Hector :

A fast multi-purpose simulator for particle propagation

X. Rouby & J. de Favereau

Contents

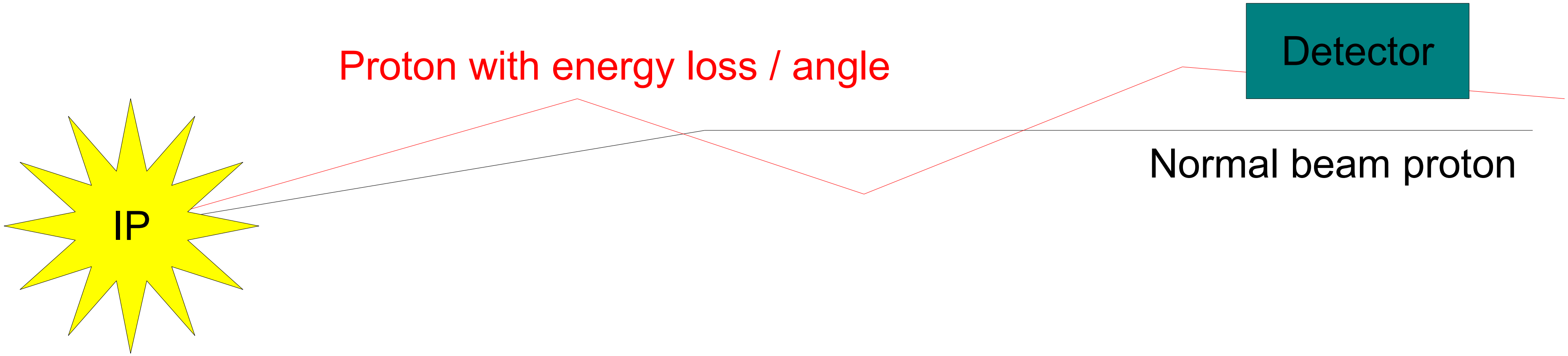
- Introduction : the need for a new tool
- The LHC beamline as an example
- Implementation
- Validation with MAD
- Some Physics
- Prospects

Bonus : Where's Waldo (Où est Charlie) ?

Introduction (I)

- Forward Physics : physics with very forward objects.
- Includes : diffractive physics, photon-related physics.
- Low σ -> need some help to tag !
- How ? By detecting forward objects -> new detectors («Roman Pots») far from IP.
- Such detectors could allow «full» event reconstruction.

Introduction (II)



-> Need for a realistic simulation of particle propagation in the beamline !

Introduction (III)

Existing tools :

- MAD : Beam simulator used by the LHC Machine group. Problem : beam-oriented*, while we need particle-by-particle propagation. Also very hard to adapt to one's needs.
- MARS : Used for very accurate description of interactions with fields and matter. Problem : Too heavy.
- Transport : Was used for UA1. Good, but not easy to adapt for LHC.

→ There's room for a new Simulator !

Introduction (IV)

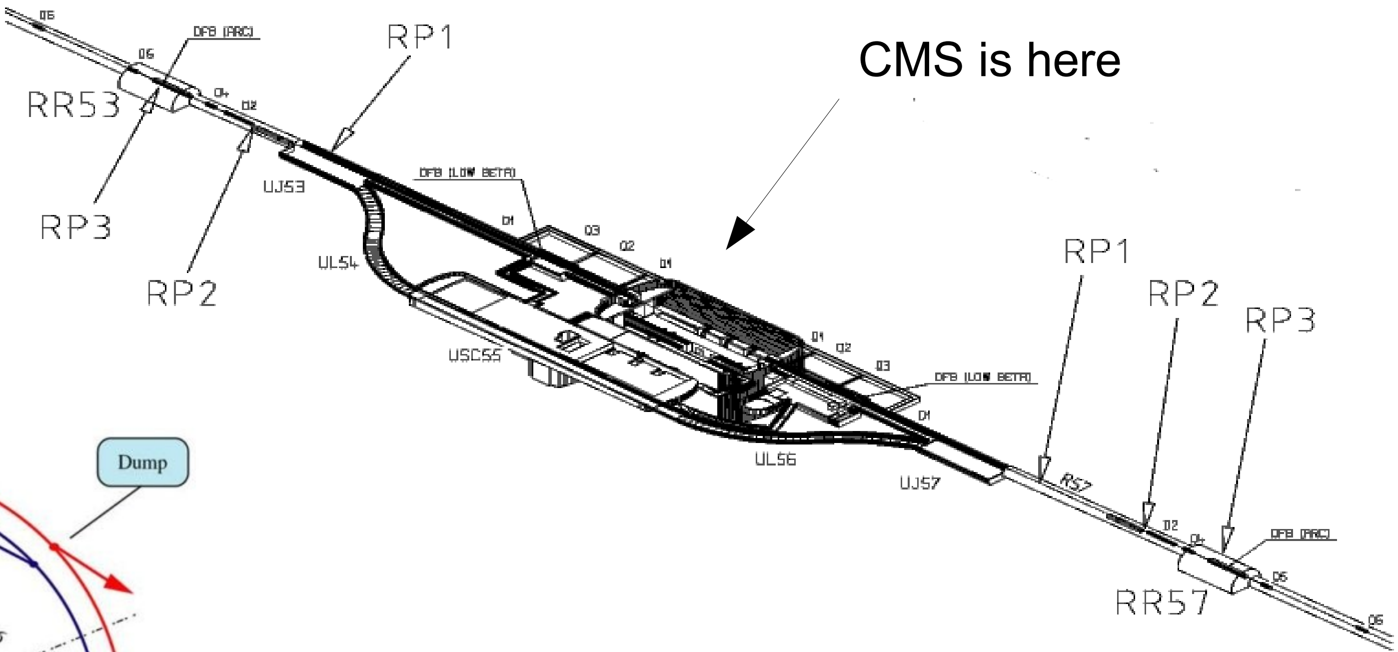
This new program should be :

- Fast
- Lightweight
- LHC-capable
- Particle-oriented

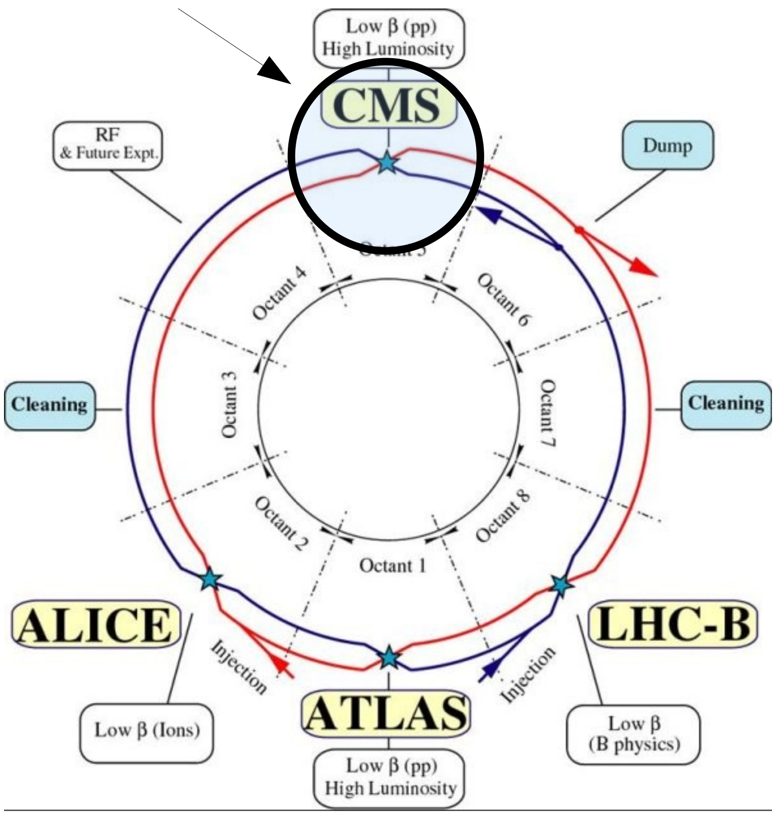
This new program could be :

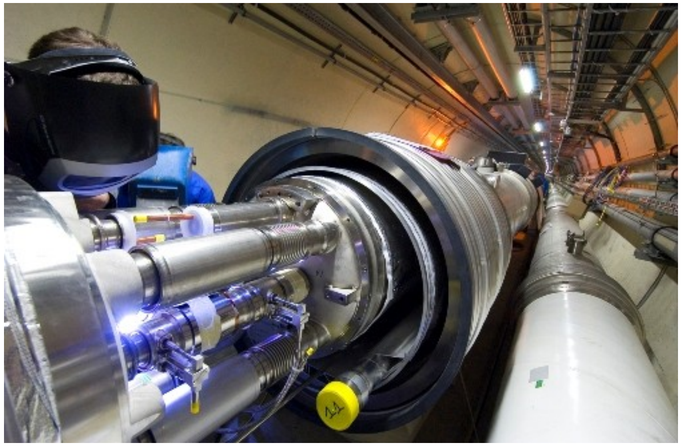
- Object-oriented
- General-purpose
- Easy to use for anyone interested

The LHC beamline



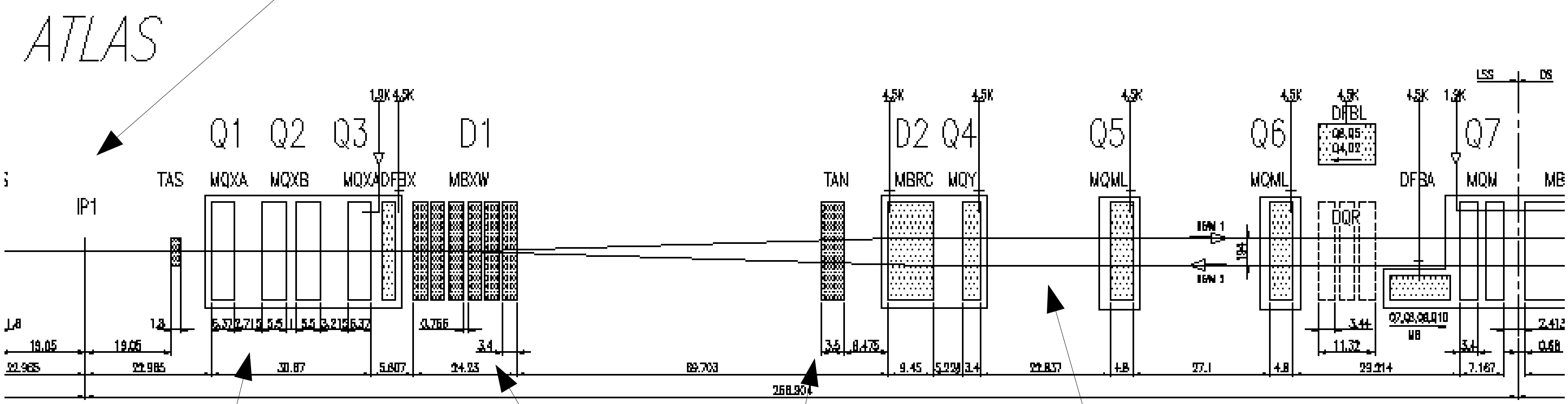
Crossing angle !





