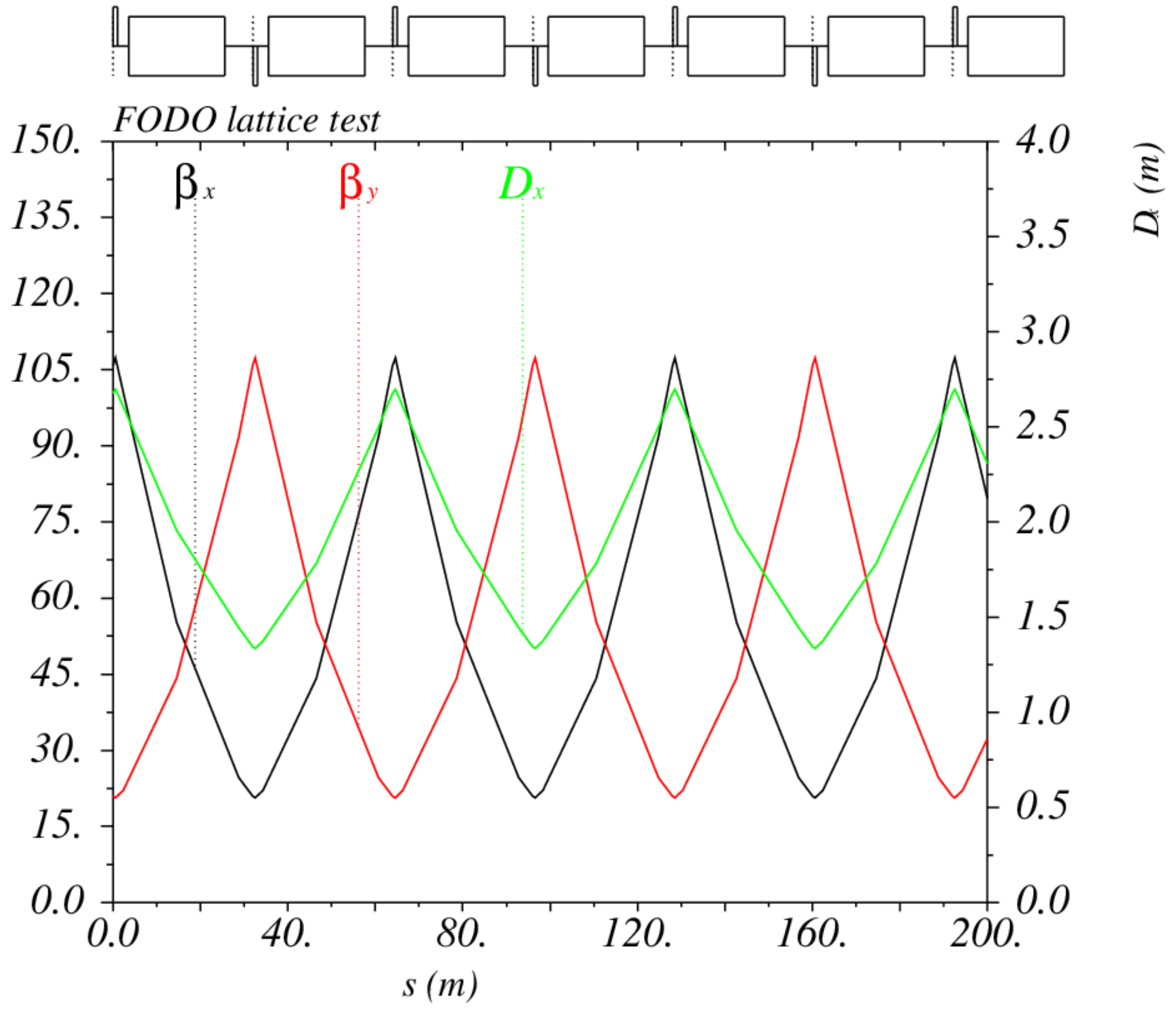
Accelerator Lattice Design – Part I Lattices

Ralph J. Steinhagen, CERN



- Acknowledgements and credits to: W. Herr, B. Holzer, A. Streun, A. Wolski

Overview

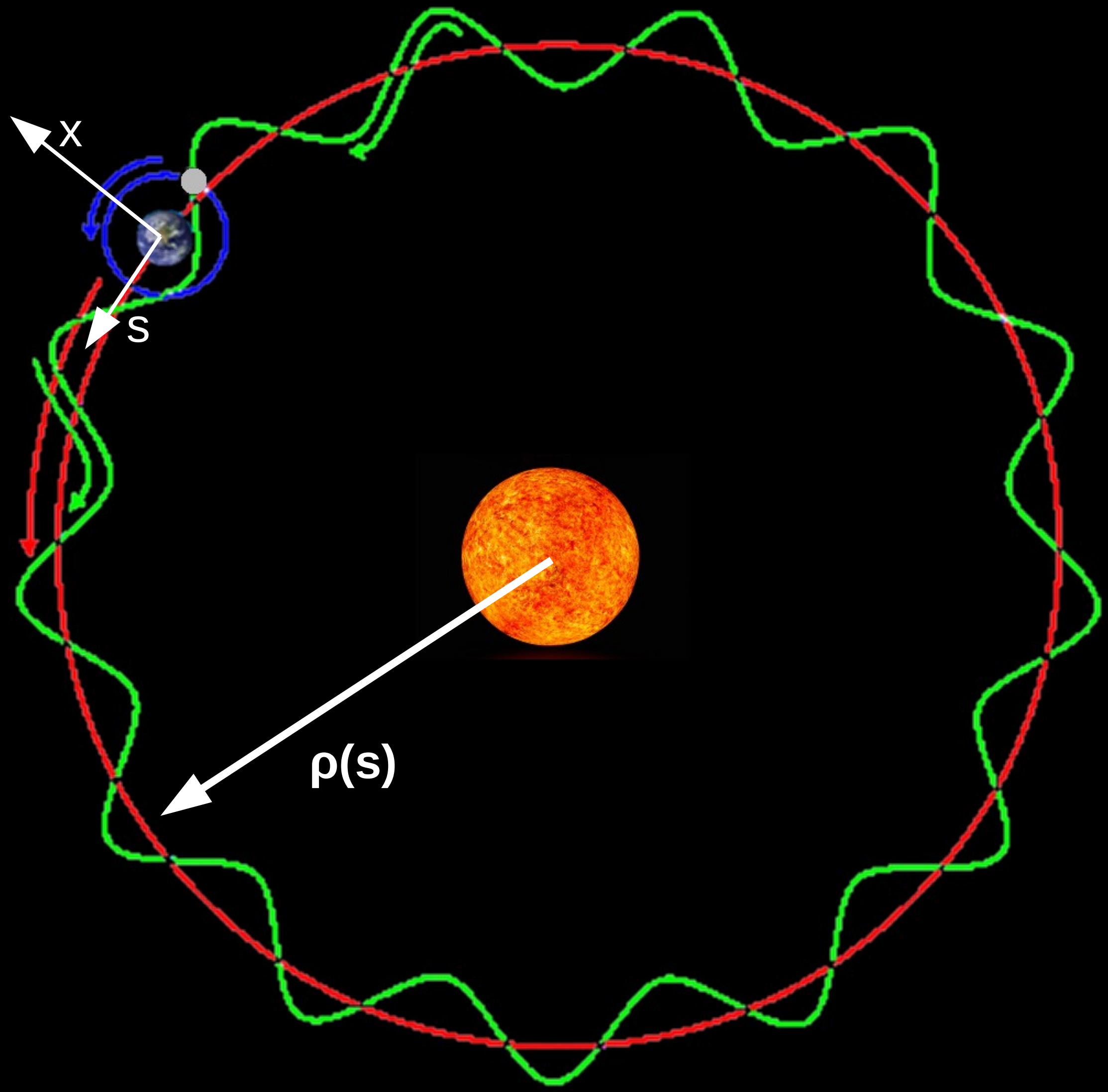
- Part I
  - Recap: basic concepts, terminology and MAD-X language
  - design a simple FoDo lattice & compute optical functions
    - betatron functions, dispersion
  - compute and correct tune, chromaticity
  - particle tracking
  - simple fitting

- Part II
  - Design of insertions
    - Dispersion suppressor
    - Low-β insertion

Question:

Does the Moon revolve around the Earth or the Sun?

Moon's Trajectory around Sun

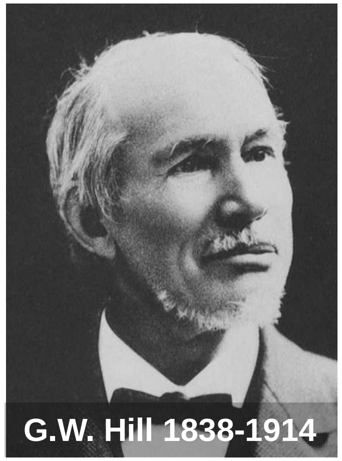


Recap: Transverse Beam Dynamics I/IV

- Hill's equation[1,2]:

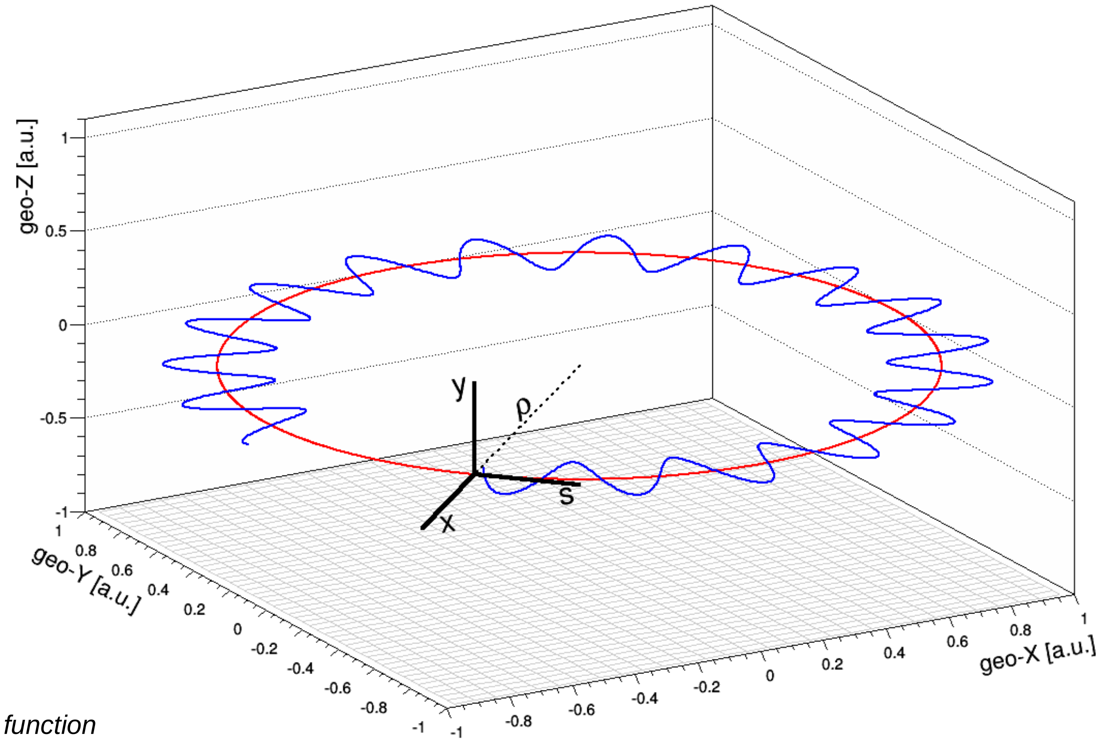
$$\boxed{z'' + K(s)\cdot z = f(s,t)}$$

$$K(s) = \underbrace{\left(\frac{q}{p}B_{dipole}\right)^2}_{weak\ focusing:\frac{1}{\rho^2}} - \underbrace{\frac{q}{p}\frac{\partial B_y}{\partial x}}_{strong\ focusing: k(s)}$$


G.W. Hill 1838-1914

  – k(s): focusing strength, defines:
    - betatron function β(s) → envelope of the oscillation
    - dispersion function D(s) → trajectory for off-momentum $\Delta p/p_0$ particles

  – f(s,t): external driving force



[1]George William Hill, *"On the part of the motion of the lunar perigee which is a function of the mean motions of the sun and moon"*, Acta Mathematica, 8:1–36, 1886
[2]coordinate 'z' being place holder for either x,y

Recap: Transverse Beam Dynamics II/IV

- Defines add. 'Twiss' functions[1]: betatron phase advance μ(s), α(s) & γ(s)

$$\boxed{\Delta\mu(s):=\int_0^s \frac{1}{\beta(s')}ds' \qquad \alpha(s):=-\frac{\beta'(s)}{2} \qquad \gamma(s):=\frac{1+\alpha^2(s)}{\beta(s)}}$$

