on approach** (using the software MAD-X), complementing the Transverse Beam Dynamics course

DISCLAIMER

- ❑ This material is intended to be a very brief introduction to MAD-X: a large part of the code capabilities are not discussed in detail or are not discussed at all!

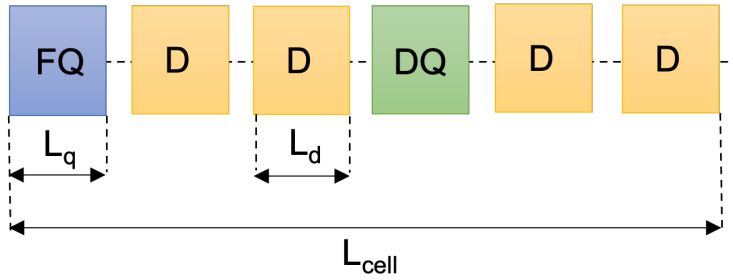
- ❑ If you want to deepen into the subject, you can find a lot of material in:
  - ✓ The official MAD-X web site: http://mad.web.cern.ch/mad/
  - ✓ An Introduction to Beam Physics: https://library.oapen.org/bitstream/handle/20.500.12657/50888/9781420011821.pdf?sequence=10

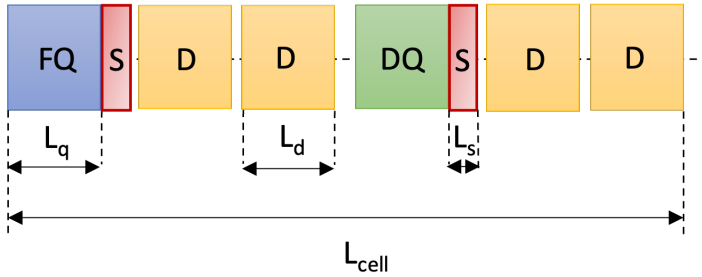
MAD-X workshop overview

- **50 minutes lecture** (now!)

- **First "hands-on" session (3h)** (22nd afternoon)
  - **Tutorial 1**: My first accelerator, a FODO cell
  - **Tutorial 2**: My first matching

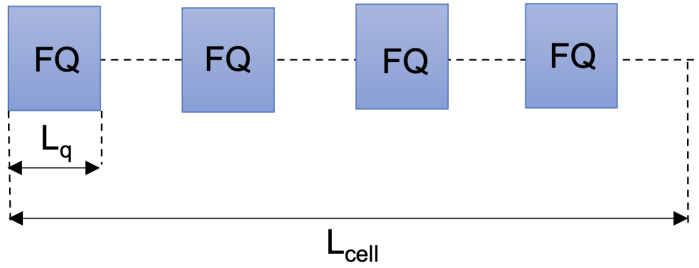
    30 minutes break
  - **Tutorial 3**: Building a circular machine





- **Second "hands-on" session (3h)** (23rd afternoon)
  - **Tutorial 4**: Natural chromaticity
  - **Tutorial 5**: Chromaticity correction and non-linearities

    30 minutes break
  - **Tutorial 6**: Building a transfer line



Tutors: Nuria Fuster-Martínez, Guido Sterbini, Davide Gamba, Javier Olivares

Outline

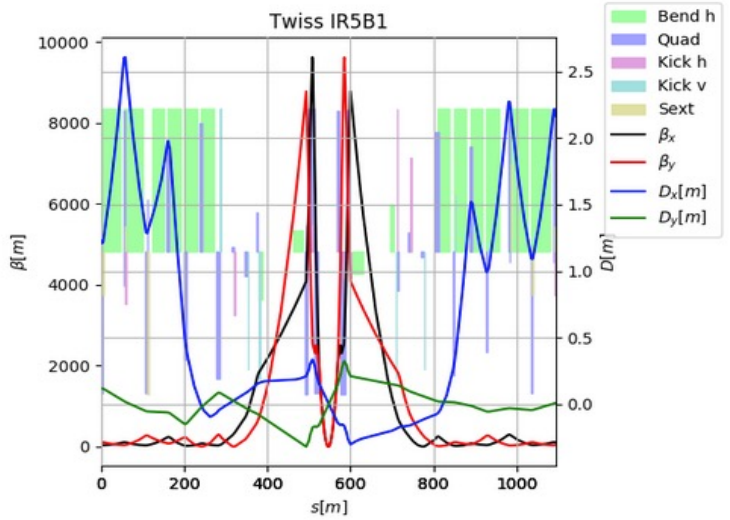
- ❑ Introduction. *Description of basic concepts*
- ❑ MAD-X language. *Syntax, variables*
- ❑ MAD-X commands. *Magnets, sequence and beam definition*
- ❑ Advanced MAD-X commands. *Twiss, matching and tracking*
- ❑ Example case. *A FODO cell*
- ❑ How to run MAD-X. *Interactive mode, batch mode and python interface*

Introduction

- *General purpose of a beam optics code.*
- *What is MAD-X?*
- *Do we use MAD-X for everything? No!*
- *Why MAD-X?*
- *How does it work?*
- *Source code: the basic components.*

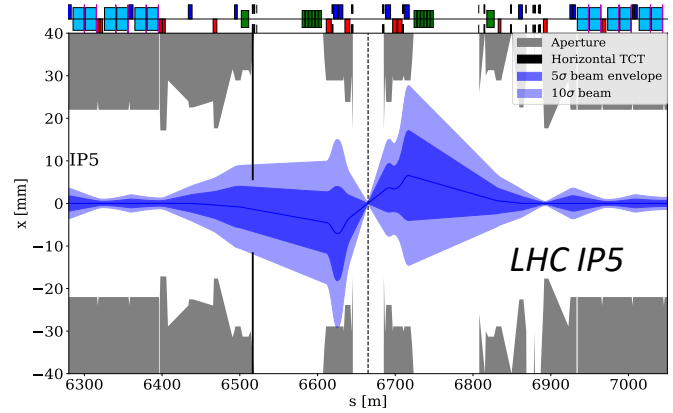
General purpose beam optics code

- Enables to design accelerators meeting specific research and application goals and to predict the behaviour of particle beams in those accelerators with a high degree of accuracy



- In such codes, the charge particle's equation of motion is solved under the effect of external EM fields from different accelerator components

- They are used to:
  - **Define the lattice** of circular or linear accelerators
  - Compute **linear optics functions** and **beam properties**
  - Design a lattice for getting the desired properties (**matching**)
  - **Simulate** accelerator **imperfections** and design **correction schemes**
  - Simulate **beam dynamics**: study single-particle motion



What is MAD-X?

```
+++++++++++++++++++++++++++++++++++++++++++
+     MAD-X 5.07.00  (64 bit, Darwin)     +
+ Support: mad@cern.ch, http://cern.ch/mad +
+ Release   date: 2021.05.03               +
+ Execution date: 2021.12.20 16:04:22      +
+++++++++++++++++++++++++++++++++++++++++++
>>>
```

- **MAD-X** (Methodical accelerator Design)
- A **general-purpose beam optics software**
- Distributed **for free by CERN** and used since more than 25 years for machine design and simulations (PS, SPS, LHC, linacs…)
- MAD-X is written in **C/C++/Fortran** (source code available under the CERN copyright)

Do we use MAD-X for everything? No!