The LHC beamline : closer

Interaction point



Quadrupole triplets

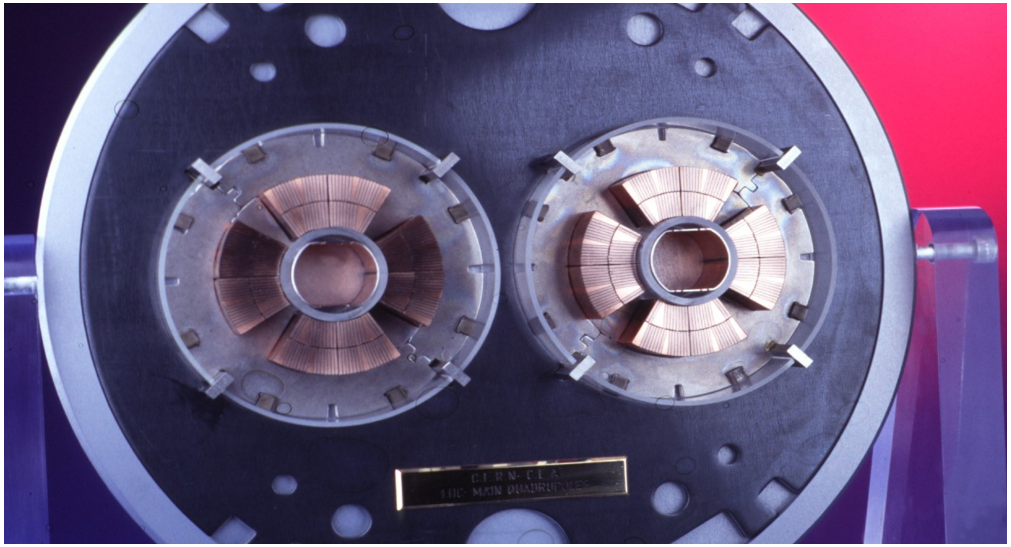
Separation dipole

Collimator

Drift space

Triplet Q + separation D

LHC dipoles & quadrupoles



Implementation (I)

B around its central value :

$$\frac{e}{p}B_y(x) = \frac{e}{p}B_y + \frac{e}{p}\frac{\partial B_y}{\partial x}x + \frac{1}{2}\frac{e}{p}\frac{\partial^2 B_y}{\partial x^2}x^2 + \ldots$$

$k_0 = 1/R$ $k_1 = k$

Taking only dipolar (k_0) and quadrupolar (k_1) terms :

$$x''(s) + \left(\frac{1}{R^2(s)} - k(s)\right)x(s) = \frac{1}{R(s)}\frac{\Delta p}{p}$$
$$y''(s) + k(s)y(s) = 0.$$

The solutions $x(s)$, $x'(s)$, $y(s)$, $y'(s)$ can be expressed (if $\Delta p << p$) as a linear combination of the initial phase-space vector x_0, x'_0, y_0, y'_0

Implementation (II)

Linear behaviour -> matrix representation of the transport :

$$X(s) = X(0)\underbrace{M_1 M_2 ... M_n}_{M_{\text{beamline}}}$$

Where :

X is the phase-space vector of the particle

M_i are the matrices associated to the magnets

Rem : As considered energy losses are not negligible, we introduce an energy dependence of M_i as a correction to linearity

Implementation (III)

Matrix structure :

$$\mathbf{M_{units}} = \begin{pmatrix} \mathcal{A} & \mathcal{A} & 0 & 0 & 0 & 0 \\ \mathcal{A} & \mathcal{A} & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathcal{B} & \mathcal{B} & 0 & 0 \\ 0 & 0 & \mathcal{B} & \mathcal{B} & 0 & 0 \\ \mathcal{D} & \mathcal{D} & 0 & 0 & 1 & 0 \\ 0 & K & 0 & K & 0 & 1 \end{pmatrix}$$

► (de)focusing

► bending

Matrix example : Quadrupole

$$\mathbf{M_{vertical-quadrupole}} = \begin{pmatrix} \cosh(\omega) & \sqrt{k}\sinh(\omega) & 0 & 0 & 0 & 0 \\ (1/\sqrt{k})sinh(\omega) & \cosh(\omega) & 0 & 0 & 0 & 0 \\ 0 & 0 & \cos(\omega) & -\sqrt{k}sin(\omega) & 0 & 0 \\ 0 & 0 & (1/\sqrt{k}) * sin(\omega) & \cos(\omega) & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Implementation (IV)

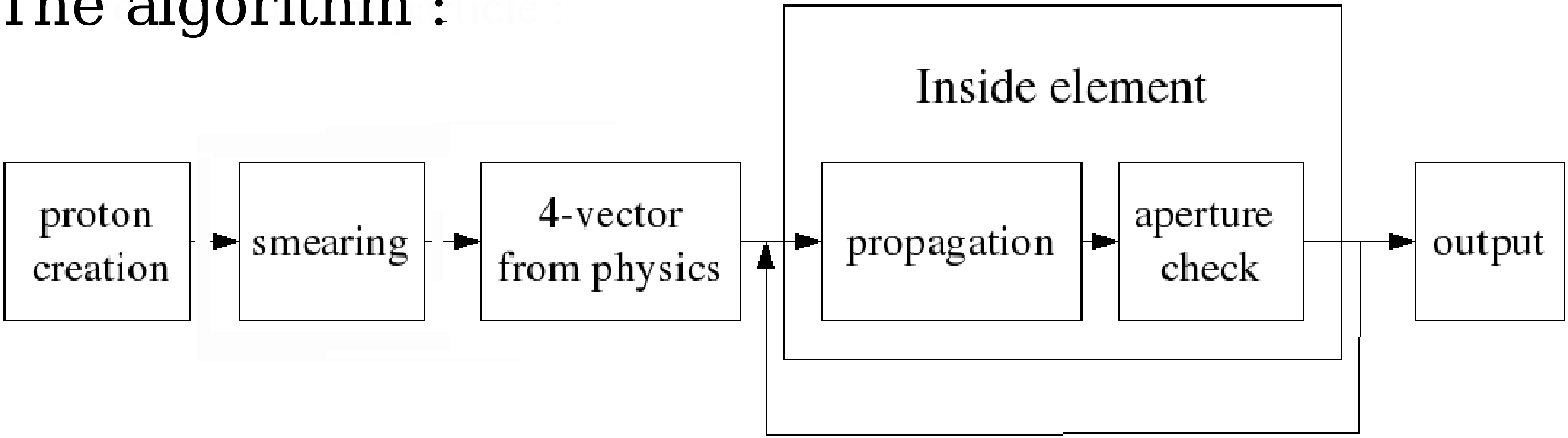
Input Needed :

- k_i
- effective field length
- magnet position
- magnet aperture

All directly provided by the LHC group tables !

Implementation (V)

The algorithm :