- First-order solution to Hill's equation:

$$z(s) = \underbrace{z_{co}(s)}_{\textit{closed orbit}} + \underbrace{D(s)\cdot\frac{\Delta p}{p_0}}_{\textit{dispersion orbit}} + \underbrace{z_\beta(s)}_{\textit{betatron oscillations}}$$

→ sinusoidal particle motion in accelerators:

$$\boxed{z_\beta(s)=\sqrt{\epsilon_i\beta(s)}\cdot\sin(\mu(s)+\phi_i)}$$

ε_i, Φ_i : initial particle state

- Courant-Synder invariant of motion (↔ energy conservation)

$$\epsilon=\beta(s)\cdot x'^2+2\alpha(s)\cdot xx'+\gamma(s)\cdot x^2$$

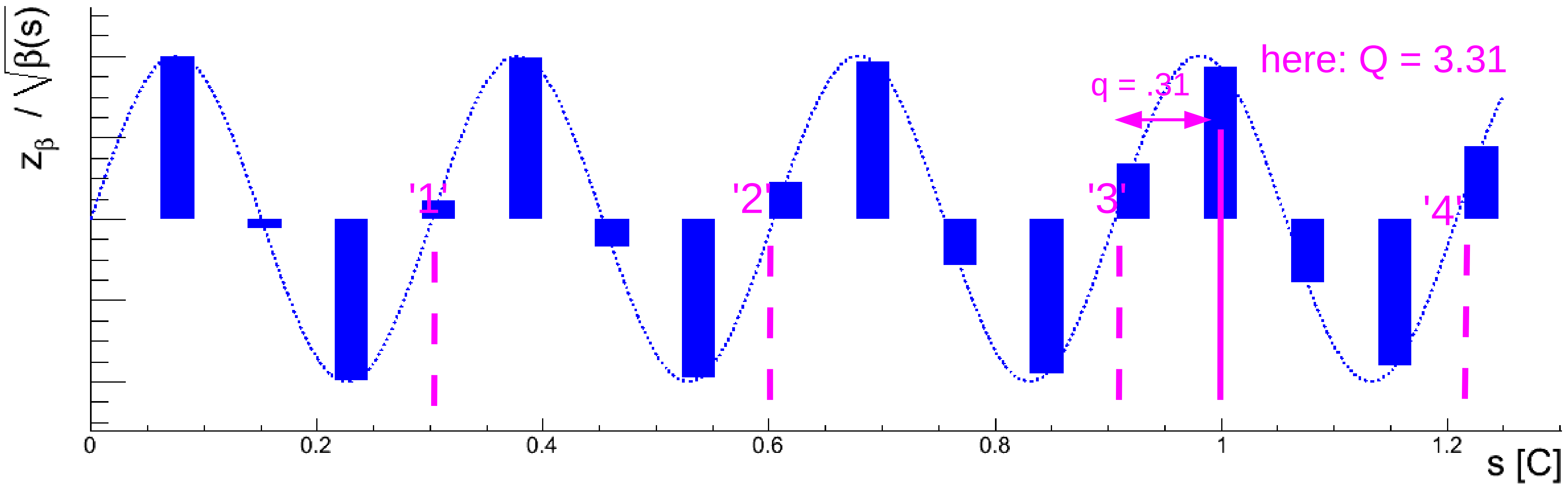
[1]Richard Q. Twiss and N. H. Frank, “Orbital stability in a proton synchrotron”, Rev. Sci. Instr., 20(1):1–17, January 1949.

shorthand: $x'=\frac{dx}{ds}$

Recap: Transverse Beam Dynamics III/IV

- Free Betatron Oscillations:

$$\boxed{z_\beta(s)=\sqrt{\epsilon_i\beta(s)}\cdot\sin(\mu(s)+\phi_i)}$$



- Betatron Phase Advance: Δμ(s)

Tune defined as betatron phase advance over one turn:

$$\boxed{Q := \frac{1}{2\pi}\oint_C \mu(s)\, ds}$$

common: $Q = \underbrace{Q_{int}}_{integer\ \ tune} + \underbrace{q_{frac}}_{fractional\ \ tune}$

Recap: Transfer Matrix Formalism

- Transfer matrix M
  - describes the transformation of amplitude x and angle x' through a number of lattice elements
  - ... and can be expressed by the optics parameters:

$$\begin{pmatrix} x \\ x' \end{pmatrix}_s = M \begin{pmatrix} x \\ x' \end{pmatrix}_0$$

$$M = \begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix} = \begin{pmatrix} \sqrt{\frac{\beta}{\beta_0}}\left(\cos\Delta\mu + \alpha_0 \sin\Delta\mu\right) & \sqrt{\beta\beta_0}\sin\Delta\mu \\ \frac{(\alpha_0 - \alpha)\cos\Delta\mu - (1 + \alpha_0\alpha)\sin\Delta\mu}{\sqrt{\beta\beta_0}} & \sqrt{\frac{\beta_0}{\beta}}\left(\cos\Delta\mu - \alpha\sin\Delta\mu\right) \end{pmatrix}$$

- Any other

$$M = M_n \cdot \ldots \cdot M_2 \cdot M_1$$

- Stability criterion: $|trace(M)| < 2$

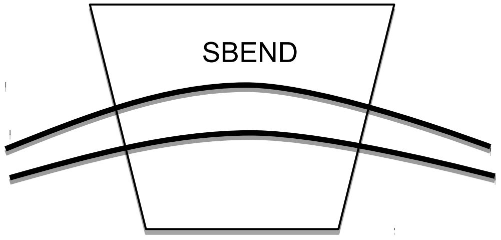
Recap: Linear Elements

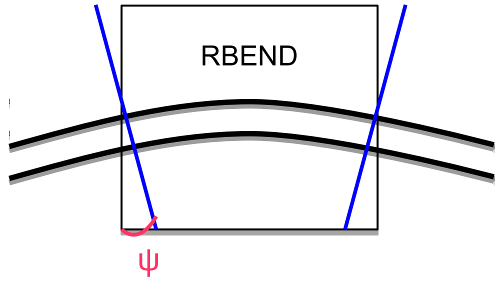
- Drift space length *l*:

$$M_{drift} = \begin{pmatrix} 1 & l \\ 0 & 1 \end{pmatrix}$$

- Dipole magnet:
  - ρ: bending radius, *l:* length, ψ: edge-tilt (focusing effect)

$$\boxed{\frac{1}{\rho} = \frac{q}{p} B}$$





$$M_{dipole} = \begin{pmatrix} \cos\frac{l}{\rho} & \rho\sin\frac{l}{\rho} \\ -\frac{1}{\rho}\sin\frac{l}{\rho} & \cos\frac{l}{\rho} \end{pmatrix} \quad \text{and:} \quad M_{edge-focusing} = \begin{pmatrix} 1 & 0 \\ -\frac{\tan\psi}{\rho} & 1 \end{pmatrix}$$

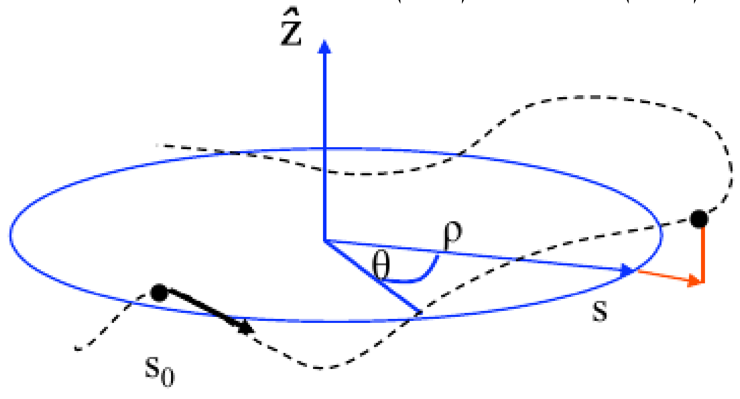
- (De-)Focusing Quadrupole magnet:

$$\boxed{k(s) = \frac{q}{p}\frac{\partial B_y}{\partial x}}$$

$$M_{Q[F,D]} = \begin{pmatrix} \cos(\sqrt{k}\cdot l) & \frac{1}{\sqrt{k}}\sin(\sqrt{k}\cdot l) \\ \mp\sqrt{k}\sin(\sqrt{k}\cdot l) & \cos(\sqrt{k}\cdot l) \end{pmatrix} \underbrace{\rightarrow}_{f=\frac{1}{kl}\gg l} M_{Q[F,D]} = \begin{pmatrix} 1 & 0 \\ \pm\frac{1}{f} & 1 \end{pmatrix}$$

Recap: Transfer of Optics Parameter

$$\begin{pmatrix} x \\ x' \end{pmatrix}_s = M \begin{pmatrix} x \\ x' \end{pmatrix}_0$$



- Conservation of emittance

$$\epsilon = \beta x'^2 + 2\alpha x x' + \gamma x^2$$
$$\epsilon = \beta_0 x'^2_0 + 2\alpha_0 x_0 x'_0 + \gamma_0 x^2 \qquad \textbf{(1)}$$

- Express x_0, x'_0 as a function of x,x'

$$\begin{pmatrix} x \\ x' \end{pmatrix}_0 = M^{-1} \begin{pmatrix} x \\ x' \end{pmatrix} \rightarrow \begin{matrix} x_0 = m_{22} x - m_{12} x' \\ x_0' = -m_{21} x + m_{11} x' \end{matrix} \qquad \textbf{(2)}$$

- Inserting (2) into (1), sorting via x,x', the rest is math ...

$$\begin{pmatrix} \beta \\ \alpha \\ \gamma \end{pmatrix} = \begin{pmatrix} m_{11}^2 & -2 m_{11} m_{12} & m_{12}^2 \\ -m_{11} m_{21} & m_{12} m_{21} + m_{22} m_{11} & -m_{12} m_{22} \\ m_{12}^2 & -2 m_{22} m_{21} & m_{22}^2 \end{pmatrix} \cdot \begin{pmatrix} \beta \\ \alpha \\ \gamma \end{pmatrix}_0$$

Most Simple Example – Drift Space

$$M_{drift}=\begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix}=\begin{pmatrix} 1 & L \\ 0 & 1 \end{pmatrix}$$

- Particle coordinates:

$$\begin{pmatrix} x \\ x' \end{pmatrix}=\begin{pmatrix} 1 & L \\ 0 & 1 \end{pmatrix}\begin{pmatrix} x \\ x' \end{pmatrix}_0$$

$$\boxed{\begin{aligned} x(L)&=x_0+L\cdot x'_0 \\ x'(L)&=x'_0 \end{aligned}}$$

- Transformation of Twiss parameters

$$\begin{pmatrix} \beta \\ \alpha \\ \gamma \end{pmatrix}=\begin{pmatrix} 1 & -2\mathrm{L} & L^2 \\ 0 & 1 & -L \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} \beta \\ \alpha \\ \gamma \end{pmatrix}_0$$

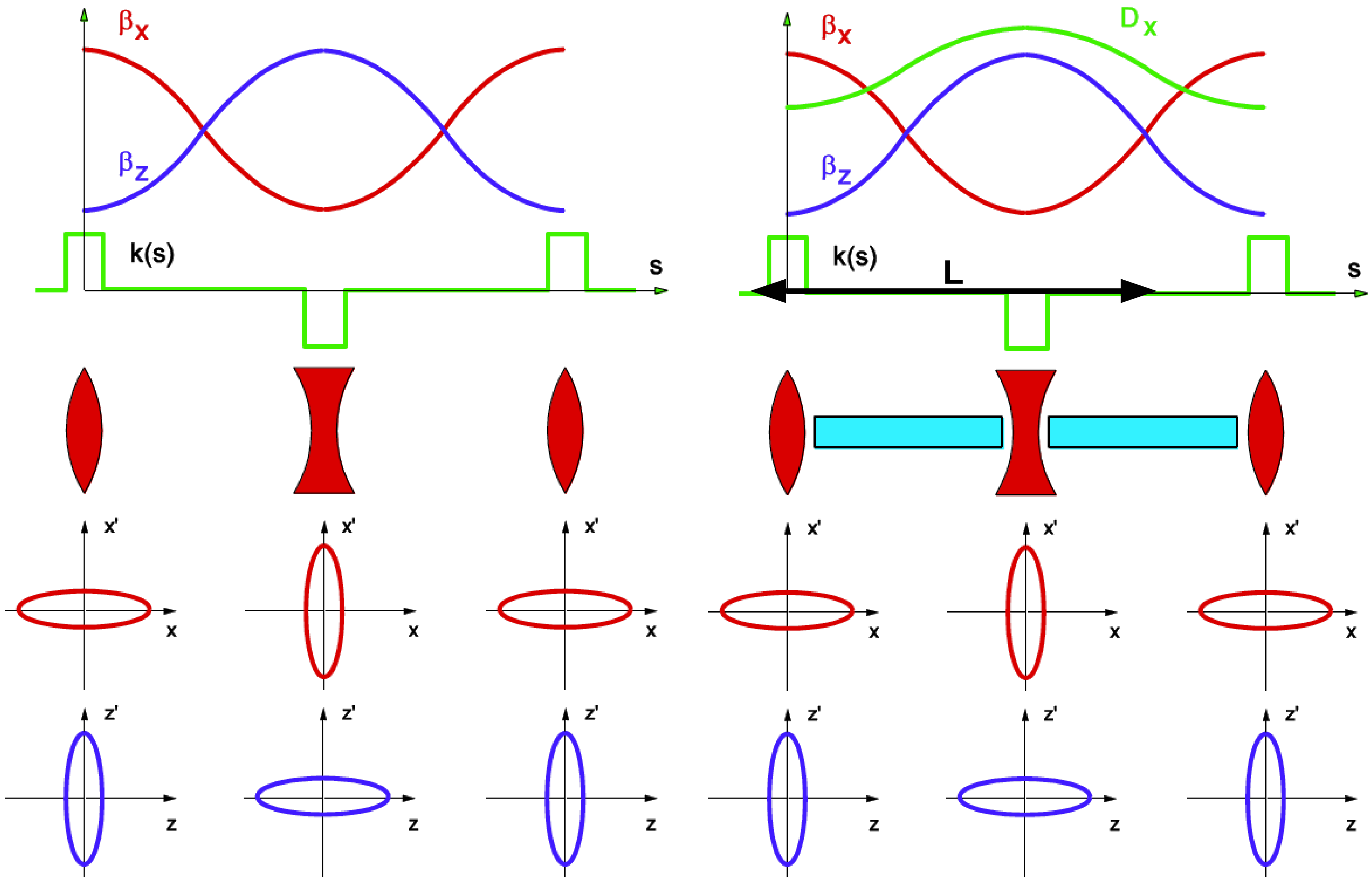
$$\boxed{\beta(s)=\beta_0-2\alpha_0\cdot L+\gamma_0\cdot L^2}$$