We use MAD-X to:

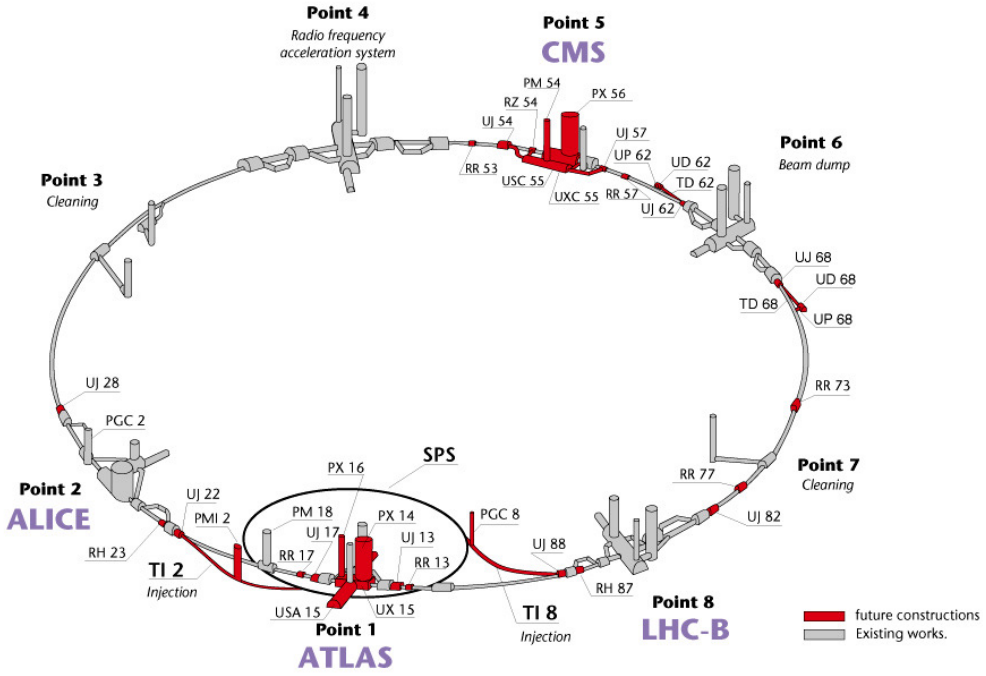
- ❑ Perform basic layout design and optimization
- ❑ Perform basic single-particle tracking and sensitivity analysis of linear and circular accelerators

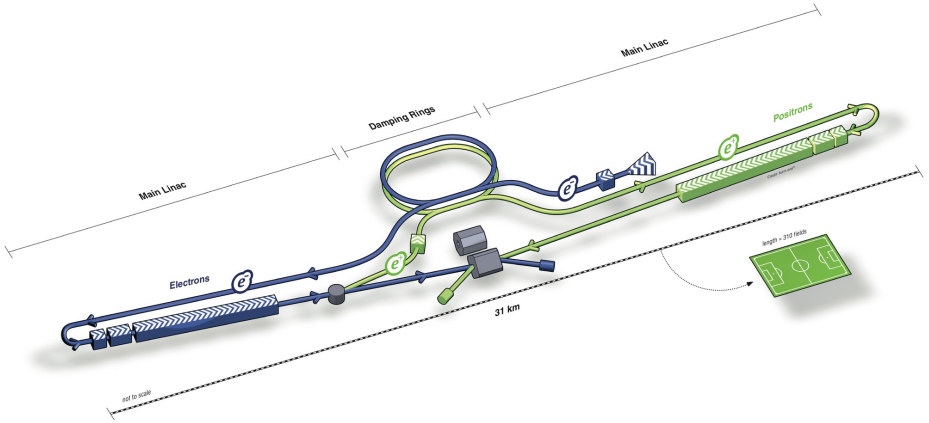
We don't use MAD-X for:

- ❑ Multi-particle and multi-bunch simulations
- ❑ Simulations requiring non static machines, i.e., beam changes its own environment (space charge, instabilities, beam-beam effects)
  - Often the programs used for these kind of studies use inputs from MAD-X

Why MAD-X?

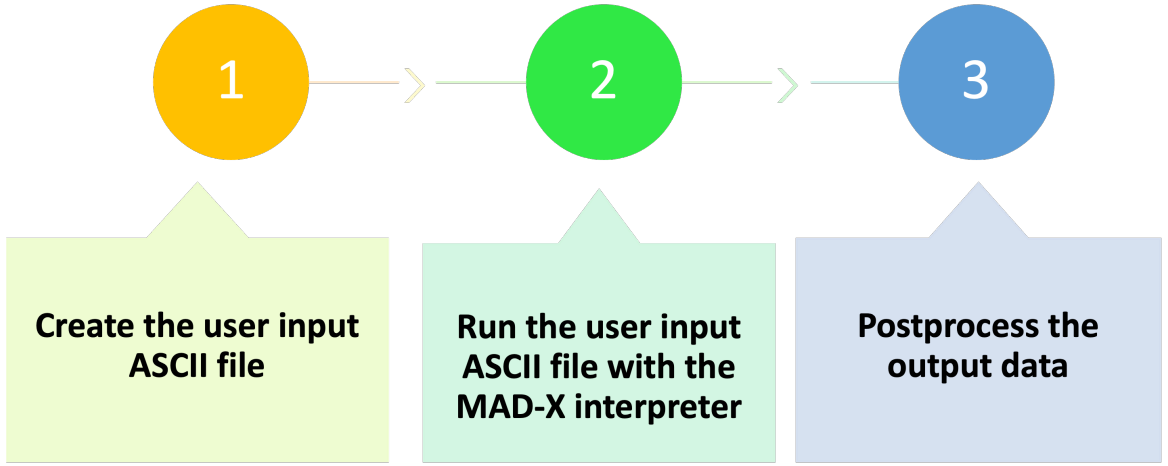
- **Multipurpose** from early to final stages of design studies.
- **Multiplatform** (Linux, OSX, WINDOWS).
- Source is **free** and very **flexible** and possible to extend.
- Developed for complicated applications, powerful and rather complete.
- Mainly **designed for large projects** (LEP, LHC, CLIC, ILC…) with $\geq 10^4$ elements.