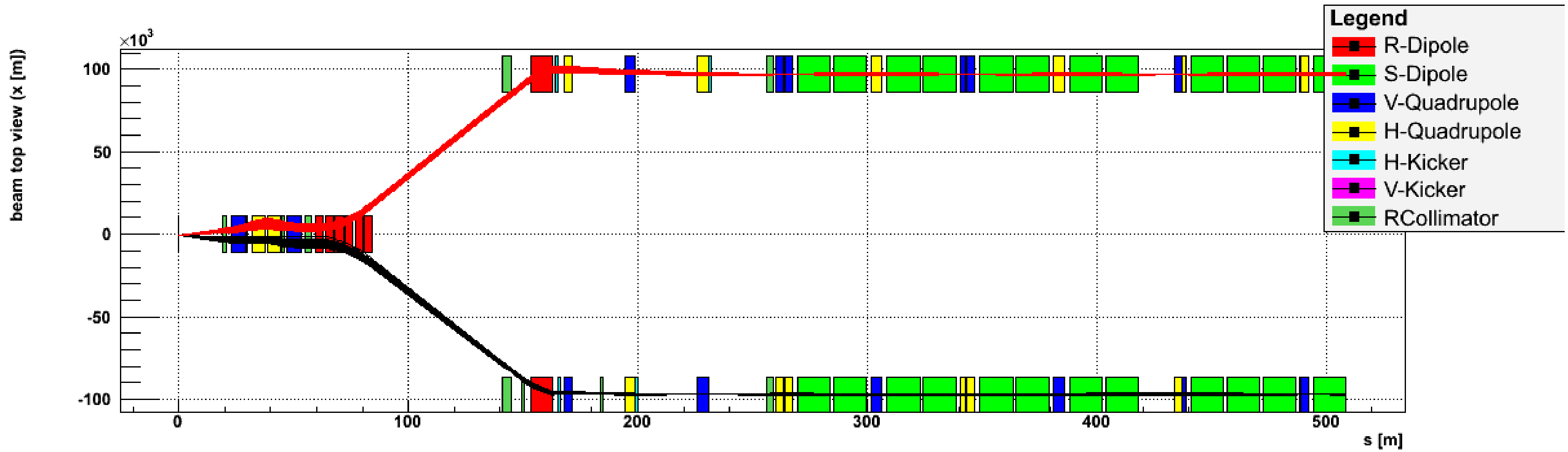


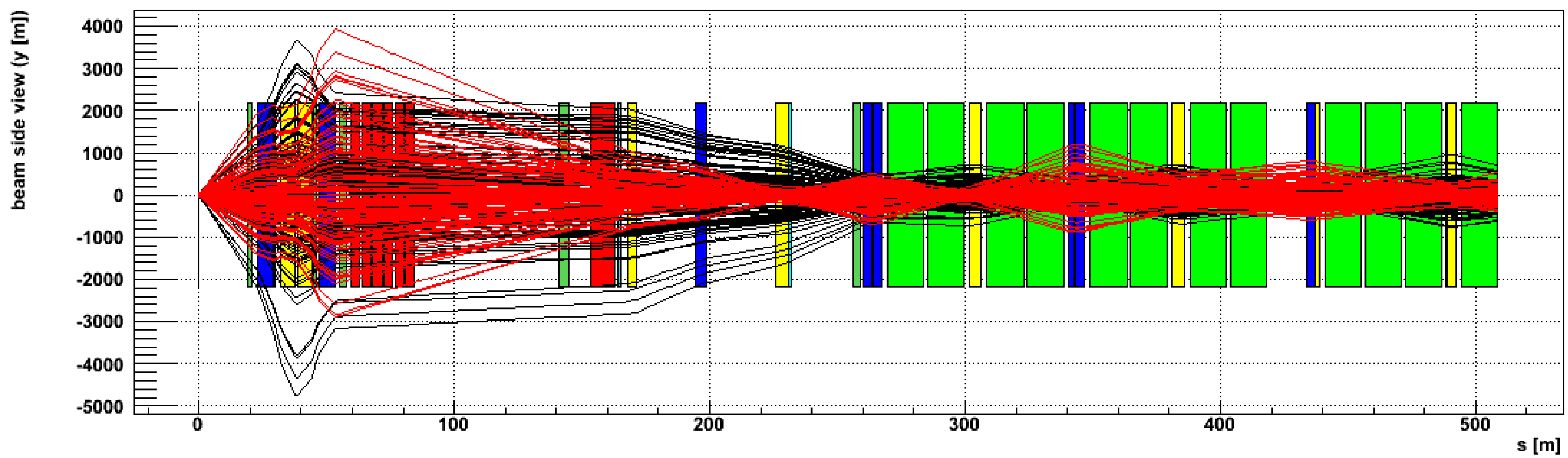
The 4-vector can be specified :

- completely (from generator)
- by choosing energy loss and Q^2 of emitted object

Implementation (VI)

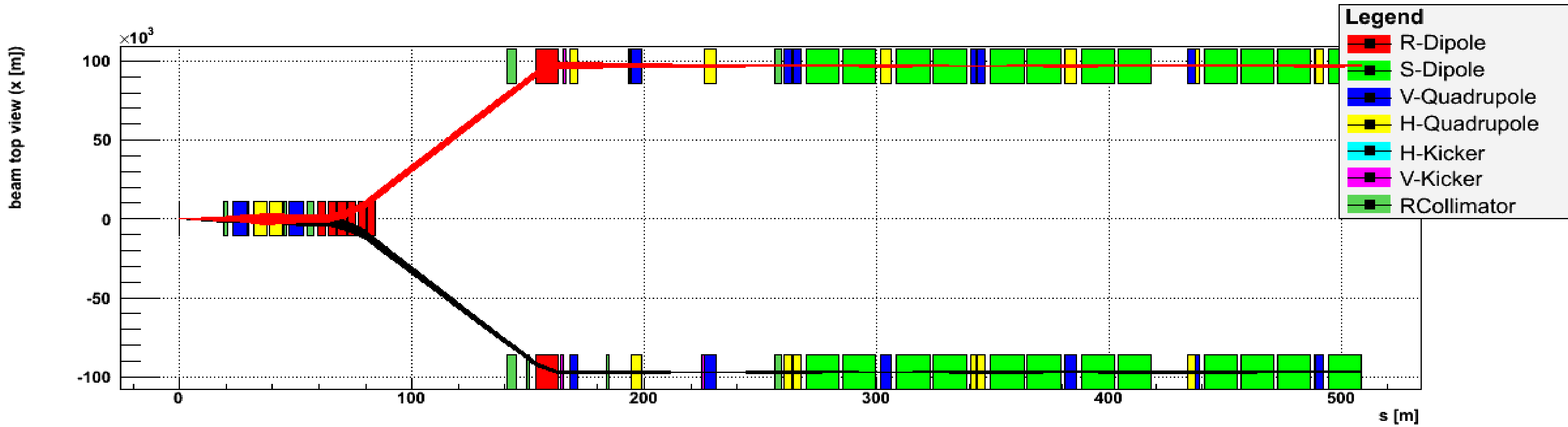
The LHC beams (right of CMS) :

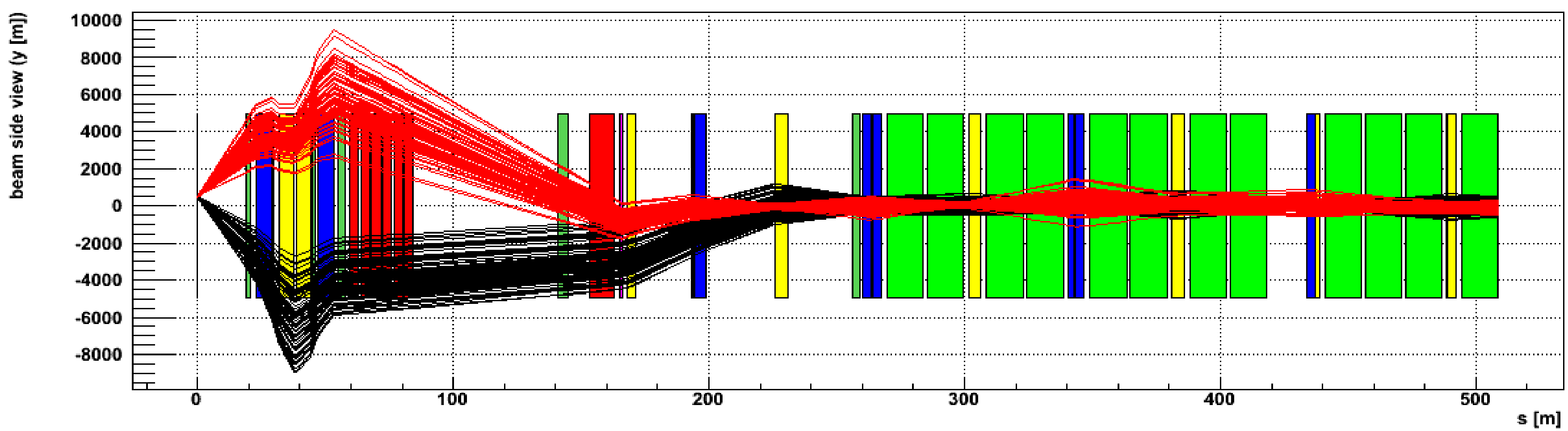




Implementation (VII)

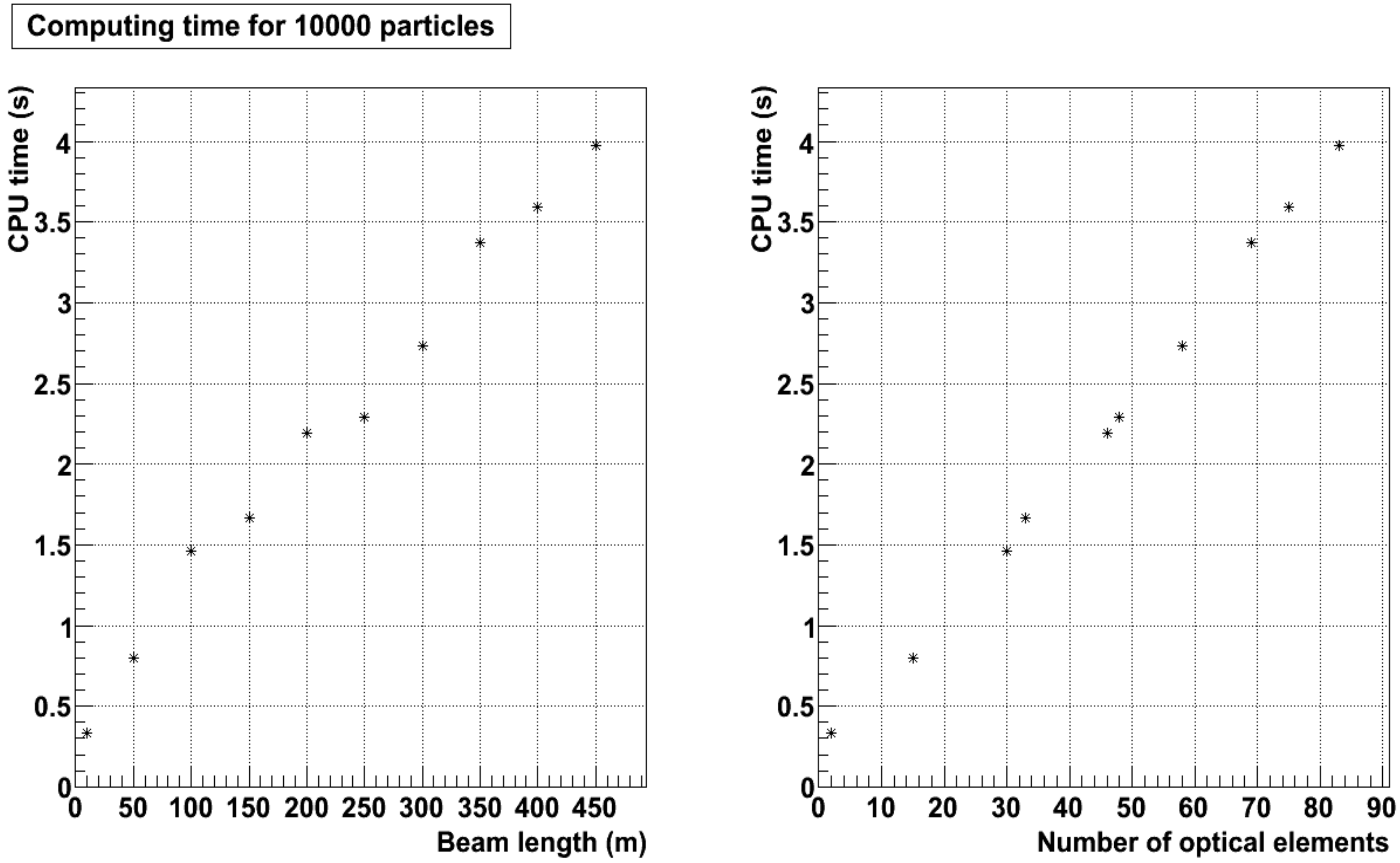
Same for ATLAS :