→ equation being important for low-beta insertions

- Stability?

$$trace(M)=1+1=2$$

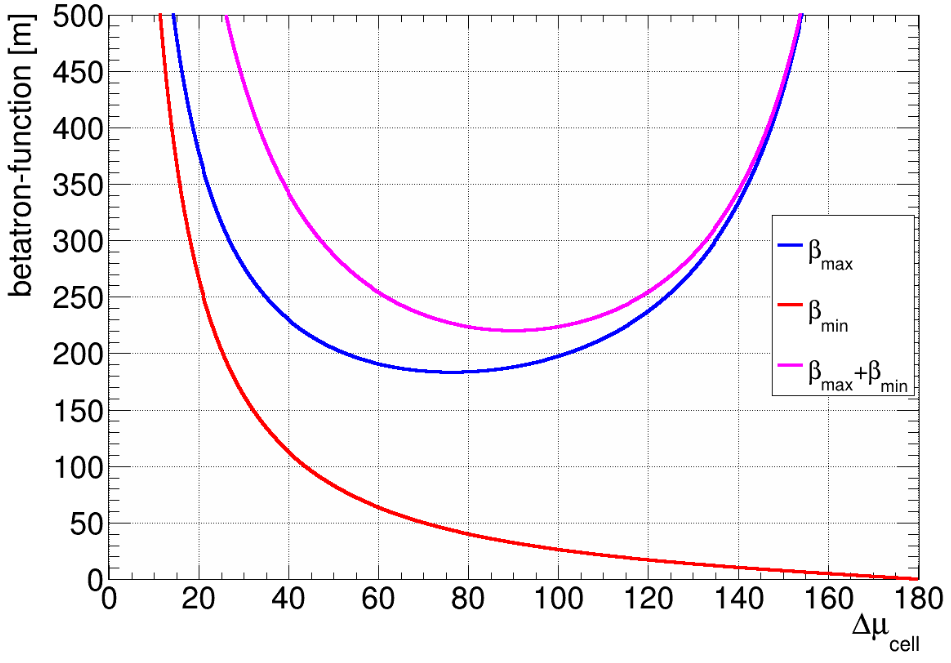
Recap: FoDo Lattice

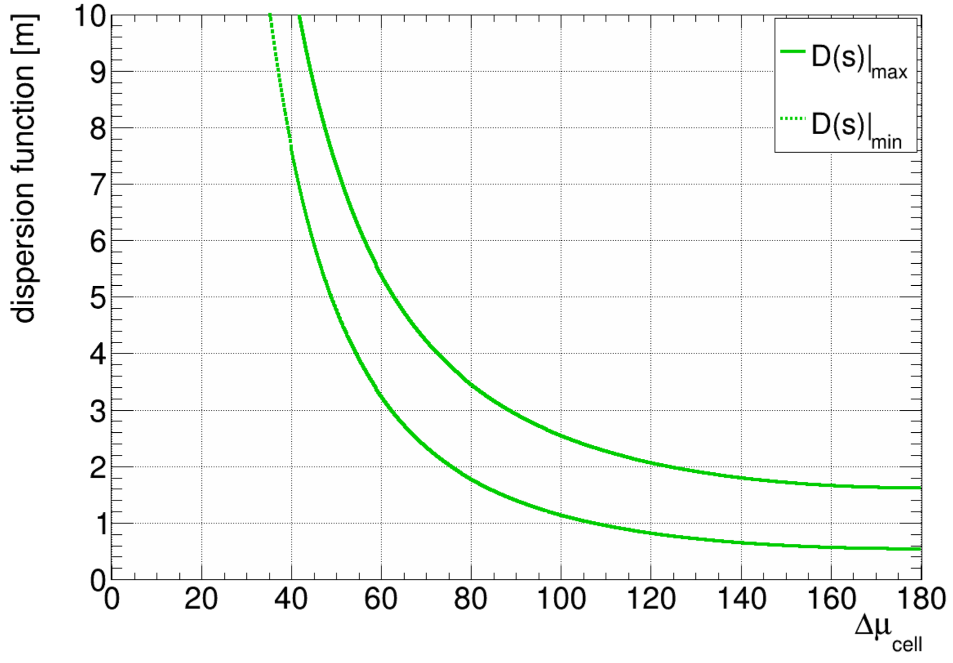


Courtesy L. Rivkin, EPFL & PSI

Recap: FoDo Lattice – Summary I/II

- Beam size optimisation:
  - Hadrons prefer μ=90°
    - minimises: $\epsilon_x \approx \epsilon_y$ & $a = \sqrt{\sigma_x^2 + \sigma_y^2} \sim \beta_x + \beta_y$
  - Leptons prefer μ~137°
    - vertical emittance very small
      → optimise mainly β_x & $D_x|_{max}$
- Dispersion minimisation





Recap: FoDo Lattice – Summary II/II

- FoDo cell transfer matrix (→ tutorial)

$$M_{FoDo}=\begin{pmatrix} 1-\frac{L^2}{2f^2} & L\left(1+\frac{L}{2f}\right) \\ \left(\frac{L^2}{2f^3}-\frac{L}{f^2}\right) & 1-\frac{L^2}{2f^2} \end{pmatrix}$$

- Phase advance per cell (→ tutorial)
  - N.B. also correct for non-FoDo cells

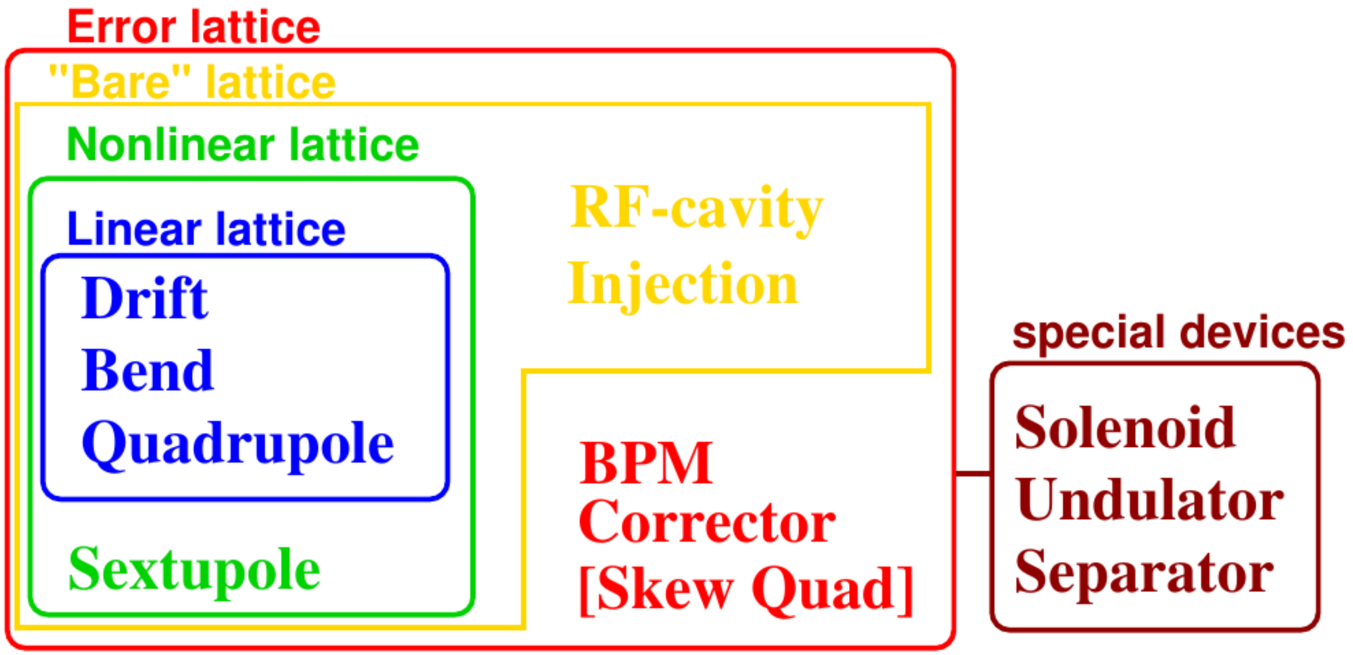
$$\boxed{\cos\mu_{cell}=\frac{1}{2}\,trace(M)}$$

$$\boxed{\begin{aligned} f &= \pm\frac{L}{4\sin\frac{\mu}{2}}=(k\,l_q)^{-1} \\ \beta^{\pm} &= \frac{L\left(1\pm\sin\frac{\mu}{2}\right)}{\sin\frac{\mu}{2}} \\ \alpha^{\pm} &= \frac{\mp 1-\sin\frac{\mu}{2}}{\cos\frac{\mu}{2}} \\ D^{\pm} &= \frac{L\varphi\left(1\pm\frac{1}{2}\sin\frac{\mu}{2}\right)}{4\sin^2\frac{\mu}{2}} \\ \xi_{FODO} &= -\frac{1}{\pi}\tan\frac{\mu}{2} \end{aligned}}$$

- Equations guide 1st-order cell design
  → input for non-linear numerical optimisations (tutorial)

Lattice Design …

- … how to build a storage ring, transfer line or linear accelerator



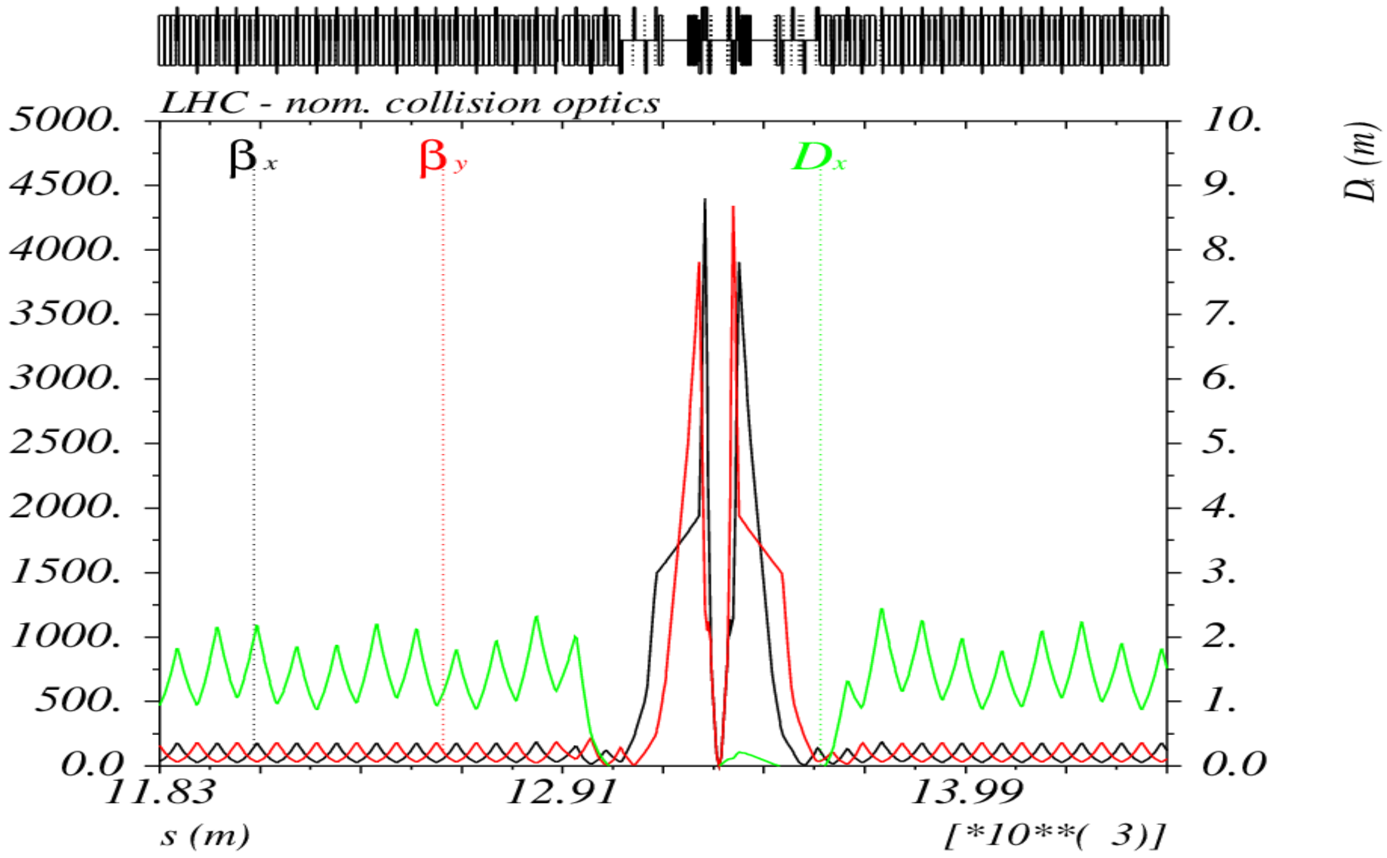
- Fundamental question of lattice design:
  – How many magnets, types? Strengths? Positions