How does it work?

- MAD-X is an **interpreter**
  - It accepts and executes **statements**.
  - Statements can be **actions** (optics…) **or assignments** (machine properties…)



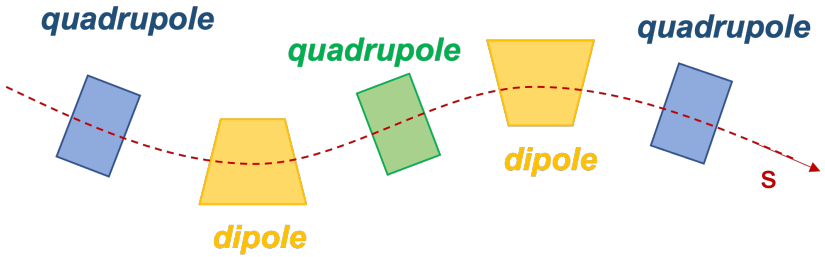
- What is the source code? **Via text**
  - It can be used in an **interactive way** (input from command line) or in **batch** (input from a file)

- MAD-X has its **own scripting language**, which is used to interface with the software.
  - Strong resemblance to “C” language (but NO need for declaration and NOT case sensitive)
  - Many features of programming language (loops, if, macros…)

Source code: the basic components

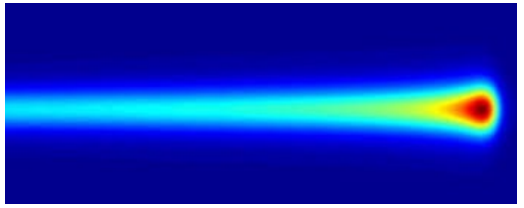
- **Description of the machine**
  - Accelerator elements, physical attributes and location….

  What is the machine in question?



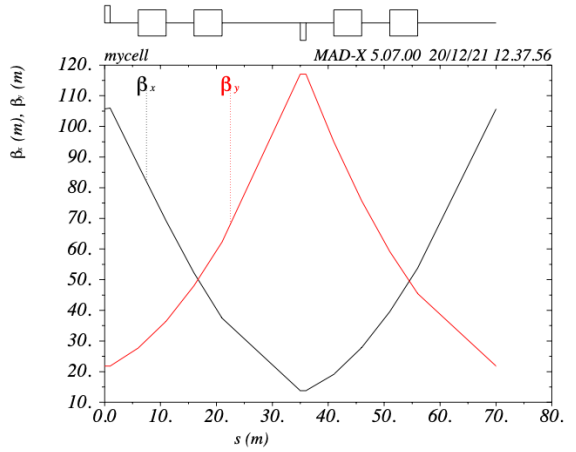
- **Description of the beam**
  - Type of particle, energy…

  What will be running in the machine?



- **Actions**
  - Optics calculation, matching, tracking….

  What do you want to study about the machine?



MAD-X language

- ❑ *Language features*
- ❑ *Conventions*
- ❑ *Optic variables*

Language features

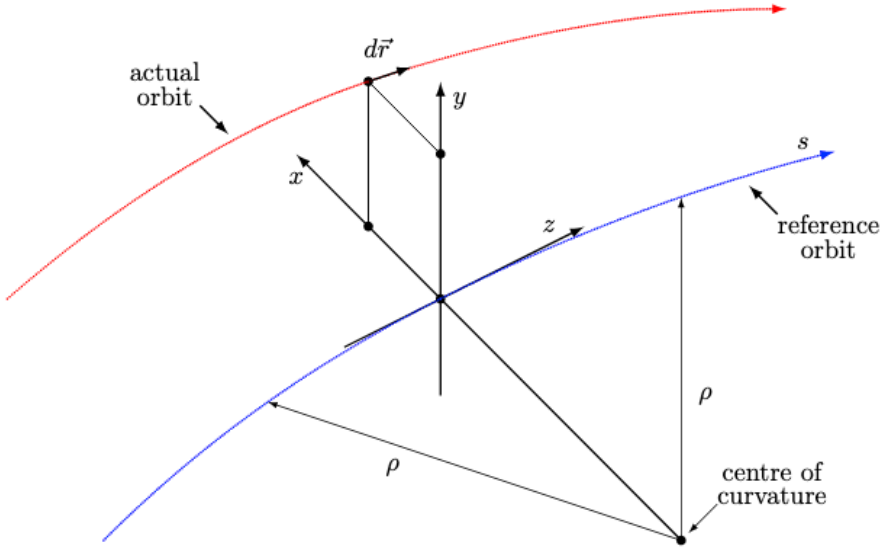
- All the sentence end with **semicolon ";"**
- **Arithmetic expressions**, including basic functions (exp, log, sin…), built-in random number generators and predefined constants (c, e, $\pi, m_e, m_p \ldots$)
- **Assignments**:
  - Regular (=). **a=b**, if **b** changes **a** does not
  - Deferred (:=). **a:=b**, if **b** changes **a** is updated too
- Variables can be used in **expressions**:
  - *NBEND = 40;*
  - *ANGLE=2*PI/NBEND;*
- **Comments:**
  - Start with two slashes (//) or an exclamation mark (!) for single lines
  - Are enclosed by (/*) and (*/) for multiple lines

Deferred expressions
Important for the variables you may need to change along the code (e.g. matching parameters…)!