Implementation (VIII)

Performances :

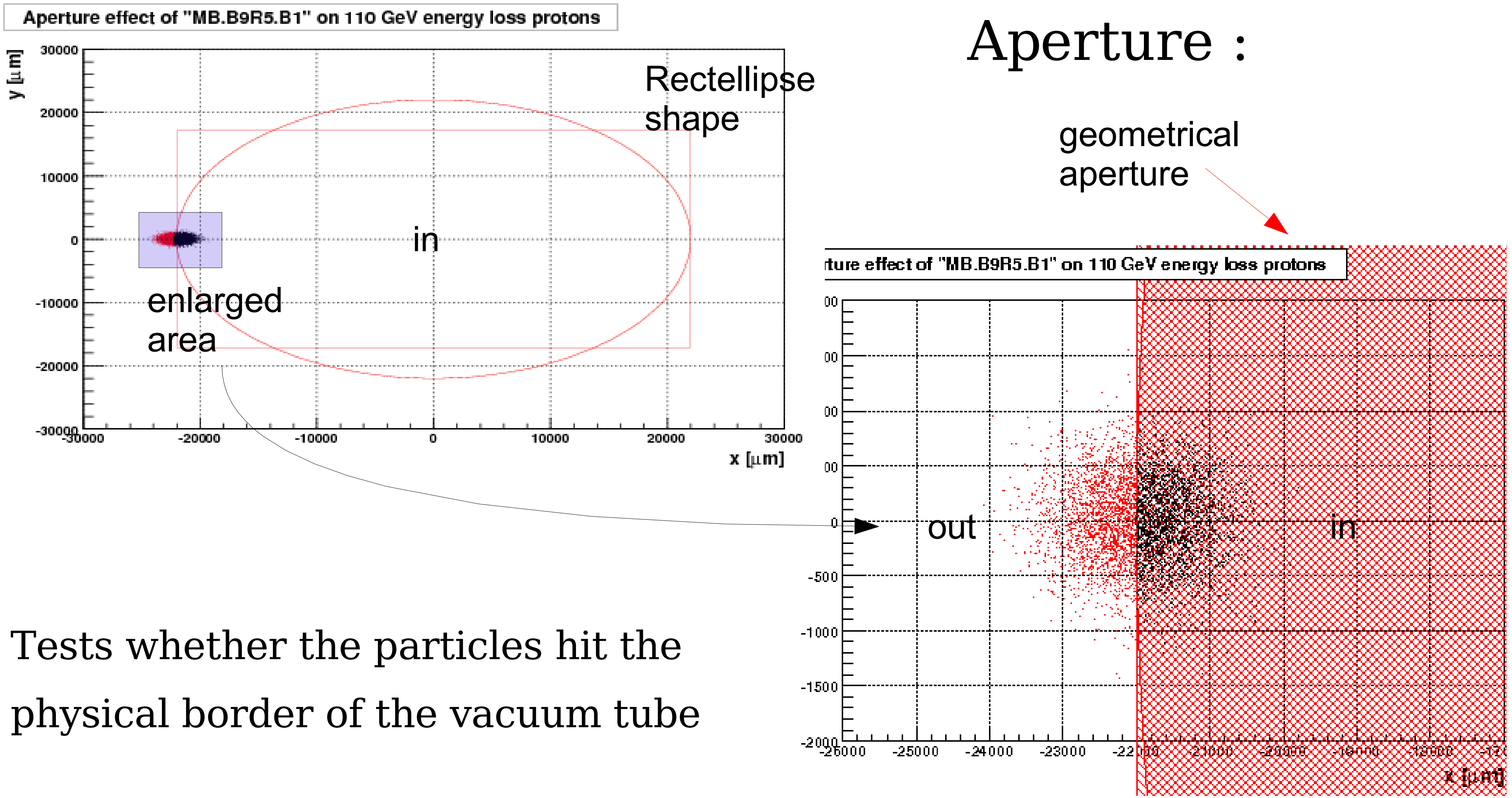


~ 3.5 µs $particle^{-1}$ $magnet^{-1}$

-> ~10^{-3} s / CMS event

Implementation (IX)

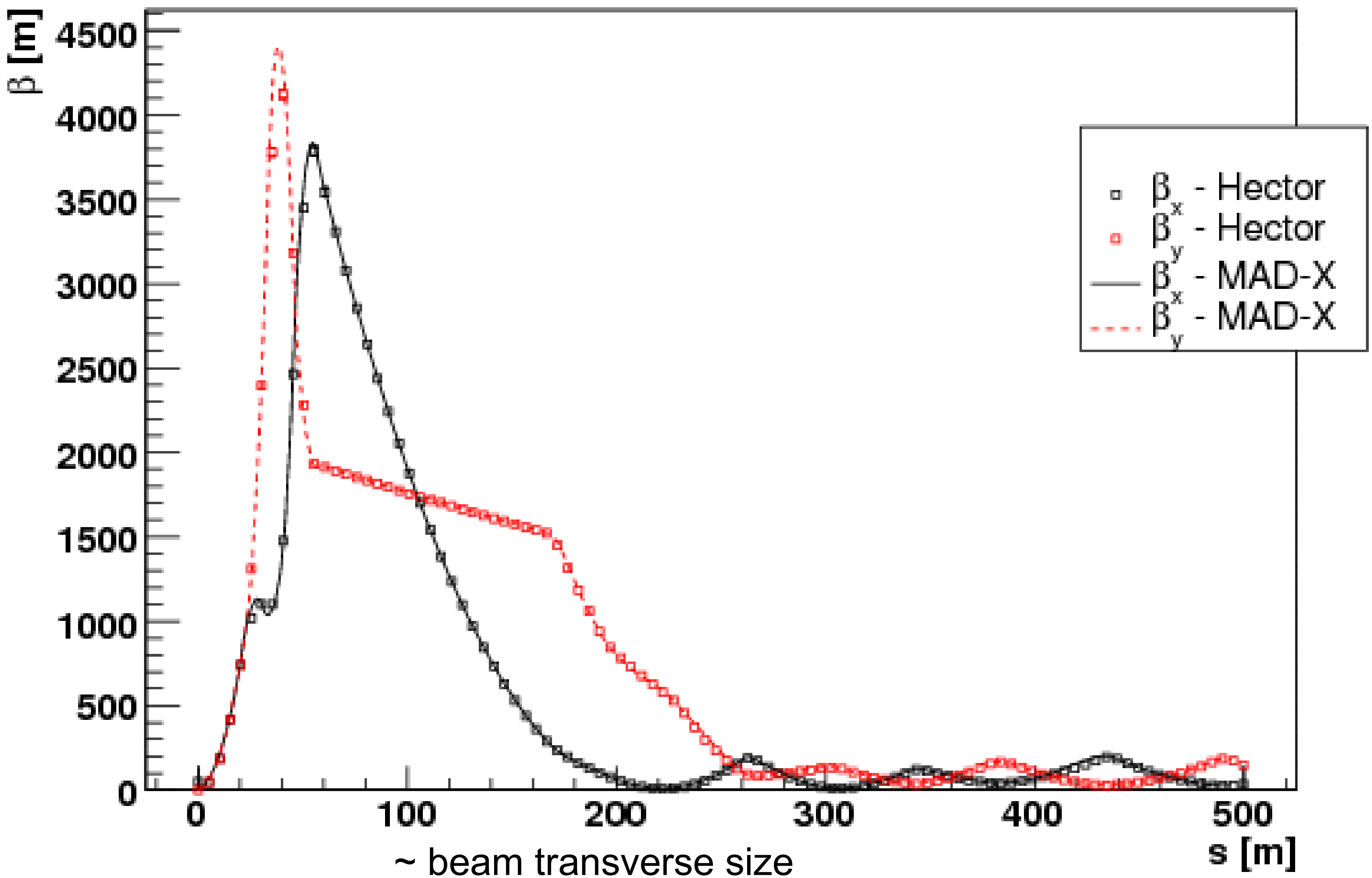
Aperture :