- We tackle today only linear lattice,
  … and maybe scratch the tip of the non-linear lattice iceberg

Lattice Design Constraints

- **Main Machine Parameters**:
  - leptons (e^+, e^-) vs. hadrons (protons, ions, …)
  - particle energy, beam intensity, is emittance critical?
  - linear vs. circular accelerator
- Besides performance and feasibility, basic requirements are:
  - **Simple**: few different component types, standardization, high lattice symmetry
  - **Robust**: tolerance to errors in alignment and component manufacturing
  - **Fail-safe**: generally conservative, taking technological risks only when it is really necessary to achieve unprecedented performance
  - **Cost-effective**: concerning both installation and operation costs
  - **Reliable**: thoroughly testing in order to enable significant performance predictions
- Of course, all these requirements contradict performance and a careful weighting has to be done.

Lattice Design



- Arc: regular (periodic) magnet structure:
  - bending magnets → defines the energy & circumference of the ring
  - main focusing & tune control, chromaticity correction,
  - multipoles for higher order corrections
- Insertions (straight sections):
  - drift spaces for injection, dispersion suppressors, RF cavities,
  - undulators, low-β insertions (high energy experiments), etc... *if they cannot be avoided*

MAD – Methodical Accelerator Design

- large zoo of codes: AT, BETA, BMAD, COMFORT, COSY-INFINITY, DIMAD, Elegant, LEGO, LIAR, LUCRETIA, MAD-X, MARYLIE, MERLIN, Orbit, PETROS, Placet, PTC, RACETRACK, SAD, Sixtrack, SYNCH, Teapot, TRACY, Transport, TURTLE, UAL, … *<this being probably only 20% of all>*

- From http://cern.ch/mad: "... a project with a long history, aiming to be at the forefront of computational physics in the field of particle accelerator design and simulation."

- Has it's "idiosyncrasies"
  - programmed by/for (experienced) physicist (mix of C,C++, F90 & F77)
  - sometimes tough for beginners: some (un-)documented features, scripting only/no-GUI

- However:
  - … de facto standard for beam dynamics, optics design and optimisation
  - … well tested and benchmarked against many real-world operational machines!
  - … open-source: Linux, Mac and Windows (32 bit & 64 bit)
  - … presently being reviewed and consolidated to permit major improvements.

General Purpose Lattice Programs

- For circular machines, transport lines or linacs
- Calculate optics parameters from machine description
- Compute (match) desired quantities
- Simulate and correct machine imperfections
- Simulate beam dynamics
- Used in this course: MAD-X
  - web-site: http://cern.ch/mad/ – version 5.01.00 (or dev: 5.01.05)
  - Manual: http://cern.ch/madx/madX/doc/usrguide/uguide.html
  - A helpful ‘primer’ by W. Herr: http://cern.ch/madx/doc/madx_primer.pdf
  - Some other tutorials:
    - http://cern.ch/madx/doc/madx_tutorial.pdf
    - http://cern.ch/madx/doc/madx_tut.pdf

MAD-X

- MAD-X is an interpreter (ie. No-GUI) and typically driven by input files
- Recommend to split (in particular) larger projects into
  - 'sequence' files (convention: ending in '.seq')
    - Definition of each machine element
    - Element attributes (ie. length, magnetic strengths, ...)
    - Positions of the elements
  - 'job' script files (convention: ending in '.mad', '.madx' or '.job')
    - Description of the beam(s)
    - Directives (what to do ?)
- Many features of a programming language (loops, if conditions, macros, subroutines ...)

MAD-X Input Language

- Strong resemblance to ”C” language
- All statements are terminated with `;`
- Comment lines start with: `//` or `!`
- Not case sensitive
- Arithmetic expressions, basic functions (`exp`, `log`, `sin`, `cosh`, …) & predefined constants (clight, e, pi, m_p , m_e …)
- Deferred expressions (:= instead of =) using variables:
  - `ANGLE = 2*PI/NBEND;`
  - `AIP = ATAN(SX1/SX2);`
- N.B. The assignment symbols = and := have a very different behaviour (here random number generator)!
  - `DX = GAUSS()*1.5E-3;`
    The value is computed once and kept in `DX`
  - `DX := GAUSS()*1.5E-3;`
    The value is recomputed every time `DX` is used

Be aware of difference! Common source for mistakes when doing fitting or error assignments!

MAD Input Statements

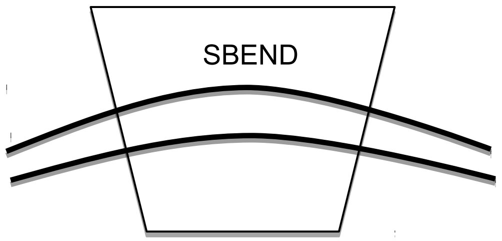
- Typical assignments
  1. Properties of machine elements
  2. Definition of element strength
  3. Set up of the lattice – physical position of magnets, cavities, etc.
  4. Definition of beam properties (particle type, energy, emittance ...)
  5. Assignment of errors and imperfections

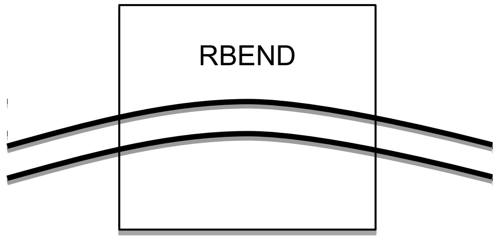
- Typical actions and directives
  - Compute lattice functions
  - Correct machines

1. – Definition of Machine Elements

- All machine elements have to be described
- One can describe a `CLASS` i.e. elements with the same attributes
- Keywords used to define the type of an element. General format:
  - `name: keyword, attributes;`
- Can define single *element* or *class* of elements and give it a name

- Some examples:
  - Dipole (bending) magnet class:
    `MB: SBEND, L=10.0, ANGLE = 0.0145444;`
  - Quadrupole magnet class:
    `MQ: QUADRUPOLE, L=3.3, K1 = 1.23E-02;`
  - QUAD01 is an instance of the class MQ:
    `QUAD01: MQ;`





2. – Thick vs. Thin Elements

- Thick elements: so far all examples were thick elements (or: lenses)
  - Specify length and strength separately (except dipoles !)
    - More precise, path lengths and fringe fields correct
    - Not symplectic in tracking (i.e. phase space not preserved)
    - May need symplectic integration
    - *N.B. supplied tutorial example: '`theonering.seq`'*

- Thin elements: specified as elements of zero length
  - Specify field integral, e.g.: $k_0 \cdot L$, $k_1 \cdot L$, $k_2 \cdot L$, …
    - Easy to use
    - Uses (amplitude dependent) kicks → always symplectic
    - Used for tracking
    - Path lengths not correctly described
    - Fringe fields not correctly described
    - Maybe problematic for small machines
    - *N.B. supplied tutorial example: '`theonering_thin.seq`'*

2. – Magnet Strength I/II – Thick Lenses

- Dipole Bending Magnet

$$k_0 = \frac{q}{p_0} \cdot B_y \quad = \frac{1}{\rho} = \frac{\text{ANGLE}}{\text{L}} \quad [\text{T}]$$

  – *DIP01*: SBEND, L=10.0, ANGLE=angle, K0 = k_0; or
  – *DIP02*: MBL; ! (instances of the class MBL)

- Quadrupole Magnet:

$$k_1 = \frac{q}{p_0} \cdot \frac{\partial B_y}{\partial y} = \frac{1}{L \cdot f} \quad [\text{T}/m]$$

  – *MQA*: QUADRUPOLE, L=3.3, K1 := k_1;

- Sextupole and Octupole Magnet

$$k_2 = \frac{q}{p_0} \cdot \frac{\partial^2 B_y}{\partial y^2} \quad [\text{T}/m^2] \qquad\qquad k_3 = \frac{q}{p_0} \cdot \frac{\partial^3 B_y}{\partial y^3} \quad [\text{T}/m^3]$$

  – *MSF*: SEXTUPOLE, L=1.1, K2 := k_2;
  – *MOF*: OCTUPOLE, L=1.1, K3 := k_3;

2. – Magnet Strength I/II – Thin Lenses

- Multipole: general (thin) element of zero length, any order possible:
  `multip: MULTIPOLE, knl := {kn0 L, kn1 L, kn2 L, kn3 L, ....};`
  $\rightarrow$ `knl` $= k_n \cdot L$ (normal components of nth order)

- Very simple to use:
  - `MB: MULTIPOLE, KNL = {angle,0,0,....};`
    is equivalent to definition of dipole magnet
  - `MQF: MULTIPOLE, knl := {0,`k_1L`,0,0,....};`
    is equivalent to definition of quadrupole ($k_1 \cdot L = \int \frac{1}{\rho/c} \frac{\partial B_y}{\partial x} \cdot dl$)
  - `MSF: MULTIPOLE, knl := {0,0,`k_2L`,0,0,....};`
    is equivalent to definition of sextupole ($k_2 \cdot L = \int \frac{1}{\rho/c} \frac{\partial^2 B_y}{\partial x^2} \cdot dl$)

- For the exercises choose 'thick' and 'thin' lenses only as a fall-back

3. – Definition of Sequence (the Lattice)

- The lattice *name* is defined by a `SEQUENCE` of elements.
  - position is defined at element `CENTRE`, `EXIT` or `ENTRY`
  - both absolute and/or relative position possible

- Example:

```
TheOneRing: SEQUENCE, REFER=CENTRE, L=6912;
...
...
here specify position of all elements ...
...
...
ENDSEQUENCE;
```

MAD-X Input Example – Sequence File I/II

```
// demo sequence file -- some resemblance with CERN-SPS lattice
circum          = 6912.0;       // define the total length
ncell           = 108;          // define number of cells
lcell           = circum/ncell; // cell length

// [..]

// some generic field strength
angleMB     = 2.0*pi/(2*ncell); // fixed, we want a circular accelerator
KQF         = +4.36588E-02;
KQD         = -4.36952E-02;
KSF         = +0.02;
KSD         = -0.04;

// machine element definition
MB:  SBEND, l=lengthMB, angle= angleMB;
MQF: QUADRUPOLE, L=1.0, K1 := KQF; // equivalent to: 'MQF: multipole, knl := {0.0, KQF};'
MQD: QUADRUPOLE, L=1.0, K1 := KQD; // equivalent to: 'MQD: multipole, knl := {0.0, KQD};'
MSF: MULTIPOLE, knl := {0.0, 0.0, KSF};
MSD: MULTIPOLE, knl := {0.0, 0.0, KSD};
BPM: MONITOR, L=0.1;

// [..]
```

MAD-X Input Example – Sequence File II/II

```
// sequence declaration;
TheOneRing: sequence, REFER=centre, l=circum;
  n = 0;
  while (n < ncell) {
    cell_start = n*lcell;
    cell_mid   = cell_start+0.5*lcell;
    cell_end   = cell_start+lcell;
    BPM: BPM,      at = cell_start +0.5*lengthBPM;
    MQF: MQF,      at = cell_start +1.0*lengthBPM +0.5*lengthQuad;
    MB:  MB,       at = cell_start +1.0*lengthBPM +1.0*lengthQuad +0.5*lengthMB;

    BPM: BPM,      at = cell_start +1.5*lengthBPM +1.0*lengthQuad +1.0*lengthMB;
    MQD: MQD,      at = cell_start +2.0*lengthBPM +1.5*lengthQuad +1.0*lengthMB;
    MB:  MB,       at = cell_start +2.0*lengthBPM +2.0*lengthQuad +1.5*lengthMB;
    n = n + 1;
  }
Endsequence;
// sequence declaration – done
```

N.B. being naughty/lazy here – using loops.
Professional context would prefer/need to define element on a one-by-one basis.
However: for the tutorial this is OK!