Conventions

- **Units**: all parameters are in terms of **SI** units, except the **energy, momentum and mass** expressed in **GeV, GeV/c and GeV/c^2**
- **Horizontal** plane is assumed to be the **bending plane**

- **Elements are placed along the reference orbit moving along s**
  - x = y = 0 in a curvilinear system
  - The **reference orbit** is the path of a charge particle having the central momentum of the accelerator though idealised and perfectly aligned magnets
  - The closed orbit is described with respect to the reference orbit, using the local reference system (x, y, s) and includes all nonlinear effects



Main particle's coordinates and optics functions

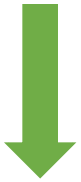
Description	MAD-X variable	Units
Horizontal and vertical position, x, y	x, y	[m]
Horizontal and vertical normalized momentum, $p_{x,y}/p_{0x,0y}$	px, py	[-]
Arc length, s	s	[m]
Momentum deviation from the design momentum, $\delta p/p_0$	deltap	[-]
Horizontal and vertical β-function, $\beta_{x,y}$	betx, bety	[m]
Horizontal and vertical α-function, $\alpha_{x,y}$	alfx, alfy	[-]
Horizontal and vertical phase advance, $\mu_{x,y}$	mux, muy	[2π]
Horizontal and vertical dispersion function, $D_{x,y}$	dx, dy	[m]

MAD-X commands

- *Generic pattern for MAD-X commands*
- *Definition of lattice elements*
  - *The strength of the magnets.*
- *The lattice sequence*
- *Basic MAD-X commands*

Generic pattern for MAD-X commands

label: keyword, {attributes};



Name given by the USER to the stored command.
If a label is given, MAD-X keeps this command in memory and can be used as
'**EXEC label**;'

Action desired.
Protected by MAD-X, cannot be used as label

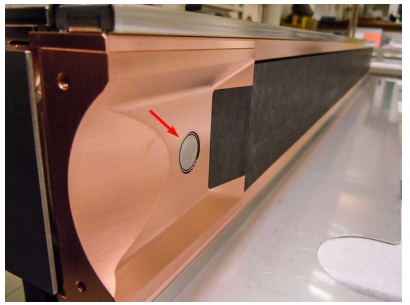
Defined data to the command.
String, logical, integer, expression or range selection. The items enclosed in {} can be omitted

Definition of lattice elements

❑ Each machine element or group of elements must be defined by a command.

Generic pattern to define an element

***label*: *keyword*, *properties*;**

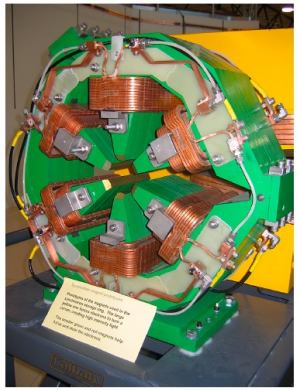


❑ MAD-X **keywords** are used to define the type of element:

- magnets, markers, RF cavities, collimators…

❑ Examples:

- Dipole magnet: ***MBL*: *SBEND*, *L=10.0*;**
- Quadrupole magnet: ***MQ*: *QUADRUPOLE*, *L=3.3*;**
- Sextupole magnet: ***MSF*: *SEXTUPOLE*, *L=1.0*;**
- *Marker:* ***MARKER1*: *MARKER*;**



The strength of the magnets

❑ The name of the parameter that defines the **magnetic strength** depends on the element:

- For a dipole magnet (horizontal bending) is: *angle [rad]*

  MBL: SBEND, ANGLE=0.05, L=10.0;

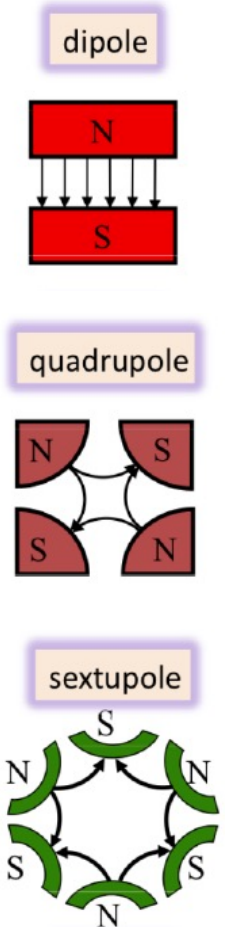
- For a quadrupole magnet is *:* $k_1 = \frac{1}{B\rho}\frac{\partial B_y}{\partial x}\,[\,m^{-2}]\left(=\frac{1}{lf}\right)$

  MQ: QUADRUPOLE, K1=0.00156, L=3.3;

- For a sextupole magnet is*:* $k_2 = \frac{1}{B\rho}\frac{\partial^2 B_y}{\partial x^2}\,[m^{-3}]$

  ksf= 0.00015;
  MSF: SEXTUPOLE, K2=ksf, L=1.0;

- Octupoles, multiple magnet “thin” element…

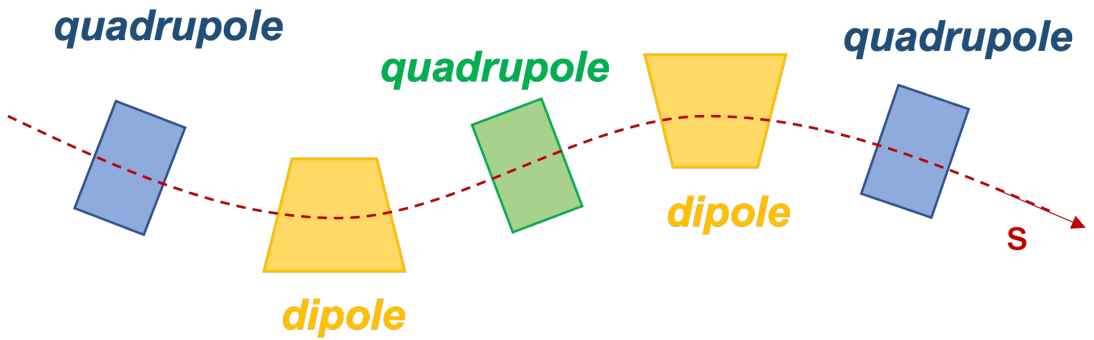


The lattice sequence

- Is an ordered collection of machine elements defining the accelerator to be studied
- The position of each element in the sequence can be defined with respect to the *CENTRE*, *EXIT* or *ENTRY* of the element, the start of the sequence or the position of another element
- Example:

```
#Element definition
MQ: QUADRUPOLE, L=3.3, K1=0.005;

# Sequence definition
mycell: SEQUENCE, REFER=ENTRY, L=10;
q1: MQ, at s=0;
marker1: marker, at s=2.5;
ENDSEQUENCE;
```



Basic useful MAD-X commands

- Beam definition:

  ***BEAM**, PARTICLE=particle_type (proton, electron…), ENERGY=value ;*

- A sequence has to be activated in order to be usable for other operations with the USE command:

  ***USE**, SEQUENCE=name_sequence;*

- Call an external file:

  ***CALL**, FILE=name_file.madx;*

- Print a value in the terminal:

  ***VALUE** variable_name;*

- Production of graphical output (e.g. β-function)*:

  ***PLOT**, HAXIS=s, VAXIS=betx,bety, COLOUR=100, FILE="test" ;*

** Note that the data must be generated first using the advances commands such as TWISS or TRACK.*

Questions on this first part?

Advances MAD-X commands

- ❑ *TWISS*. Optics calculations*
- ❑ *MATCHING*. Optics optimization*
- ❑ *TRACKING*. Beam dynamics studies*

*These commands require a **beam** and an **active sequence**.