Tests whether the particles hit the physical border of the vacuum tube

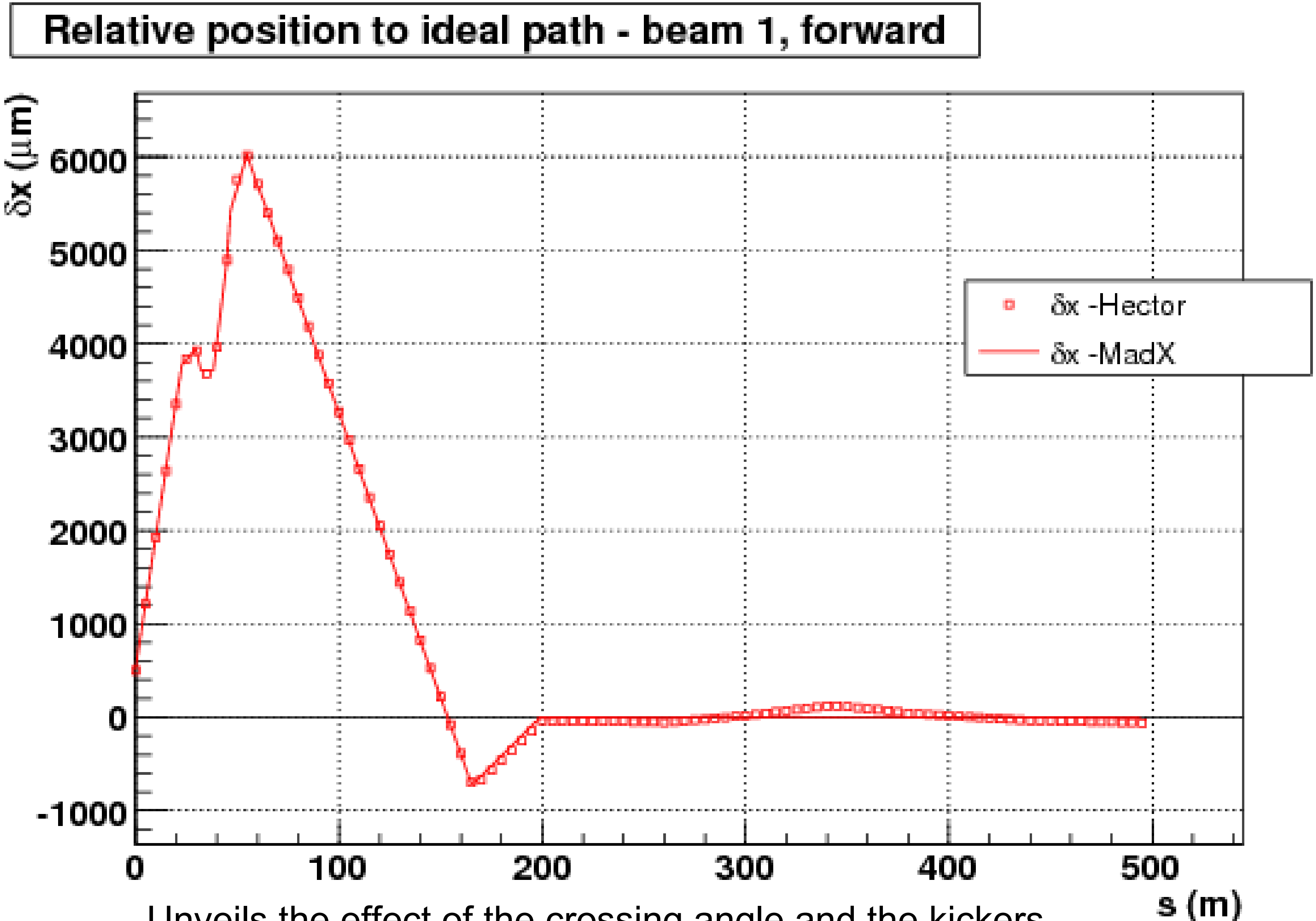
Validation (I)

β functions - beam 1, forward



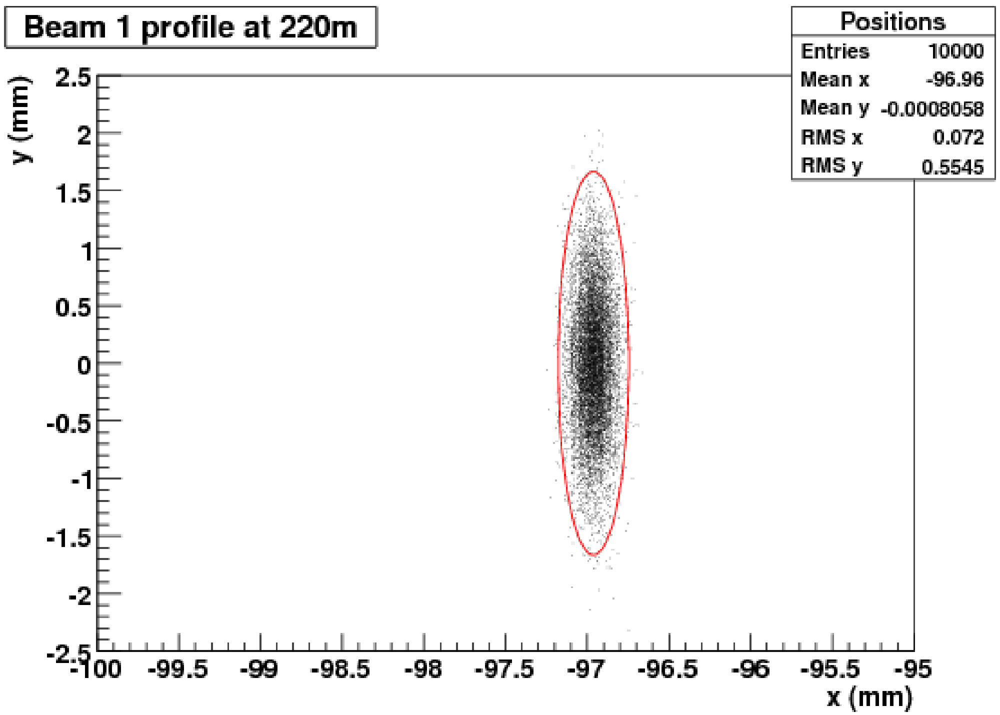
~ beam transverse size

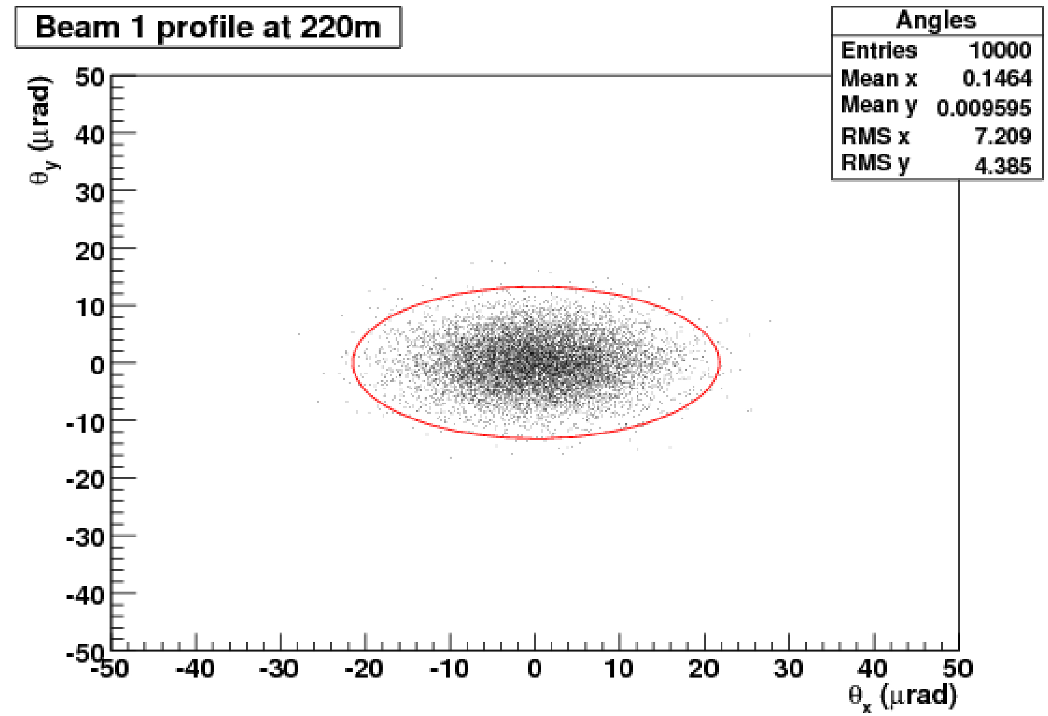
Validation (II)



Unveils the effect of the crossing angle and the kickers

Direct physics output (I)

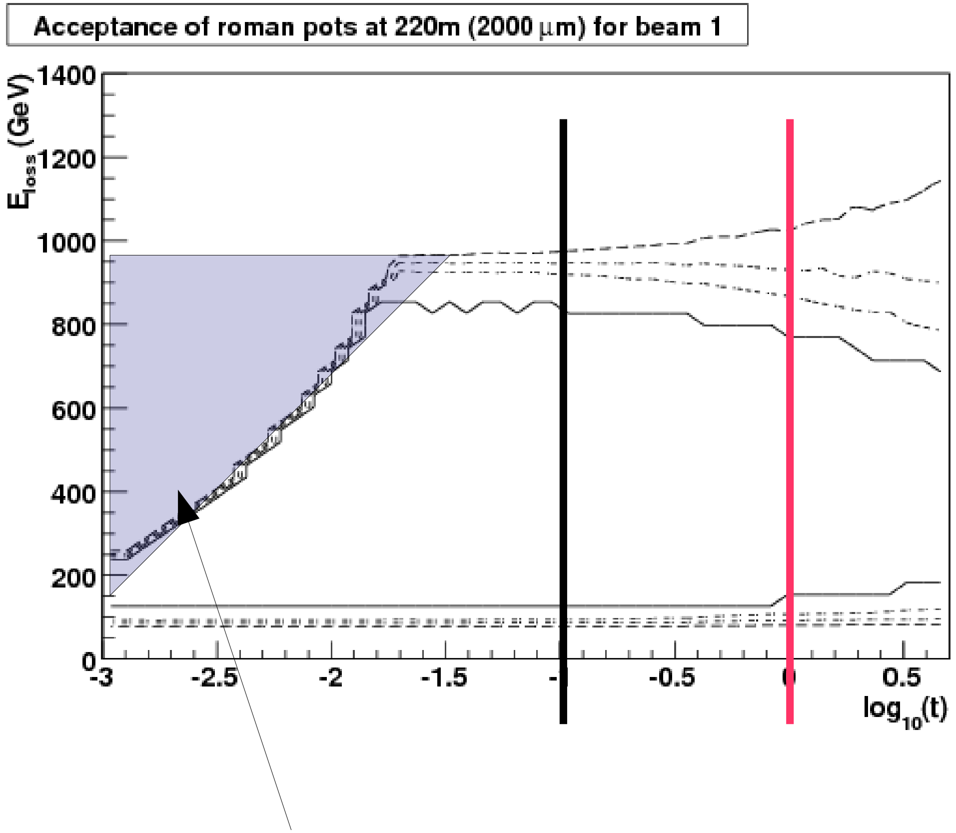




- Just take some protons, from LHC beam 1
- Propagate them to your favourite Roman pot detector
- Plot the x,y,x',y' in the transverse plane