- Positions can be defined in loops:
  - Loop over number of cells (*ncell*)
  - use the above as a guide for the exercises!

Inserting Sequences

- Sequences can be defined and used like (new) elements:

```
FoDo: SEQUENCE, refer=entry, l=lcell; // FoDo is now a CLASS
     qfsps: qfsps, at=0.0;
     mbsps: mbsps, at=0.25*lcell;
     qdsps: qdsps, at=0.50*lcell;
     mbsps: mbsps, at=0.75*lcell;
ENDSEQUENCE;
```

- Define ring as multiple sub-sequences chained together

```
TheOneRing: SEQUENCE, refer=centre, l=ncell*lcell;
     n = 0;
     while (n < ncell) {
     FoDo, at=n*lcell;
     n = n + 1;
     }
ENDSEQUENCE;
```

- This is particularly useful when interleaving regular FoDo cells with special insertions (i.e. dispersion suppressors, beta*-insertions, etc.)
  - Note bug for prior V5.01.03: http://cern.ch/madx/madX/doc/usrguide/Introduction/sequence_bug.html

MAD Directives

- Typical job sequence
  1. define the input sequence
     - `CALL, file “<your_definition.seq>”  // N.B. can contain several sequences`
     - `USE, SEQUENCE = TheOneRing; //    need to be done once`
  2. define the beam type
     - Need to now particle type, energy, emittance, number of particles, intensity, …
     - `BEAM, PARTICLE=proton, NPART=1.1E11, ENERGY=450,.......;`
  3. initiate computations (Twiss calculation, error assignment, tracking etc.)
     - `TWISS; // or:`
     - `TWISS, file=output; // or:`
     - `TWISS, file=output, sequence=TheOneRing;`
  4. Output results (tables, plotting)
     - `plot, haxis=s, vaxis1=betx, bety, vaxis2=dx;`
  5. Match desired parameters
  6. Re-plot, re-match, ...

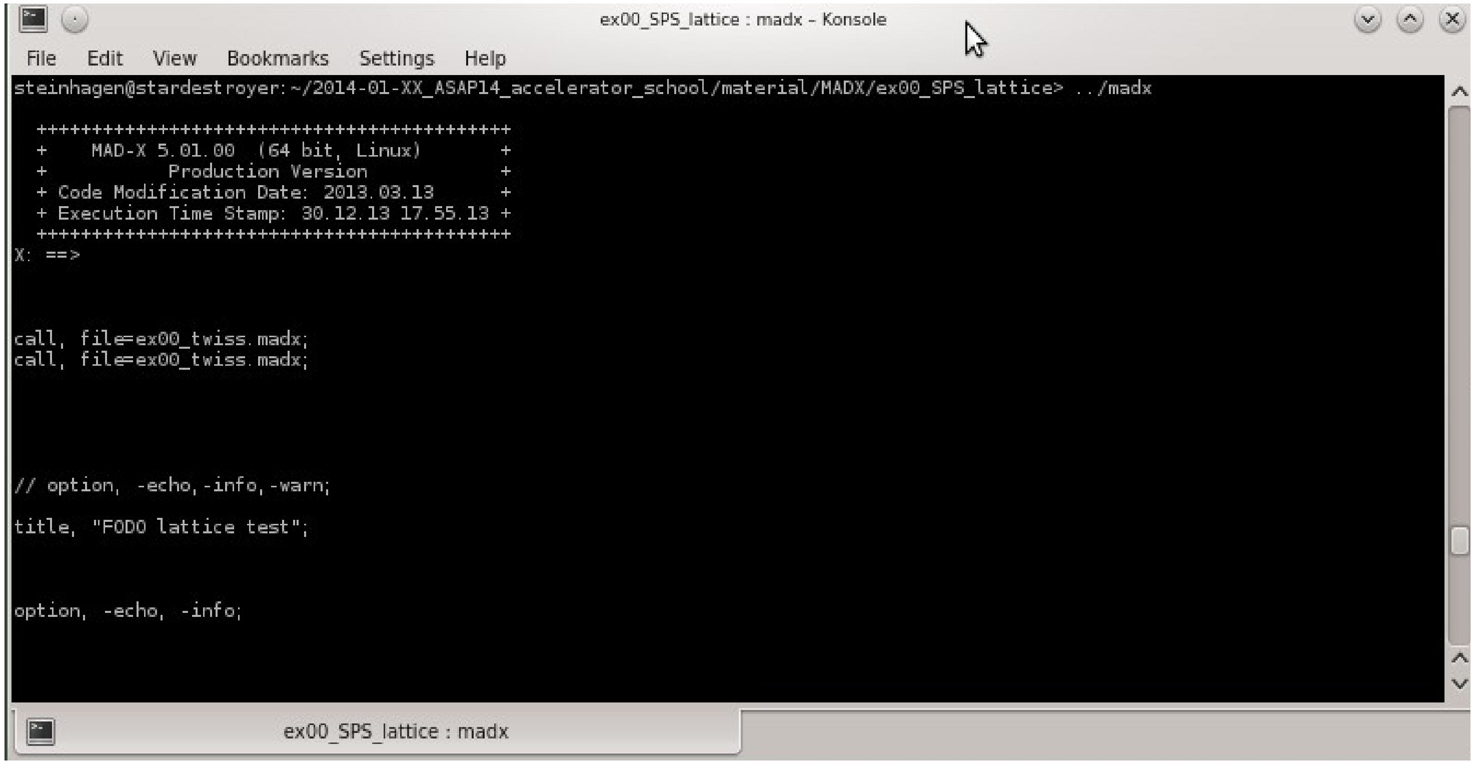
- Beware of default values/options of some commands!
  - If unsure → read the MAD-X manual

MAD-X Input Example – Job File

```
title, "FODO lattice test"; // a nice title/documentation never harms

// define accelerator sequence (lattice)
option, -echo, -info;
 call,file="theonering.seq"; // – ex. script containing sequence and strength definitions
option, info,echo,warn;

// define beam parameters and use specified sequence
BEAM, PARTICLE=proton, NPART=1.1E11, ENERGY=450, EXN=1e-6, EYN=1e-6, SIGT=0.3,SIGE=1e-3;
USE, SEQUENCE=TheOneRing;

// match tune
call,file="match_tune.job"; // – external script

// compute and twiss functions for all elements and write to file 'test_output.twiss'
select, flag=twiss, class=full, column=name,s,keyword,betx,bety,mux,muy,alfx,alfy,dx,dy;
twiss, file=test_output.twiss;

// plot beta-functions to file 'beta_functions01.eps'
setplot, post=2; // 1: 'ps', 2 for 'eps' files
plot, haxis=s, vaxis1=betx, bety, vaxis2=dx, file=beta_functions, noversion;

print, text="********* the end  *********";
```

MAD-X Execute Example

- You store everything in a file: ex00_twiss.madx
  - Must be plain-text (ie. ASCII, no word, etc)
- Execute either via:
  - Interpreter mode (Windows and Linux):
    - <path where you installed MAD-X>./madx <return>
    - type: call, file=ex00_twiss.madx; <return>



```
steinhagen@stardestroyer:~/2014-01-XX_ASAP14_accelerator_school/material/MADX/ex00_SPS_lattice> ../madx

  +++++++++++++++++++++++++++++++++++++++++++
  +    MAD-X 5.01.00  (64 bit, Linux)       +
  +           Production Version            +
  + Code Modification Date: 2013.03.13      +
  + Execution Time Stamp: 30.12.13 17.55.13 +
  +++++++++++++++++++++++++++++++++++++++++++
X: ==>


call, file=ex00_twiss.madx;
call, file=ex00_twiss.madx;




// option, -echo,-info,-warn;

title, "FODO lattice test";


option, -echo, -info;
```

  - Batch mode (Linux, preferred): >./madx < ex00_twiss.madx; <return>

MAD-X Typical Output I/III

- Twiss summary:

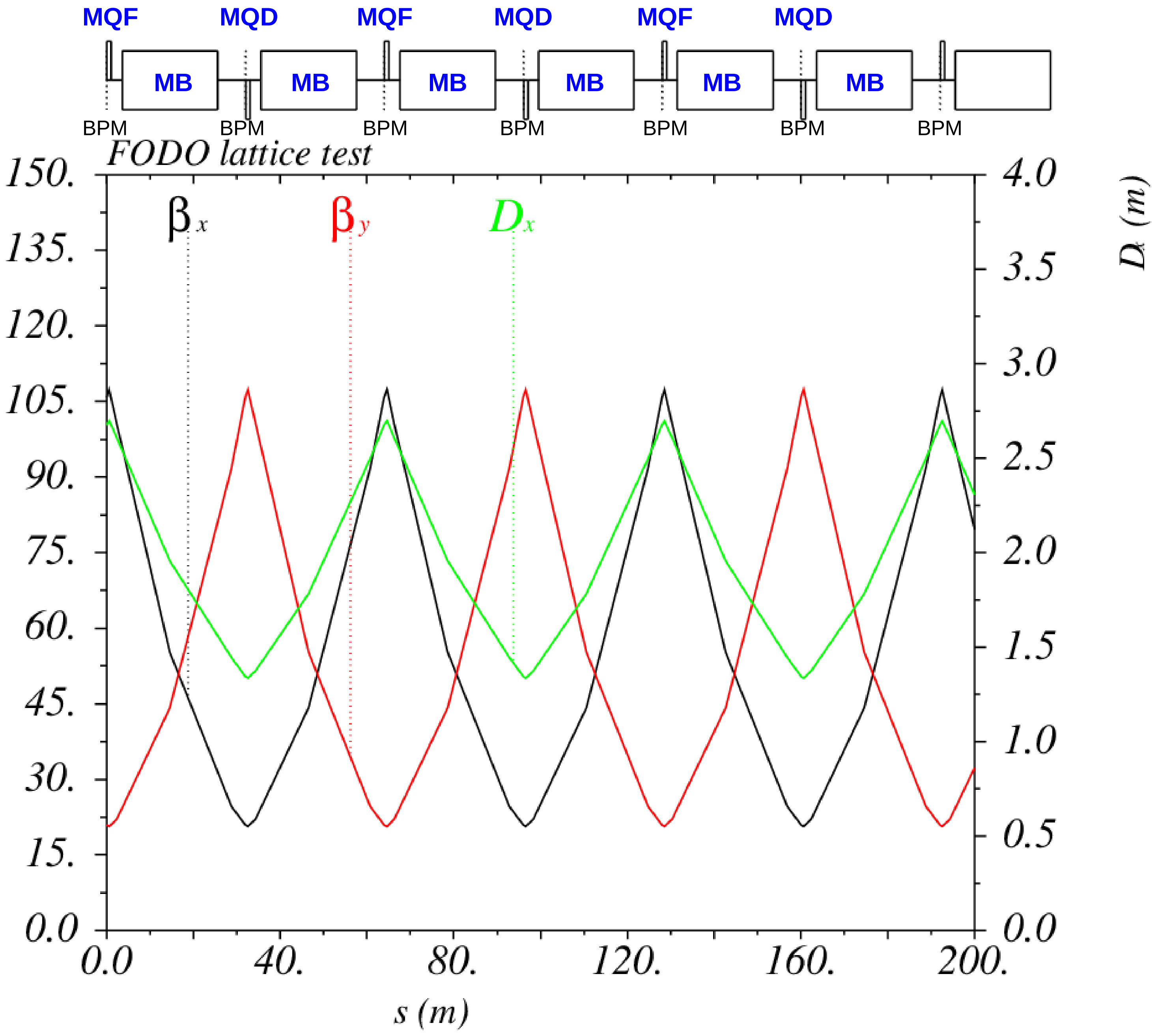
```
 +++++++ table: summm

          length             orbit5               alfa            gammatr
            6912                 -0     0.001680028744        24.39729311
       hor. tune    hor. chromaticity                  max. hor. dispersion
              q1                dq1           betxmax              dxmax
     26.25841152       -31.81887668        106.7519827        2.601681498
                                                                ver. tune
           dxrms             xcomax             xcorms                 q2
     2.122471143                  0                  0        26.26965095
ver. chromaticity max. ver. dispersion max. ver. dispersion
             dq2            betymax              dymax              dyrms
    -33.44567979        106.7708596                  0                  0

          ycomax             ycorms             deltap            synch_1
               0                  0                  0                  0

         synch_2            synch_3            synch_4            synch_5
               0                  0                  0                  0
```