TWISS

- The TWISS command will compute the **linear lattice function** (optical functions and the CO) around the machine and optionally the chromatic functions
- It is a **matrix code** where first and second order matrices are used to get the optics

```
# For a periodic solution
TWISS, SEQUENCE=sequence_label, exit(by default), FILE="test.txt", table=TWISS (by default);

# For an initial condition (IC) solution
TWISS, SEQUENCE=sequence_label, betx=1, alfx=0, bety=1, alfy=0;

# More attributes: DELTAP, CHROM…
```

- **After a successful TWISS run**, MAD-X creates a table of summary parameters named 'SUMM' which includes tunes, chromaticity, etc. as well as a 'TWISS' table and ASCII file if requested with most of the computed parameters

TWISS output

- **TWISS table**
  a=table(TWISS,q1,betx);

- **SUMM table**
  a=table(SUMM,dq1);

++ table: summ

length 70	orbit5 −0	alfa 0.1133597226	gammatr 2.97009686
q1 0.3240223299	dq1 −0.2894713472	betxmax 105.9701387	dxmax 18.28374049
dxrms 14.1747586	xcomax 0	xcorms 0	q2 0.2438418116
dq2 −0.2895087383	betymax 117.0503512	dymax 0	dyrms 0
ycomax 0	ycorms 0	deltap 0	synch_1 0
synch_2 0	synch_3 0	synch_4 0	synch_5 0
synch_6 0	synch_8 0	nflips 0	

Length of the machine

Closed orbit T (=ct) component, momentum compaction factor and transition energy

Horizontal and vertical tune

Horizontal and vertical chromaticity

Maximum horizontal and vertical β-function

Closed orbit and dispersion information

Momentum difference divided by the reference momentum

Synchrotron radiation

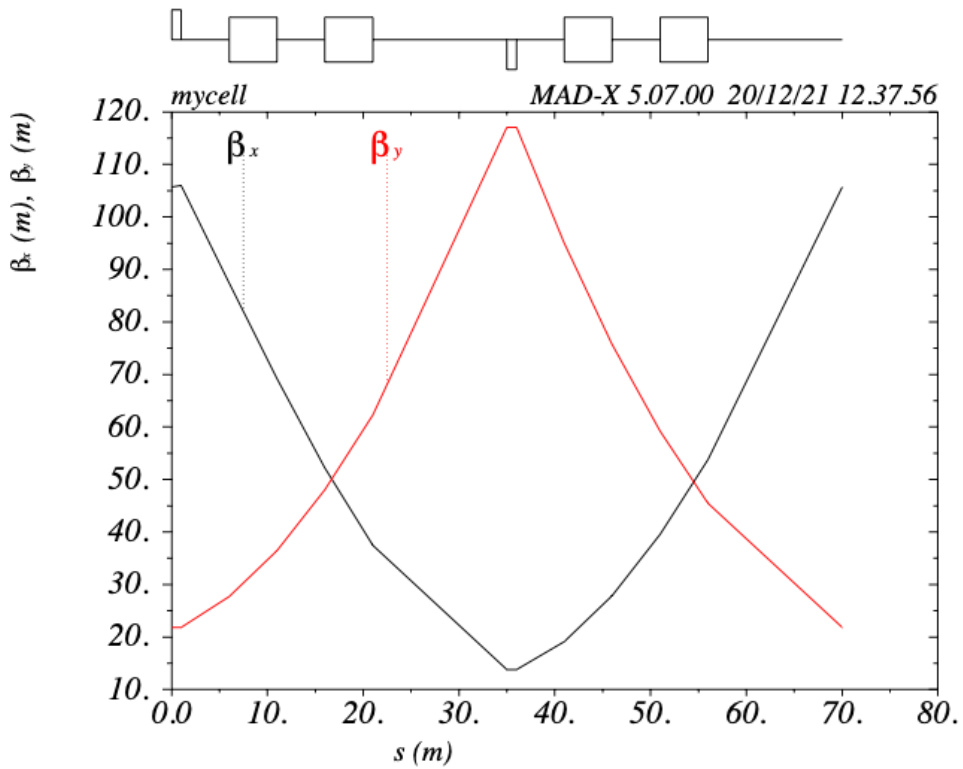
- **ASCII file.**
  - With most of the parameters computed by MAD-X
  - ***SELECT**, flag=TWISS, column=name, keyword, s, betx, alfx, mux, bety, alfy, muy;*

* NAME	KEYWORD	S	BETX	ALFX	MUX	BETY	ALFY	MUY
$ %s	%s	%le	%le	%le	%le	%le	%le	%le
"MYCELL$START"	"MARKER"	0	105.6855764	−2.259230089	0	21.7840267	0.4573523215	0
"Q1"	"QUADRUPOLE"	1	105.9701387	1.978472039	0.001493921339	21.7840267	−0.4573523215	0.00735746345
"DRIFT_0"	"DRIFT"	6	87.34479015	1.74659768	0.009769065687	27.74523085	−0.7348885096	0.03995368946
"B1"	"SBEND"	11	69.17896128	1.856667625	0.01997132749	36.48179689	−1.012424698	0.06507035749
"DRIFT_1"	"DRIFT"	16	52.21942634	1.535239362	0.0332266184	47.9937248	−1.289960886	0.08413368703
"B2"	"SBEND"	21	37.43543691	1.397224472	0.05118922695	62.2810146	−1.567497074	0.09870931652
"DRIFT_2"	"DRIFT"	35	13.77011836	0.2931554251	0.1569385083	117.0503512	−2.3445984	0.124924296
"Q2"	"QUADRUPOLE"	36	13.80615426	−0.3296705349	0.1685688634	117.0503512	2.3445984	0.126274979
"DRIFT_3"	"DRIFT"	41	19.11044739	−0.7311880903	0.2183687567	94.99204817	2.067062212	0.1338245632
"B3"	"SBEND"	46	27.82671663	−0.9977190119	0.2530083111	75.70910699	1.789526024	0.1432136723
"DRIFT_4"	"DRIFT"	51	39.59664721	−1.356267103	0.2770733231	59.20152769	1.511989836	0.1551111302
"B4"	"SBEND"	56	53.74757777	−1.450626954	0.294285777	45.46931027	1.234453648	0.1704728215
"DRIFT_5"	"DRIFT"	70	105.6855764	−2.259230089	0.3240223299	21.7840267	0.4573523215	0.2438418116
"MYCELL$END"	"MARKER"	70	105.6855764	−2.259230089	0.3240223299	21.7840267	0.4573523215	0.2438418116