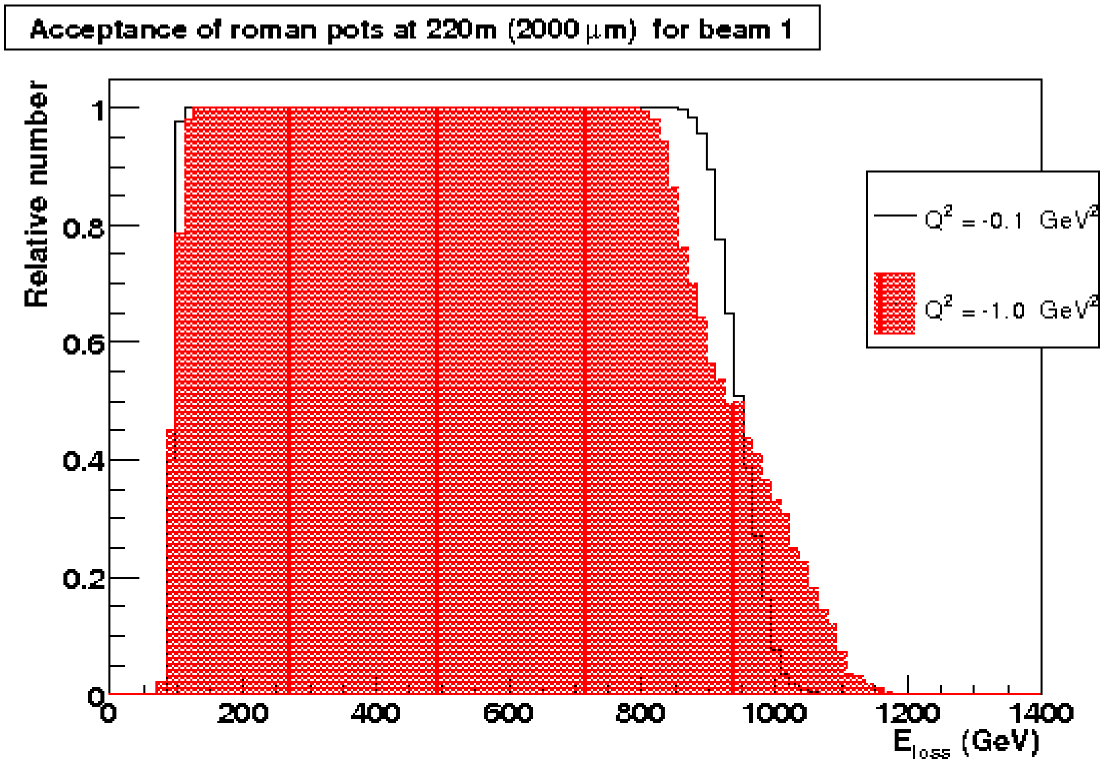
Direct physics output (II)

RP acceptances (220m) :



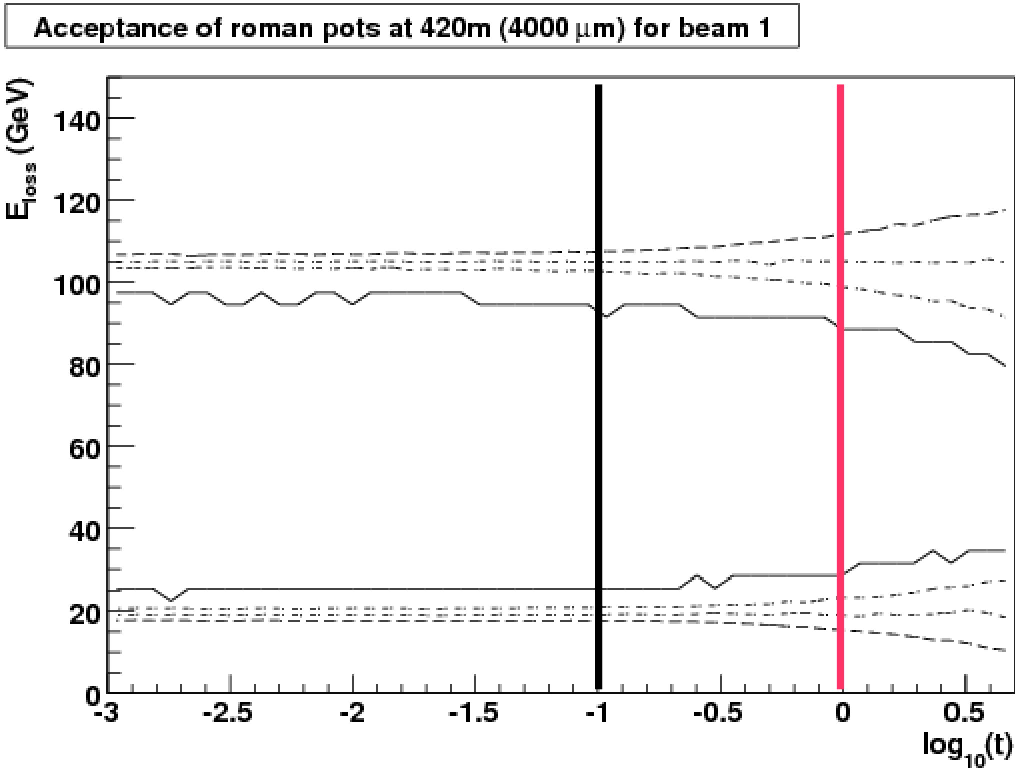
Forbidden by kinematics

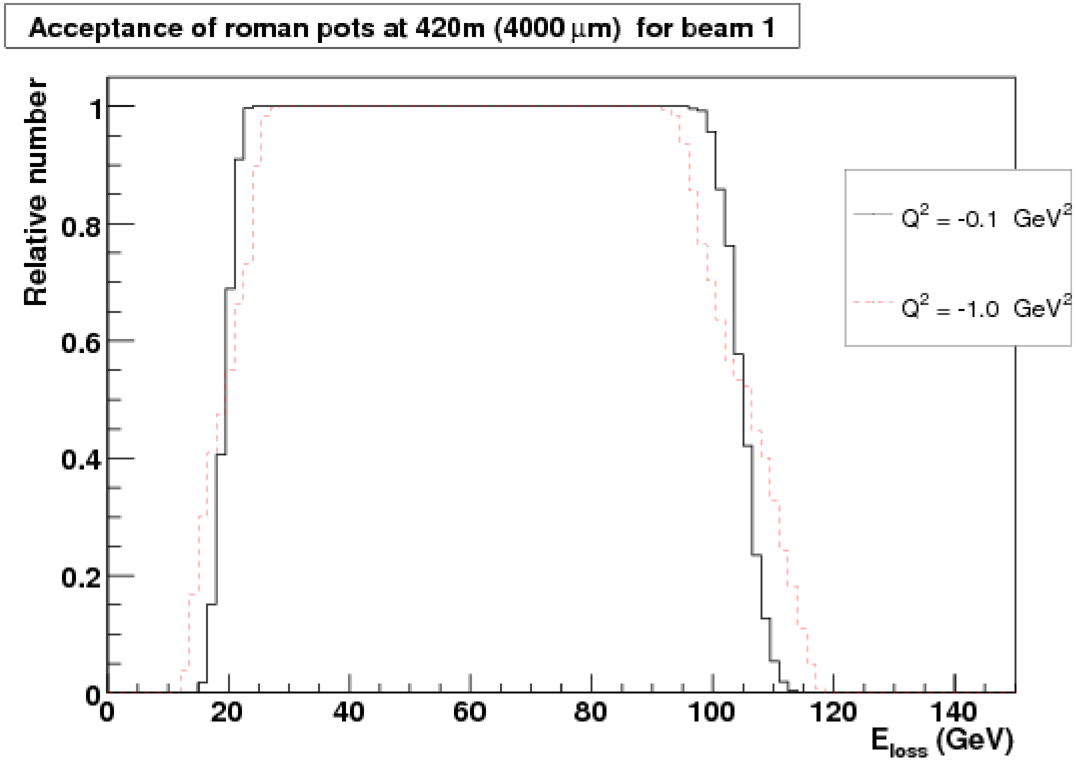
Which protons are detected ?



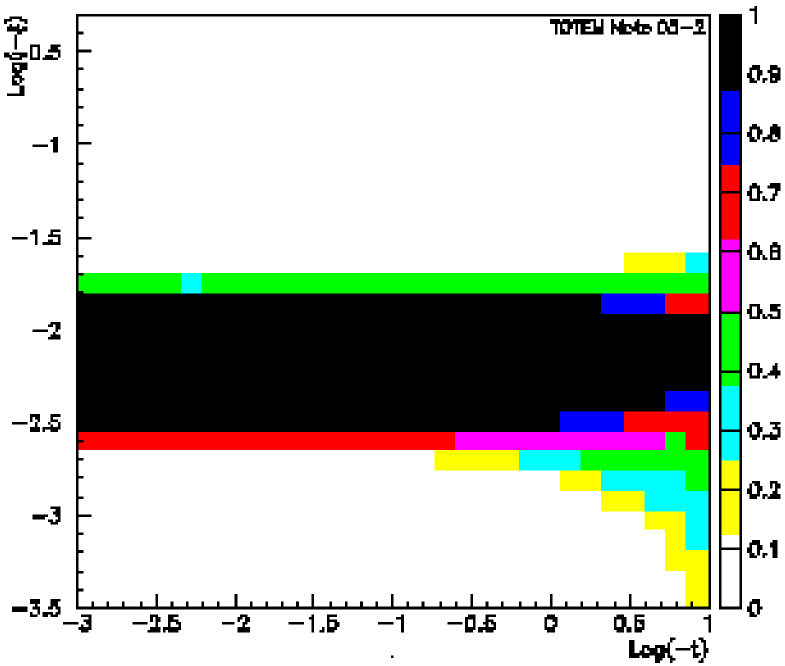
Direct physics output (III)

RP acceptances (420m)