- Variables can be also addressed within MAD, e.g.: `VALUE, q1;`

MAD-X Typical Output II/III – Twiss

```
@ NAME                 %05s "TWISS"
@ TYPE                 %05s "TWISS"
@ SEQUENCE             %10s "THEONERING"
@ PARTICLE             %06s "PROTON"
[..]
[ Summary of ring and beam parameter used for calculating the optics]
[..]
* NAME                                    S KEYWORD                     BETX               BETY               MUX                MUY
$ %s                                    %le %s                           %le                %le               %le                %le
 "BPM"                                29.45 "MONITOR"             21.5464841        100.9213213     0.09881172797       0.1197083946
 "MQD"                                 31.5 "MULTIPOLE"          19.08286421        110.9151687      0.1149295593       0.1227923937
 "MB"                                  45.9 "MULTIPOLE"          39.74722019        55.21935066      0.2024974641       0.1522449588
 "FODO1$END"                             64 "MARKER"             99.3451047        20.06779347      0.2489991767       0.2437391354
 "FODO1$START"                           64 "MARKER"             99.3451047        20.06779347      0.2489991767       0.2437391354
 "BPM"                                64.05 "MONITOR"            99.56165479        20.02446092      0.2490791916       0.2441361079
 "MQF"                                 65.6 "QUADRUPOLE"         105.2629543        19.03641109      0.2514786561       0.2568390811
 "MB"                                  80.5 "SBEND"              49.48609321        48.09814331      0.2843924058       0.3392647504
 "BPM"                                93.45 "MONITOR"             22.5052429         97.3139115      0.3478154241       0.3695733896
 "MQD"                                 95.5 "QUADRUPOLE"         20.37810985         105.970386      0.3631443987       0.3727717442
 "MB"                                 109.9 "SBEND"              42.90558177        55.01642797       0.444570067       0.4028044434
 "FODO1$END"                            128 "MARKER"             104.873968        21.64699305      0.4880510971       0.4906756166
 "FODO1$START"                          128 "MARKER"             104.873968        21.64699305      0.4880510971       0.4906756166
 "BPM"                               128.05 "MONITOR"            105.0966142        21.60439213      0.4881268959       0.4910435937
```

MAD-X Typical Output III/III – Plot



Optical Matching

- Analytical expression are good starting points but often too inaccurate due to some assumptions being made → use MAD directly for matching

- Main applications:
  - Setting global optical parameters (e.g. tune, chromaticity)
  - Setting local optical parameters (e.g. β-function, dispersion ..) → Part 2
  - Correction of imperfections → Beam Diagnostics Tutorial

- Example, approximations for FoDo cells:

$$\boxed{f = \pm\frac{L}{4\sin\frac{\mu}{2}} = (k\,l_q)^{-1} \quad \beta^{\pm} = \frac{L\left(1\pm\sin\frac{\mu}{2}\right)}{\sin\frac{\mu}{2}} \quad \alpha^{\pm} = \frac{\mp 1 - \sin\frac{\mu}{2}}{\cos\frac{\mu}{2}} \quad D^{\pm} = \frac{L\varphi\left(1\pm\frac{1}{2}\sin\frac{\mu}{2}\right)}{4\sin^2\frac{\mu}{2}}}$$

Optical Matching – Example

- Example, match horizontal (Q1) and vertical (Q2) tunes:
  - N.B. alternatively for linacs/transfer lines: mux, muy
  - Vary the quadrupole strengths KQF and KQD
  - quadrupoles must be defined with: `...,` k_1`:=KQF, ...` etc.
  - '#e' denote the end of the line

```
set_Qx = 26.61; // horizontal tune reference
set_Qy = 26.62; // vertical tune reference

match,sequence = TheOneRing;
  Constraint, SEQUENCE = TheOneRing, RANGE=#e, mux=set_Qx, muy=set_Qy;
  !global, Q1=set_Qpx, Q2=set_Qpy; // alternate definition
  vary, name=KQF, step=1.0E-6;
  vary, name=KQD, step=1.0E-6;
  lmdif,calls=5000,tolerance=1.e-21;
endmatch;

value, KQF; value, KQD; // print results
```

What we want!

What is varied!

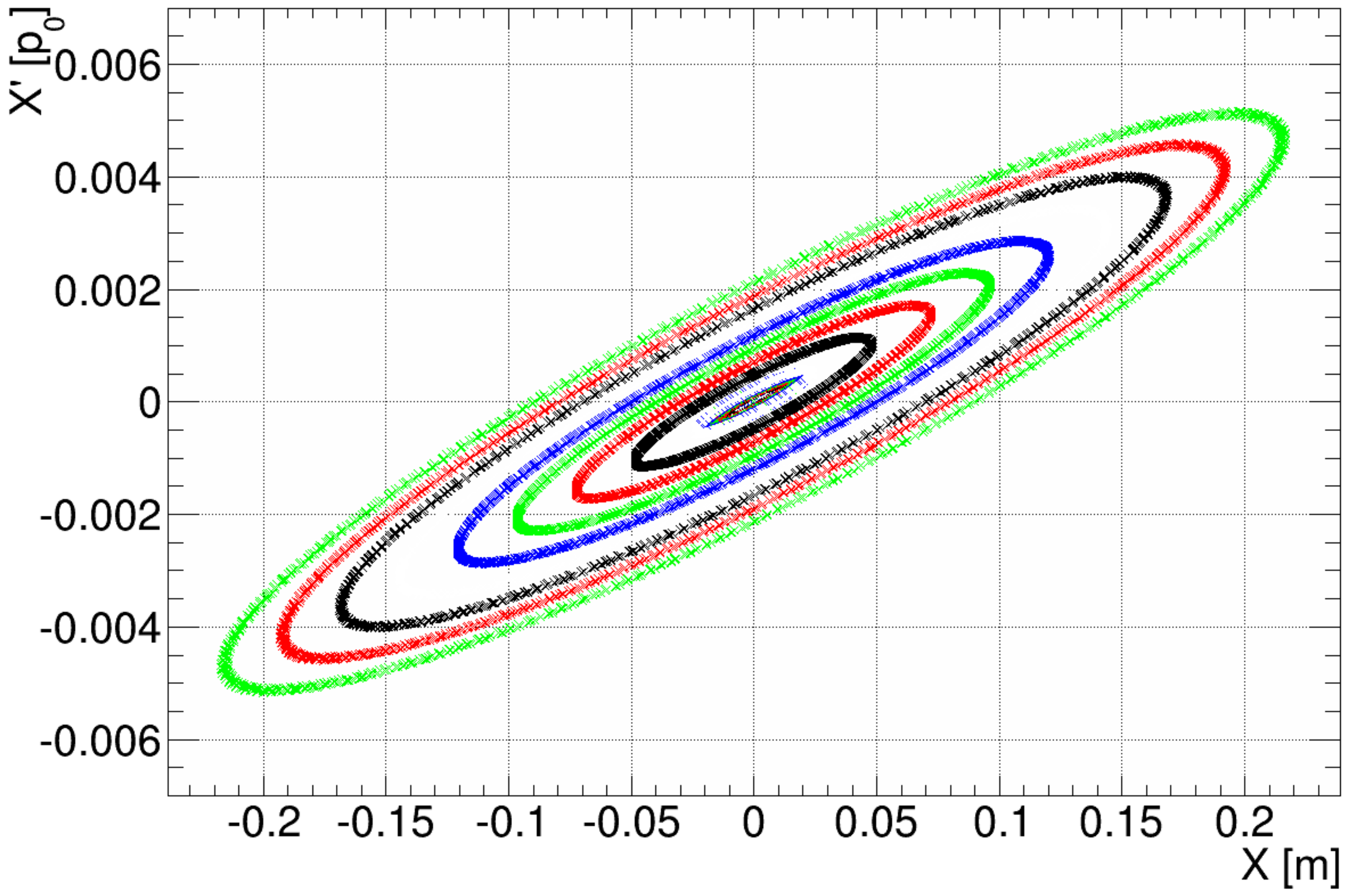
Particle Tracking – PTC

- To track 4 particles for 1024 turns, add:

```
ptc_create_universe;
  ptc_create_layout, model=2, method=6, nst=10,exact;
    ptc_start, x= 2e-2, px=0, y= 2e-2, py=0; phase-space coordinates particle 1
    ptc_start, x= 4e-2, px=0, y= 4e-2, py=0; particle 2
    ptc_start, x= 6e-2, px=0, y= 6e-2, py=0; particle 3
    ptc_start, x= 8e-2, px=0, y= 8e-2, py=0; particle 4

  ptc_track,icase=6,closed_orbit,dump, // icase: defined 4,5,6D tracking
  // radiation, radiation_energy_loss, radiation_quad,//enables synch.-light E-losses
  turns=1024,ffile=1, norm_out, norm_no=1;

  // plot trajectories
  setplot, post=2; // 1: 'ps', 2 for 'eps' files
  title, "FODO phase-space test";
  plot, file="ptc_track", table=trackone, haxis=x, vaxis=px, particle=1,2,3,4,
  style=0, colour=1000, multiple, symbol=1, noversion;
ptc_end;
```

Tracking – Phase-Space-Plot Example

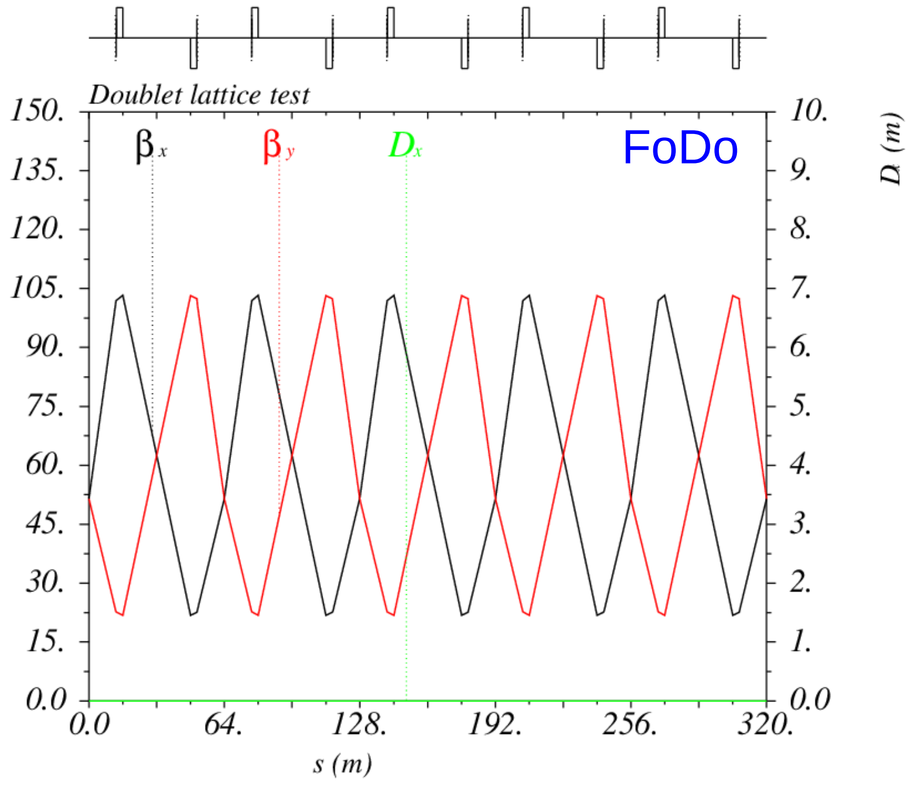
- Particles at different amplitudes/phases but same reference momentum p_0

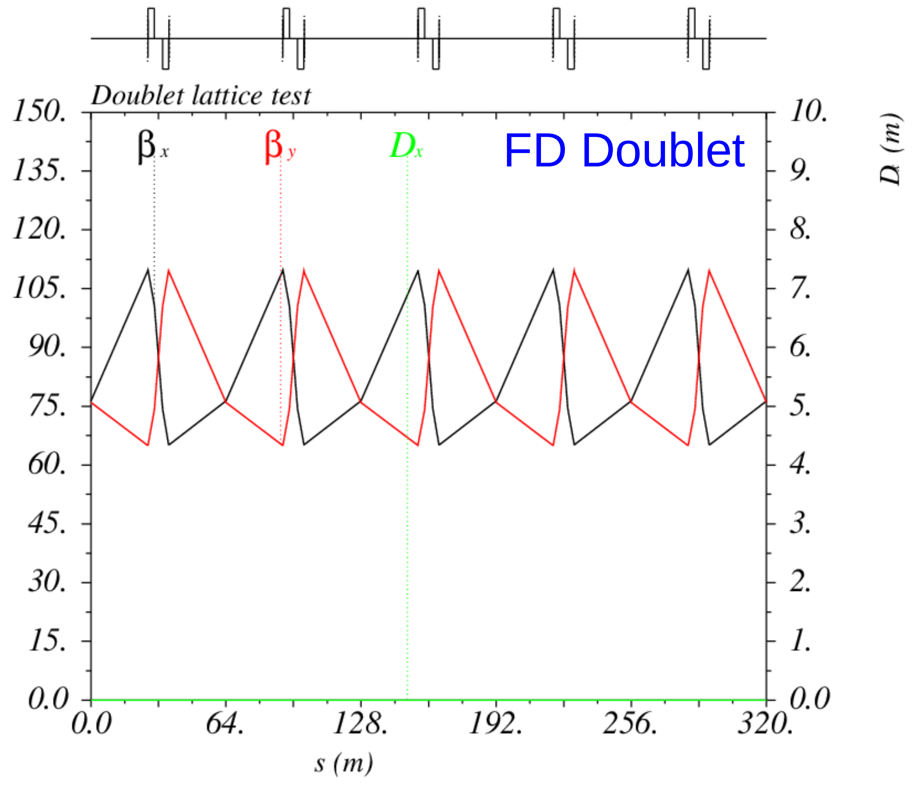


- Beware: 'thintrack' tracks only in 4D → PTC does correct 6D tracking
  - still sufficient for our "transverse optics" tutorial purposes