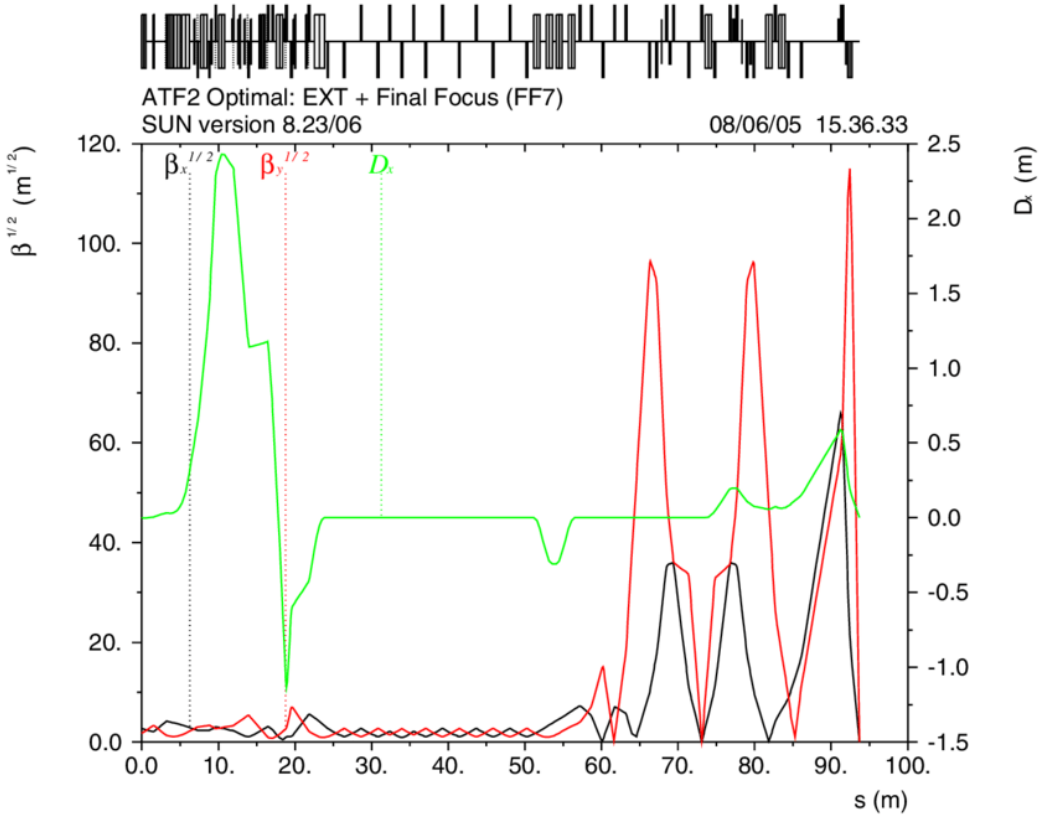
TWISS plot

- Production of **graphical output** of the main optical functions (e.g. β-function):

***PLOT**, HAXIS=s, VAXIS=betx,bety, COLOUR=100, FILE="test" ;*



FODO cell example



Transfer line example

Global matching

- It is possible to modify the optical parameters of a lattice using the **MATCHING module.**
- Adjusting the magnetic strengths to get the desired **global properties.**
  - Examples for global parameters: Q1 and Q2 (horizontal and vertical tunes) and dQ1 and dQ2 (horizontal and vertical chromaticity)

- **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;

ENDMATCH;
```

What we want!

What we change!

Methods for the minimization.

LMDIF accepts two attributes: **CALLS**, maximum number of calls to the penalty function; **TOLERANCES**, the desired tolerance for the minimum. http://mad.web.cern.ch/mad/releases/5.02.08/madxuguide.pdf

Other types of matching

- **Local matching** and **performance matching**:
  - Local optical functions (insertions, local optics changes)
  - Any user defined variable

- Example:

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

What we want!

What we change!

Tracking

- The TRACK module allows us to perform **single particle tracking** for given initial conditions
- It is based on building transfer maps by solving the Hamiltonian equations of each element
- Example

```
SELECT, FLAG=makethin, SLICE=1;
MAKETHIN, SEQUENCE=seq_name;
```

} **Convert the thick lattice to a thin lens one!**

```
TRACK, dump, file="name", DELTAP=0.01;
```

Off-momentum!

```
   START, x=1e-3, px=0, y=1e-3, py=0;
   START, x=1e-2, px=0, y=1e-3, py=0;
```

Particles initial conditions!

```
   run, turns=100;
```

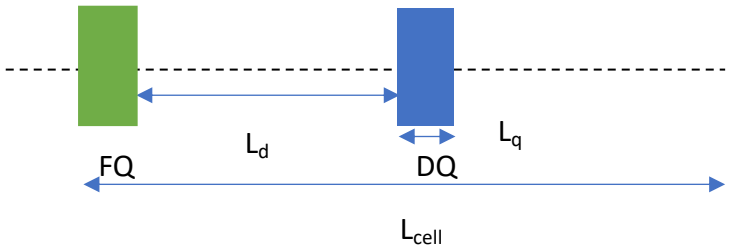
Number of turns!

- **Output:** *'TRACK' table, file.txt*
- **Plot:**

```
PLOT, file="track1",table=track, haxis=x, vaxis=px, colour=100;
```

Example case

What is the machine in question?



What will be running in the machine?

What do you want to study about the machine?

```
! ************************************************************************
! Definition of parameters
! ************************************************************************
l_cell=100;
quadrupoleLenght=5;
myK:=0.0012;// m^-2

! ************************************************************************
! Definition of magnets
! ************************************************************************
QF: quadrupole, L=quadrupoleLenght, K1:=myK;
QD: quadrupole, L=quadrupoleLenght, K1:=-myK;

! ************************************************************************
! Definition of sequence
! ************************************************************************
myCell:sequence, refer=entry, L=L_CELL;
quadrupole1: QF, at=0;
marker1: marker, at=25;
quadrupole2: QD, at=50;
endsequence;
```

```
! ************************************************************************
! Definition of beam
! ************************************************************************
beam, particle=proton, energy=2;
```