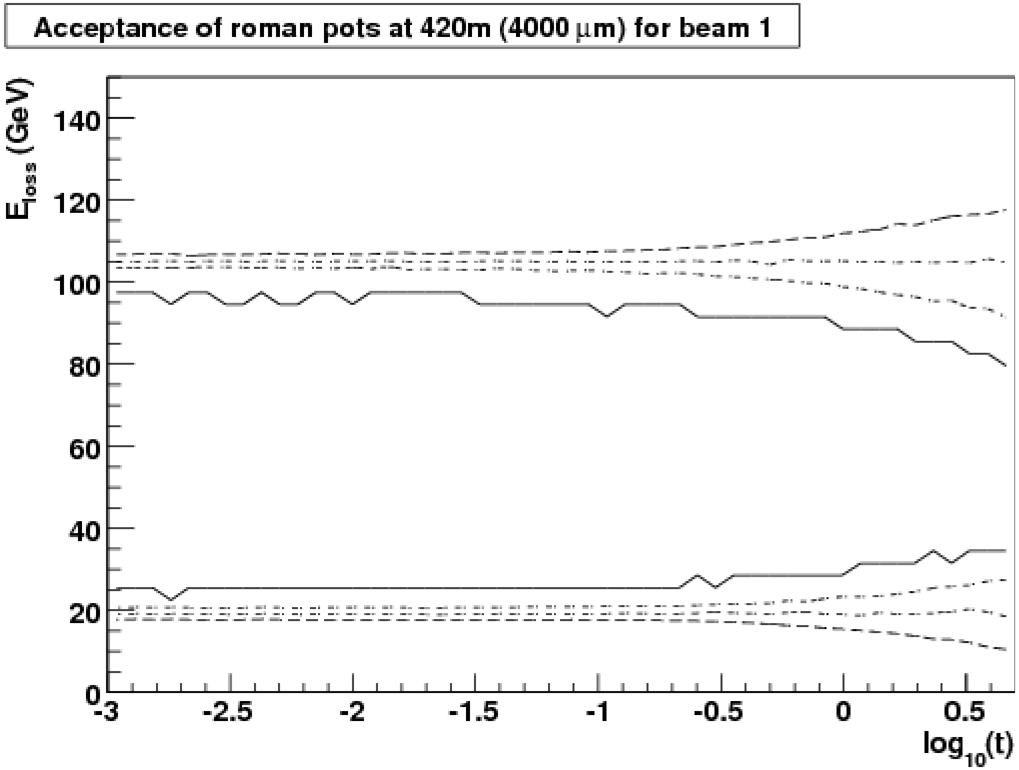




Turning MAD ?

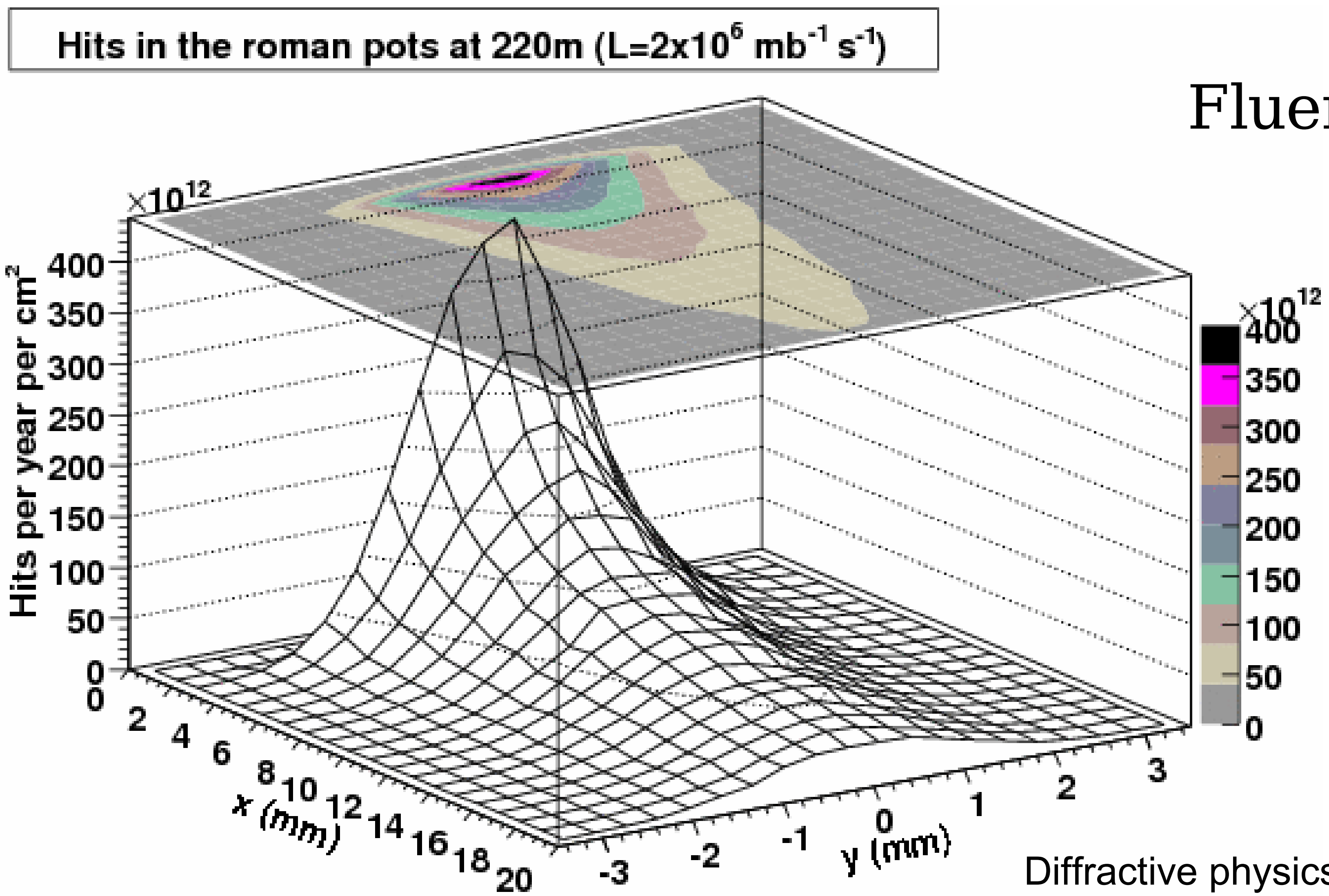


MAD-X (from TOTEM Note 05-2)



Hector (from my hard disk)

Direct physics output (V)



Fluencies :

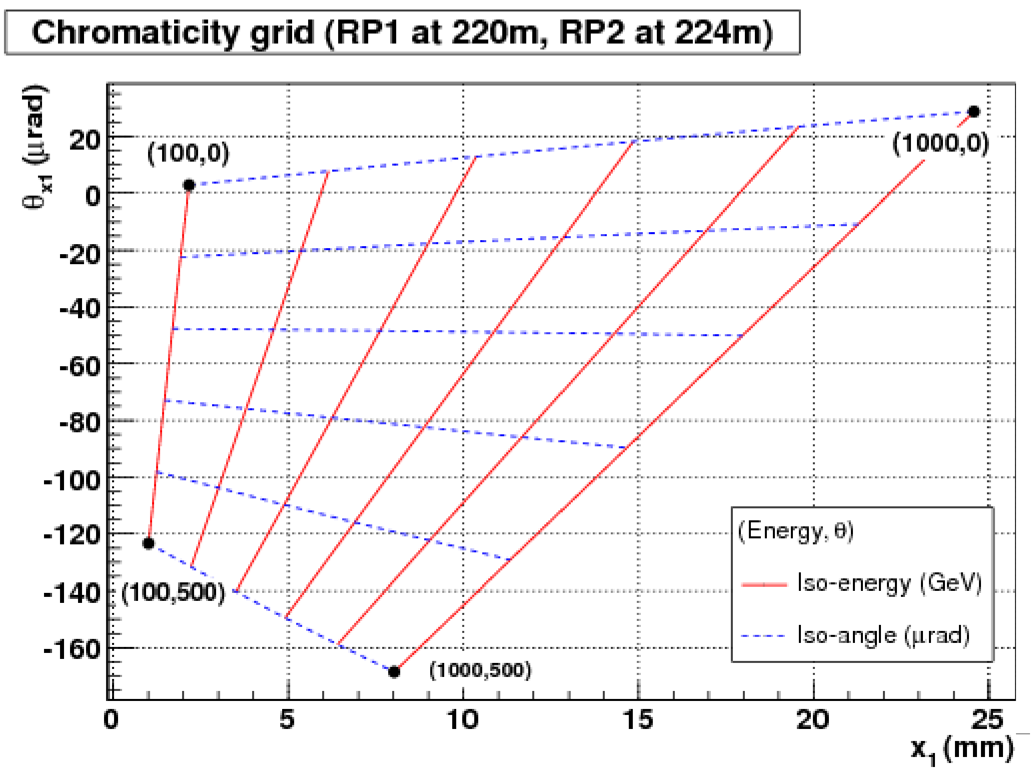
Diffractive physics : pp → pX
(PYTHIA inside)

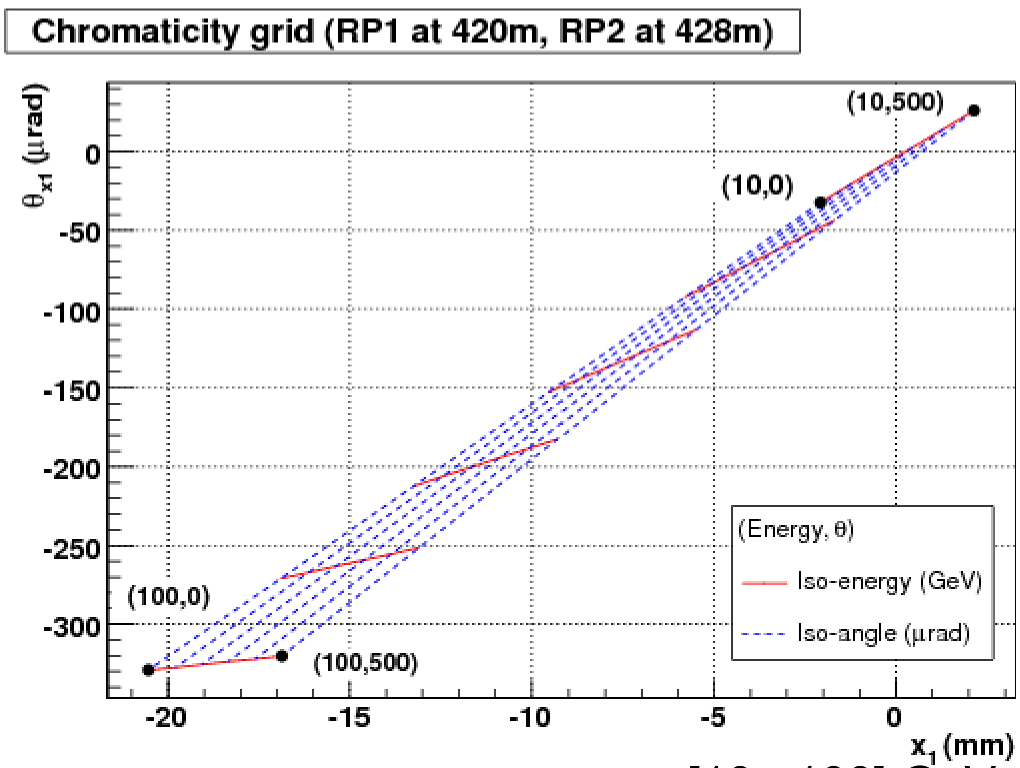
Direct physics output (VI)

Chromaticity grid :

where is your proton given its energy/angle ?