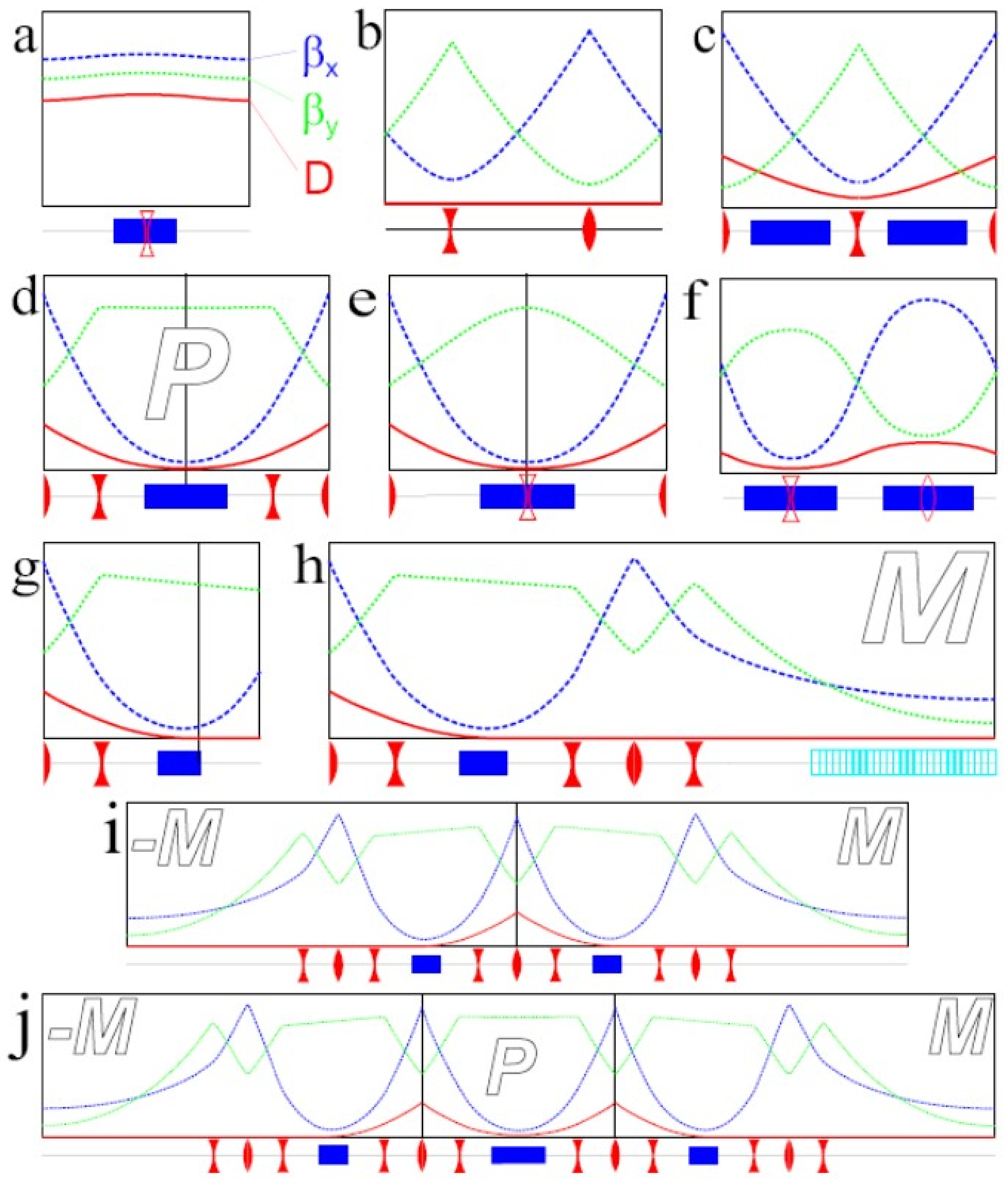
FoDo alternatives → FD Doublet Lattice

- More space between quads
- Stronger quad strengths
- Round beams
- Used e.g. in CTF3 linac





Many Alternatives – Give it a try!



a) Weak focusing (dipole only)

b) FODO line (w/o dipoles)

c) FODO cell

d) Low-emittance cell

e) CF low-emittance cell

f) Low-emittance FODO

g) Dispersion match

h) Periodic dispersion match

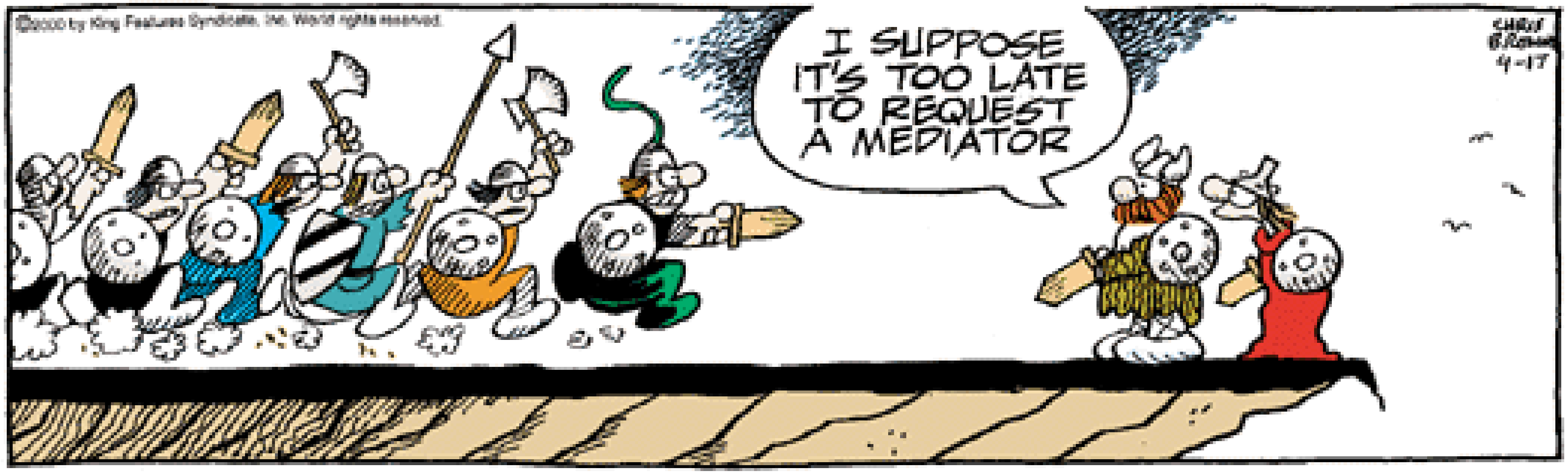
i) Double-bend achromat

j) Triple-bend achromat

k) ...

Very good course on low-emittance lattice design: A.Streun, PSI

That's all – Questions?


©2000 by King Features Syndicate, Inc. World rights reserved.
I SUPPOSE IT'S TOO LATE TO REQUEST A MEDIATOR
CHRIS BROWNE 4-17

Additional Supporting Slides

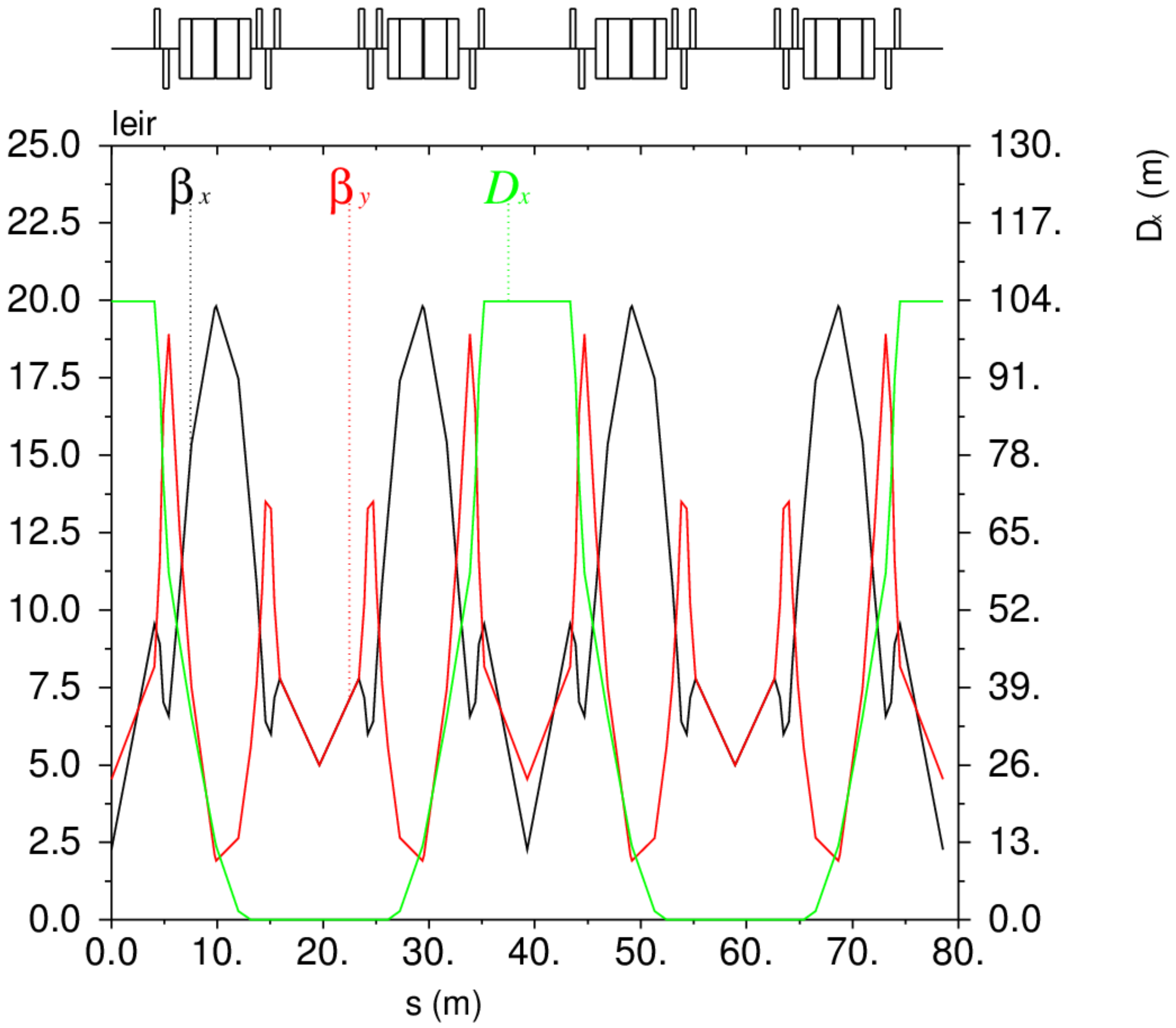
Tracking II/III

- Need more particles? Try:

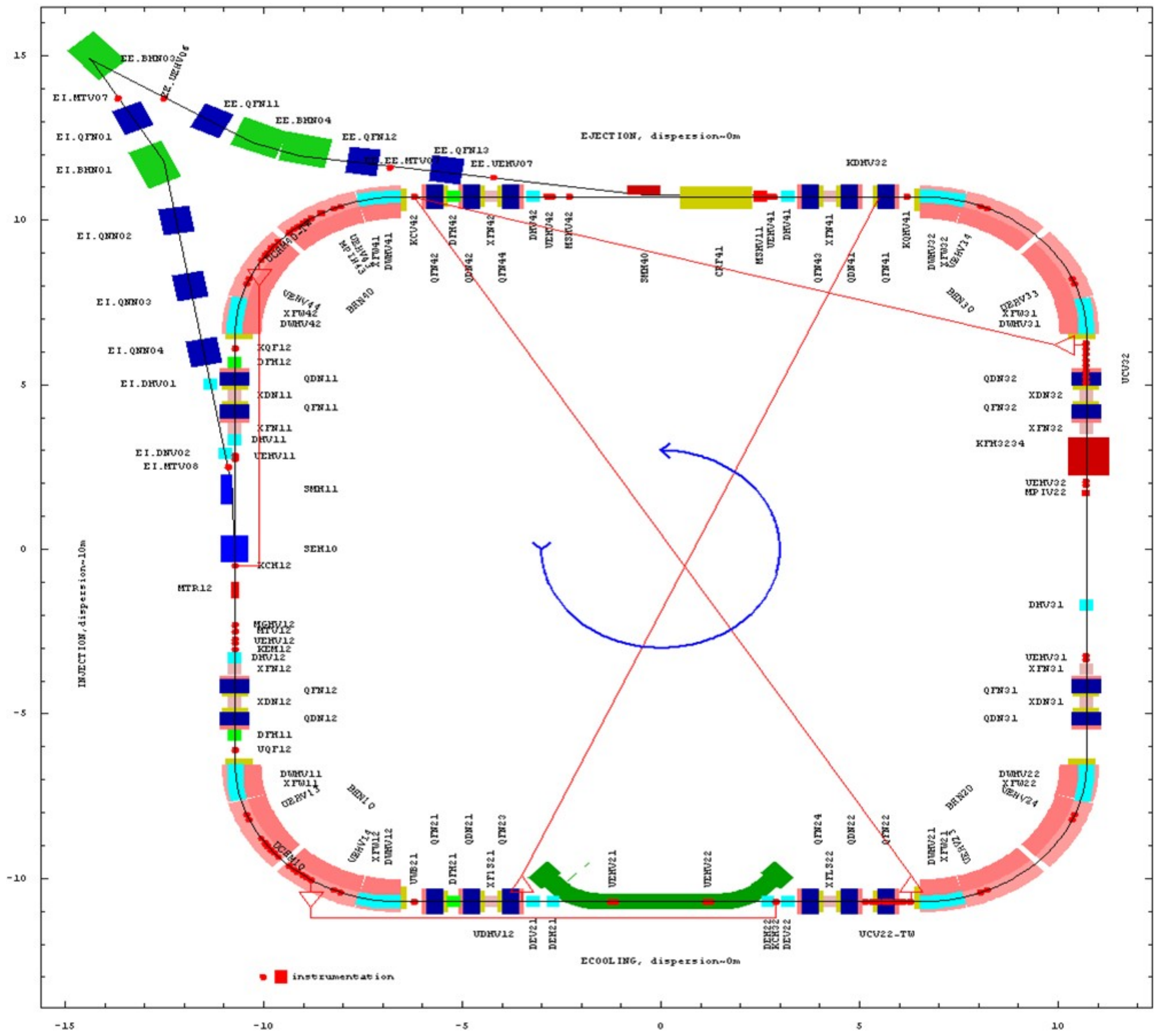
```
ptc_create_universe;
  npart = 0; i=0;
  ptc_create_layout, model=2, method=6, nst=10,exact;
    npart = 0;
    while (npart < 7) {
      i = 1;
      while (i < 10) {
        ptc_start, x= i*1e-8*(10^npart), px=0, y= i*1e-8*(10^npart), py=0;
        i = i + 1;
      }
      npart = npart +1;
    }
  ptc_track,icase=6,closed_orbit,dump, onetable,
  // radiation, radiation_energy_loss, radiation_quad,//enables synch.-light E-losses
  turns=1024,ffile=1, norm_out, norm_no=1;
ptc_end;
```

- Alternate but needs the supplied ROOT 'plot_tracking.C' script to plot tracks for npart>10 particles

What is Lattice Design?

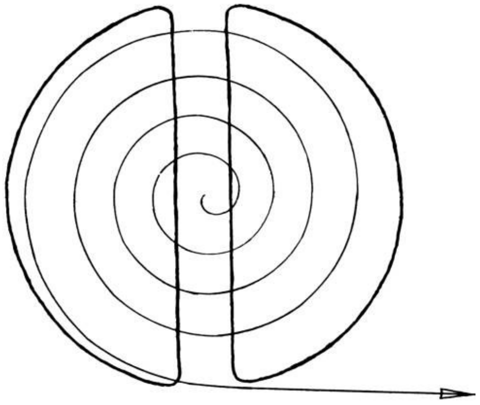


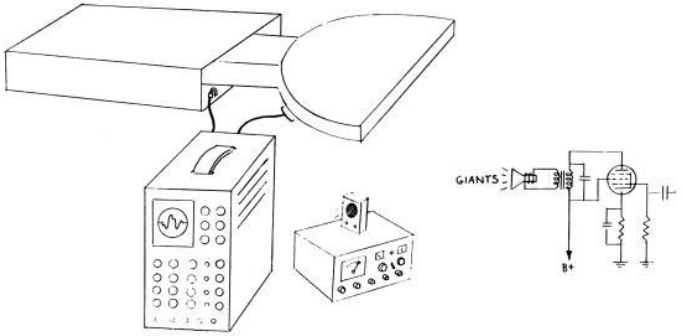
What is Lattice Design?



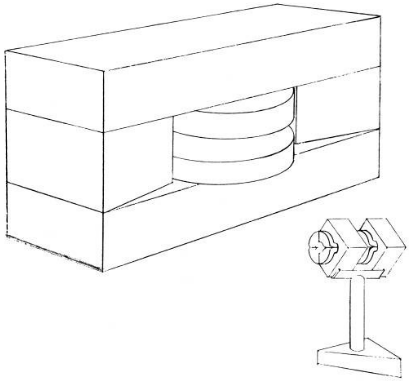
What is Lattice Design?

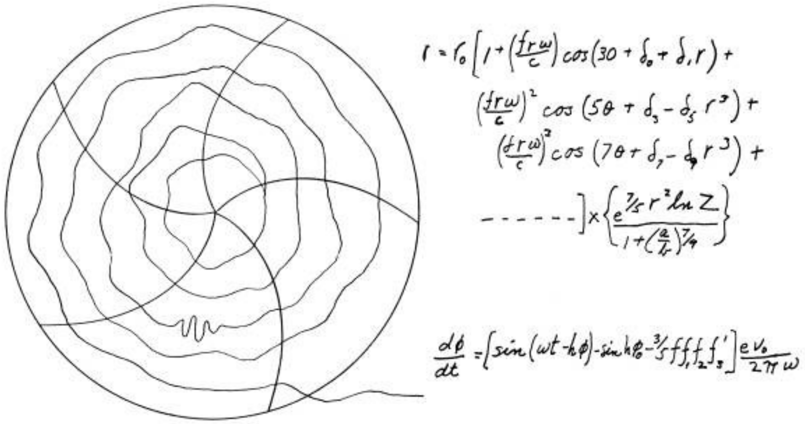
The Accelerator seen by …

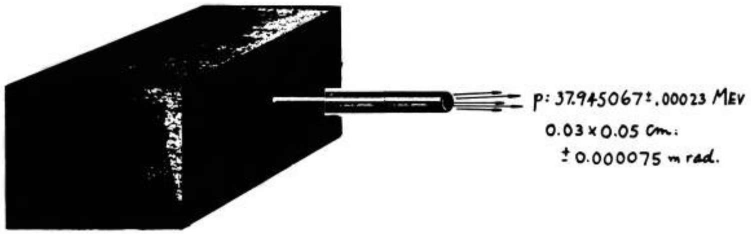

… the inventor



… the electrical engineer

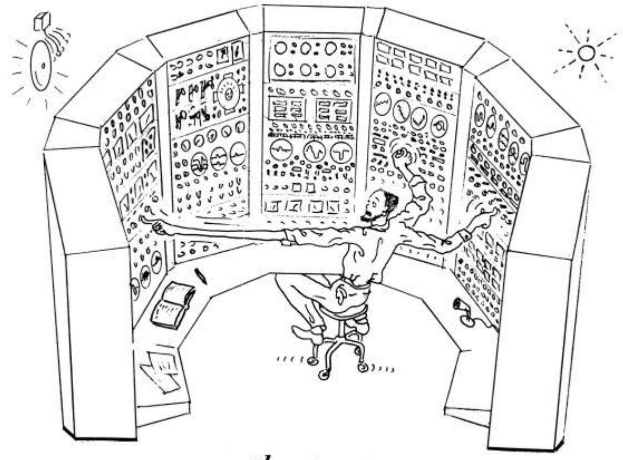

… the mechanical engineer


r = r₀[1+(frω/c) cos(3θ + δ₀ + δ₁r) +
(frω/c)² cos (5θ + δ₃ − δ₅ r³) +
(frω/c)² cos (7θ + δ₇ − δ₉ r³) +
------] × {e^{7/5} r² ln Z / 1+(a/4)^{7/4}}
dφ/dt = [sin (ωt − hφ) − sin hφ₀ − 3/5 f f₁ f₂ f₃′] eV₀/2πω
… the theoretical physicist

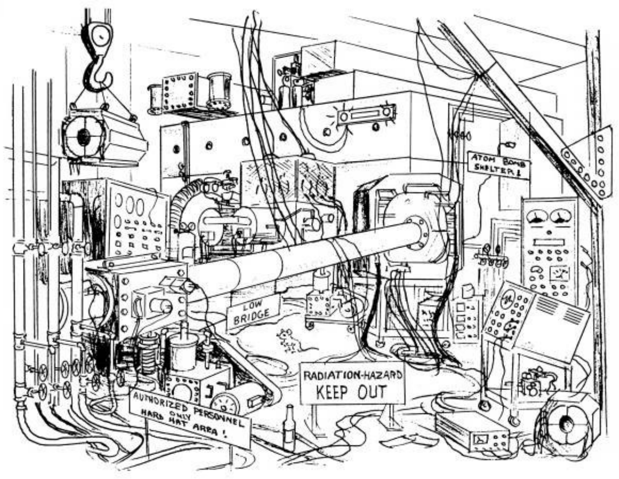


… the experimental physicist

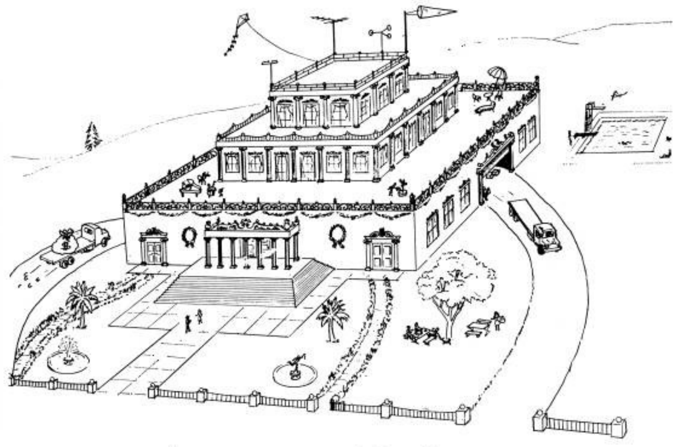
The Accelerator seen by …



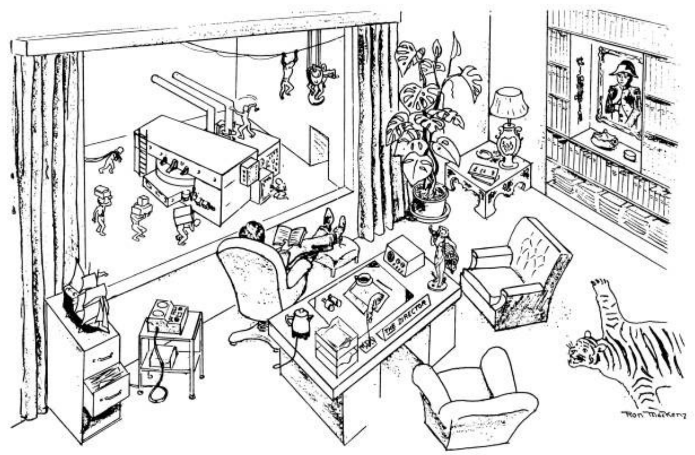
… the operator


ATOM BOMB SHELTER
LOW BRIDGE
RADIATION-HAZARD KEEP OUT
AUTHORIZED PERSONNEL ONLY HARD HAT AREA!

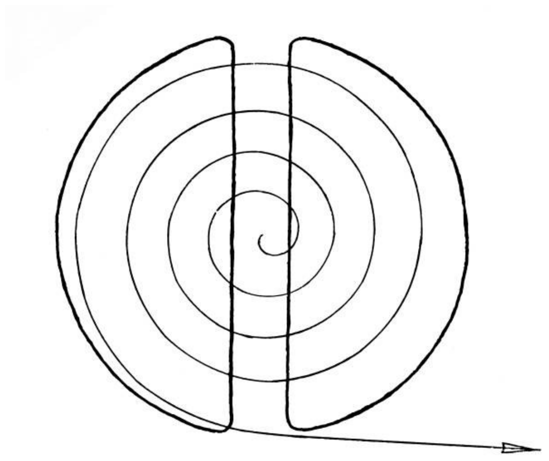
… the visitor



… the governmental funding agency



… the laboratory director



… the student

Cartoons: Dave Judd and Ronn MacKenzie