```
! ************************************************************************
! Use of the sequence
! ************************************************************************
use, sequence=myCell;
! ************************************************************************
! TWISS
! ************************************************************************
select, flag=twiss, column=[name, keyword,betx, bety, alfx, alfy];
twiss, file=MyfirstFODO.madx;
plot, haxis=s, vaxis=betx,bety,dx,colour=100,file=MyfirstFODO;
```

How to run MAD-X

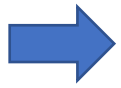
- ❑ *Interactive mode*
- ❑ *Batch mode*
- ❑ *Python interface*

How to run MAD-X: interactive mode

- You download the last release from the repository and follow the instruction in: http://madx.web.cern.ch/madx/

- In Windows, run the executable.

- In Linux (OXS), execute *./madx* in the containing directory.



Usually not optimum…

```
(base) lapnuria:MADX nuria$ ./madx

  ++++++++++++++++++++++++++++++++++++++++++++
  +     MAD-X 5.07.00  (64 bit, Darwin)      +
  + Support: mad@cern.ch, http://cern.ch/mad +
  + Release   date: 2021.05.02               +
  + Execution date: 2021.10.07 18:06:19      +
  ++++++++++++++++++++++++++++++++++++++++++++

X:> angle = 2*pi/1232;
X:> value, angle;
angle             =     0.005099988074 ;
X:> dx=gauss()*2;
X:> value, dx;
dx                =        1.295621501 ;
X:> value, dx;
dx                =        1.295621501 ;
X:> dx:=gauss()*2;
```

How to run MAD-X: batch mode

- After writing your script in a separate file *myfile.madx*, you can call it in Windows after opening MAD-X:

```
madx
X: ==> call, file=myfile.madx;
```

- In Linux, you can also type in the terminal:

```
./madx myfile.madx
```

myfile.madx

```
! ****************************************************************
! Definition of parameters
! ****************************************************************

l_cell=100;
quadrupoleLenght=5;
f=200;
myK:=1/f/quadrupoleLenght;// m^-2

! ****************************************************************
! Definition of magnets
! ****************************************************************
QF: quadrupole, L=quadrupoleLenght, K1:=myK;
QD: quadrupole, L=quadrupoleLenght, K1:=-myK;

! ****************************************************************
! Definition of sequence
! ****************************************************************
myCell:sequence, refer=entry, L=L_CELL;
quadrupole1: QF, at=0;
marker1: marker, at=25;
quadrupole2: QD, at=50;
endsequence;

! ****************************************************************
! Definition of beam
! ****************************************************************
beam, particle=proton, energy=2;

! ****************************************************************
! Use of the sequence
! ****************************************************************
use, sequence=myCell;

! ****************************************************************
! TWISS
! ****************************************************************
title, 'My first twiss';
twiss, file=MyfirstFODO.madx;
plot, haxis=s, vaxis=betx,bety,dx,colour=100, title="test",file=MyfirstFODO;
'''
```

How to run MAD-X: python interface I

- During the workshop we will use MAD-X trough a **python jupyter interface** based on the python scripting language

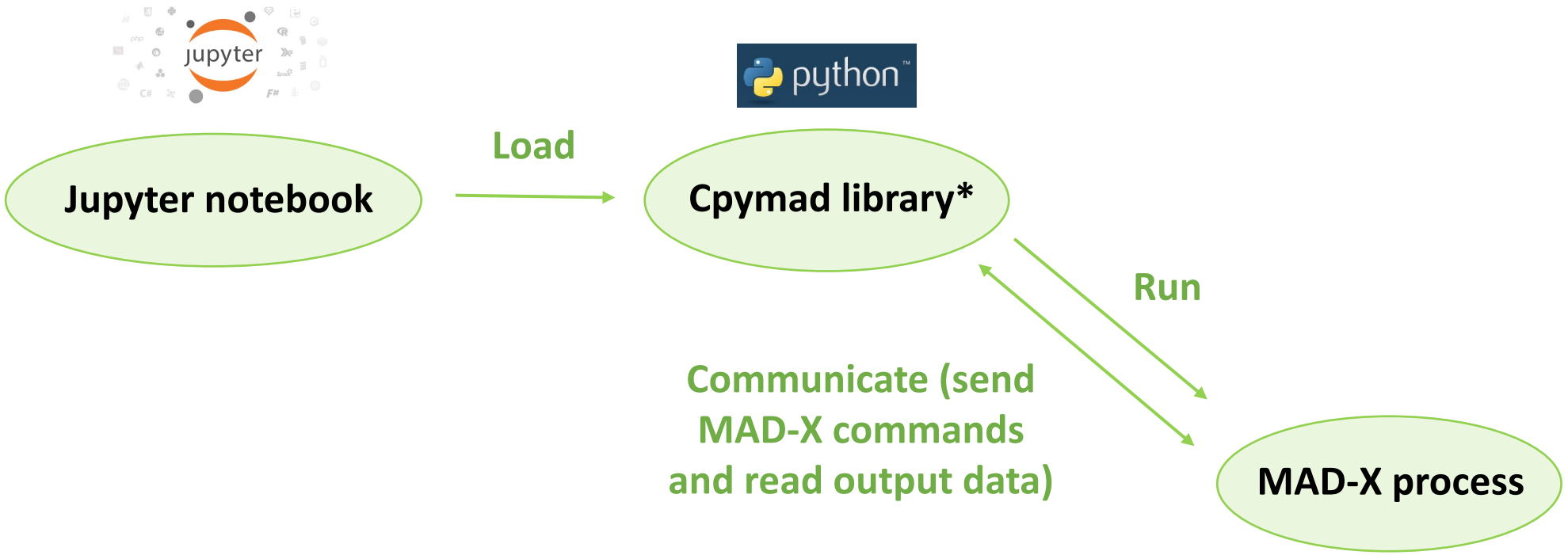
Why?

- MAD-X has no graphical interface and for detailed processing of MAD-X results, other programming languages are often used

- Python is widely used in the physics community and provides powerful numerical and plotting libraries

- Jupyter is web application that allows to create and share live code

How to run MAD-X: python interface II



*It is a binding to MAD-X for giving full control and access to a MAD-X interpreter in python
http://hibtc.github.io/cpymad/getting-started

```
++++++++++++++++++++++++++++++++++++++++++++
+     MAD-X 5.07.00  (64 bit, Darwin)      +
+ Support: mad@cern.ch, http://cern.ch/mad +
+ Release   date: 2021.05.03               +
+ Execution date: 2021.12.20 16:04:22      +
++++++++++++++++++++++++++++++++++++++++++++
```

How to run MAD-X: python interface III

- Load cpymad library*
- Run a MAD-X process
- Define lattice input file
- Use of cpymad.madx methods (call, input, table.twiss.dframe, table.sum.dframe)
- Data analysis (plots…)

jupyter-notebook