- Choose a proton, with a given energy loss and initial angle
- Propagate it to your 2 roman pots.
- Measure x,x'





[100 ; 1000] GeV ← Remember the acceptances ! → [10 ; 100] GeV

Reconstruction (I)

By linearity :

$$x_s = a_s x_0 + b_s x'_0 + d_s E$$
$$x'_s = \alpha_s x_0 + \beta_s x'_0 + \gamma_s E$$

We should solve for x_0, x_0', E (with only 2 equations)

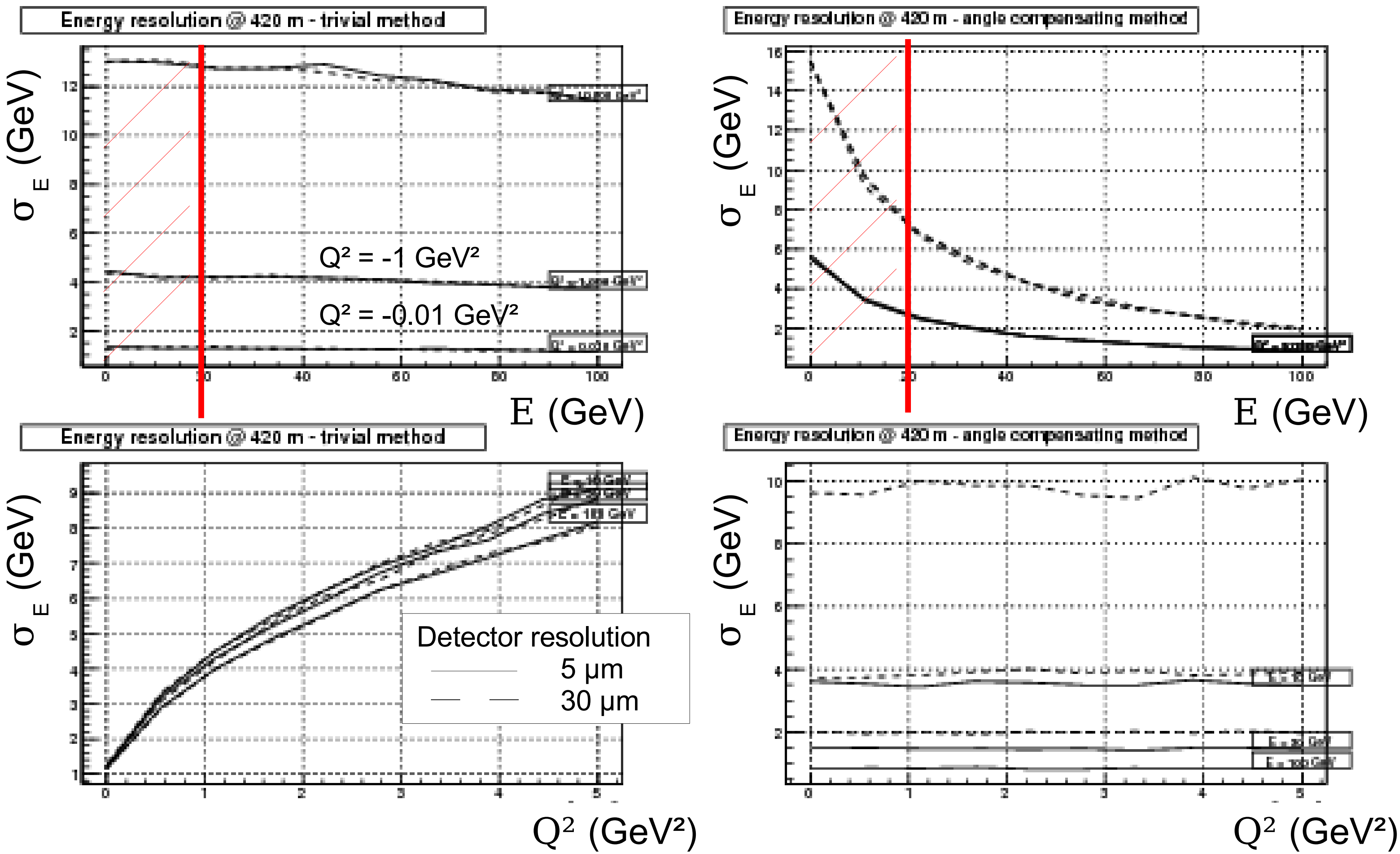
As physics won't change x_0, we choose to neglect a_s and α_s. This method leads to :

$$E = \frac{b_2 x_1 - b_1 x_2}{b_2 d_1 - b_1 d_2}$$ *Angle compensation method*

where b_1 and b_2 are the b parameters for the two detectors.

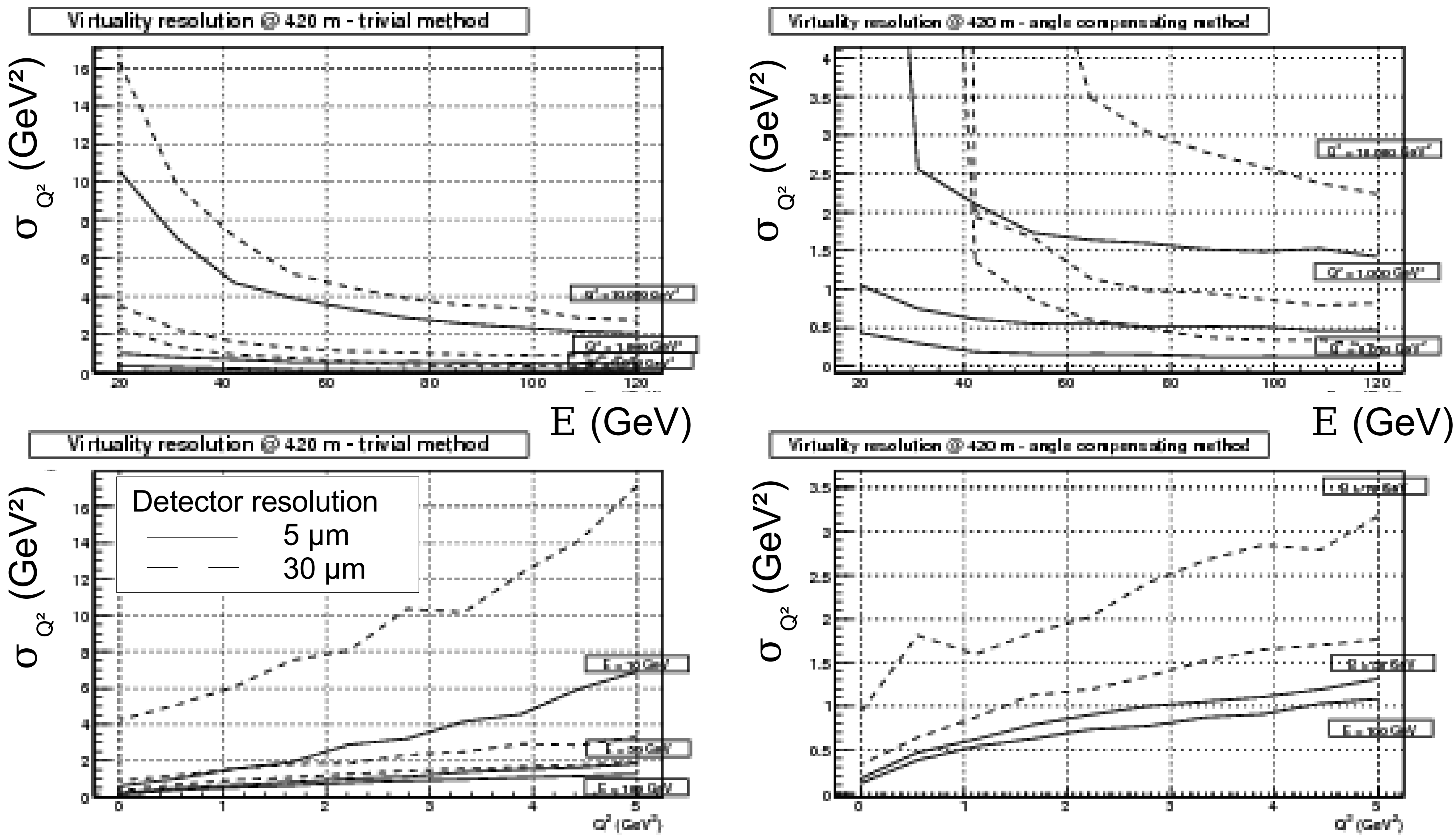
Reconstruction (II)

Reconstructed variables : energy loss (σ_E vs Q^2 and E)



Reconstruction (III)

Reconstructed variables : Q^2



Work-in-progress

Recent achievements :

- Non-proton particles propagation (mass, charge)
- Misalignment of magnets effect

In progress :

- Integration into FAMOS
- Integration into CMSSW

by some friends from Protvino.

Material :

Official website :

http://www.fynu.ucl.ac.be/hector.html

You will find there :

- Hector sources (stable or CVS)
- User Manual (kindly tested by TH-oriented CP3 members)
- Code documentation (Doxygen)
- Link to official forum
- Useful links
- Soon : note draft