```
from matplotlib import pyplot as plt

from cpymad.madx import madx
```

```
madx=Madx()
```

```
madx.call("lattice.madx")
```

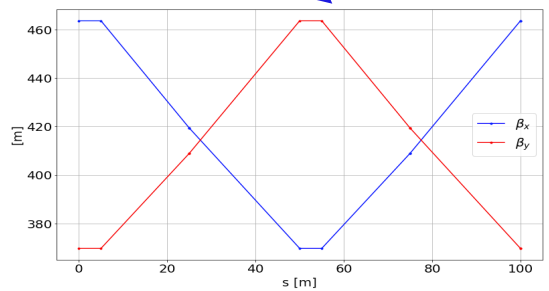
```
myString=
'''
beam, particle=proton, energy=2;
use, sequence=myCell;
twiss;
'''

madx.input(myString)
```

MAD-X commands

```
myTwiss=madx.table.twiss.dframe()

plt.plot(myTwiss["s"],myTwiss["betx"])
```



lattice.madx (ASCII file)

```
! ********************************************
! Definition of parameters
! ********************************************

l_cell=100;
quadrupoleLenght=5;
f=200;
myK:=1/f/quadrupoleLenght;// m^-2

! ********************************************
! Definition of magnets
! ********************************************
QF: quadrupole, L=quadrupoleLenght, K1:=myK;
QD: quadrupole, L=quadrupoleLenght, K1:=-myK;

! ********************************************
! Definition of sequence
! ********************************************
myCell:sequence, refer=entry, L=L_CELL;
quadrupole1: QF, at=0;
marker1: marker, at=25;
quadrupole2: QD, at=50;
marker2: marker, at=75;
endsequence;
```

	name	keyword	s	betx	bety
#s	mycell$start:1	marker	0.0	463.623288	369.779162
quadrupole1	quadrupole1:1	quadrupole	5.0	463.623288	369.779162
drift_0[0]	drift_0:0	drift	25.0	419.394867	408.967742
marker1	marker1:1	marker	25.0	419.394867	408.967742
drift_1[0]	drift_1:0	drift	50.0	369.779162	463.623288
quadrupole2	quadrupole2:1	quadrupole	55.0	369.779162	463.623288

How to run MAD-X: python interface IV

- We provided you with the instructions to **set-up the working environment:** https://fusterma.github.io/JUAS2024/

JUAS2024

MAD-X workshop JUAS2024

N. Fuster-Martinez, D. Gamba, G. Sterbini, S. Kostoglou, J. Olivares

For the MAD-X workshop at JUAS 2024 we will be using the **MAD-X accelerator design software**, together with **Python** as scripting language and **Jupyter** as interface for the analysis.

In order to be ready for the workshop please follow the instructions **before coming** to JUAS to **prepare yourself** and **your laptop** for the course.

For the workshop we will ask you to download the last version of the MAD-X Workshop JUAS2024 repository. In this repository you will find one folder for each tutorial with the corresponding jupyter-notebook and MAD-X input files to be completed during the workshop as well as the corresponding solutions.

The questions for each tutorial can be found here.

A list with the main MAD-X, cpymad and Python commands required to solved the questions of the proposed solutions can be found here.

- Once the environment is set-up you can **download** the last version of the **"MAD-X Workshop JUAS2024 repository"** and open a JupyterLab interface and start working on the tutorials!

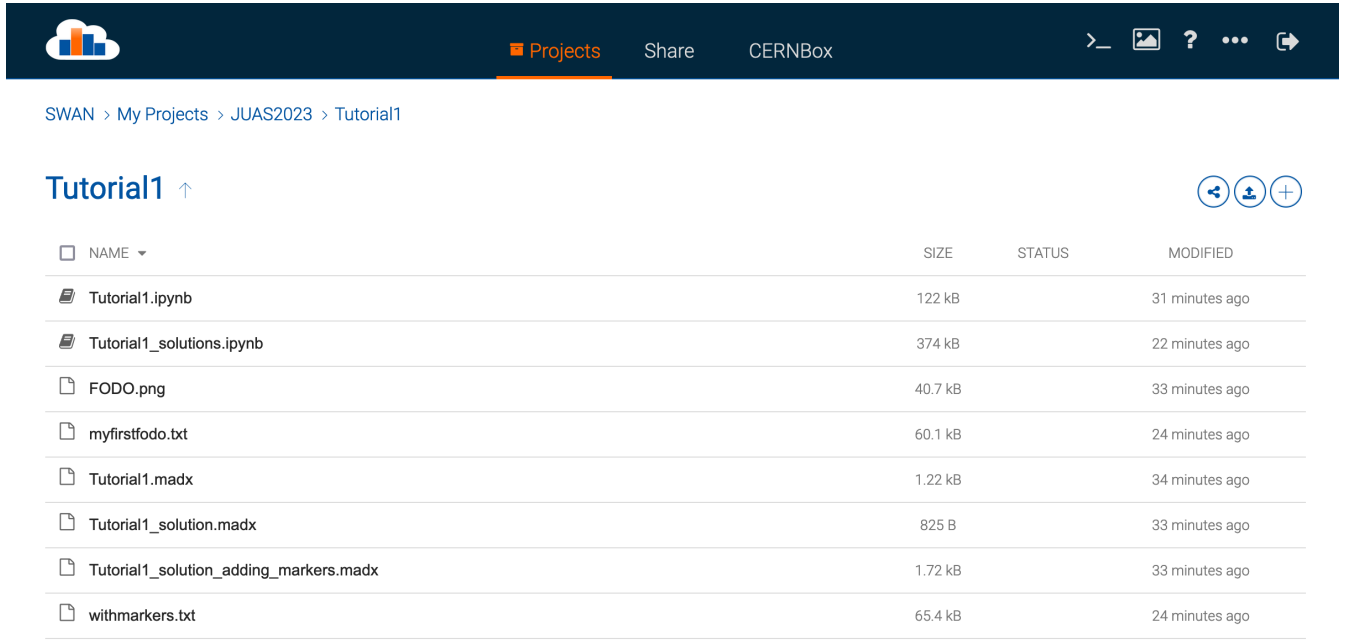
How to run MAD-X: python interface V

❑ **If you had problems with the environment set-up:**

- You can work using the SWAN CERN service using the CERN computing account

❑ **SWAN CERN service:**

- https://swan.cern.ch
- Log-in with your credentials
- Create a new project, “JUAS 2024”
- Load the folders and files in the “MAD-X Workshop JUAS2024 repository” into the new project.
- Open the first tutorial jupyter-notebook “Tutorial1.ipynb”



Thank you very much for your attention!

Questions?

Let’s finish with a quick quiz!

Kahoot!