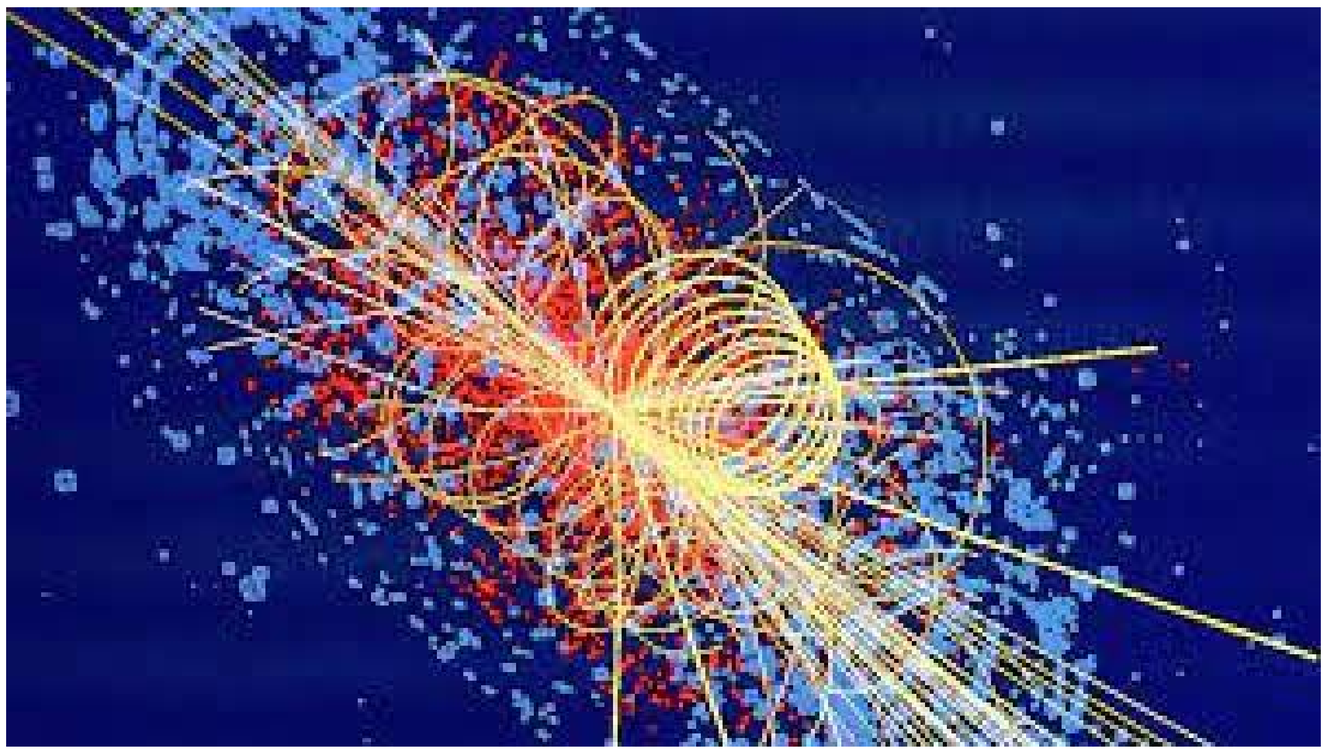
Flash-fit algorithms for circles in Particle Physics

Advanced Computing — *C++ library NXV4*

Maria Dima

Flash-algorithm:

- *non-iterative solution, and*
- *can compute χ^2-w/-no-fit*



When 2 particles collide into a detector (SPD for instance), the result will be a spray of particles. The trajectory of the particles is helical.

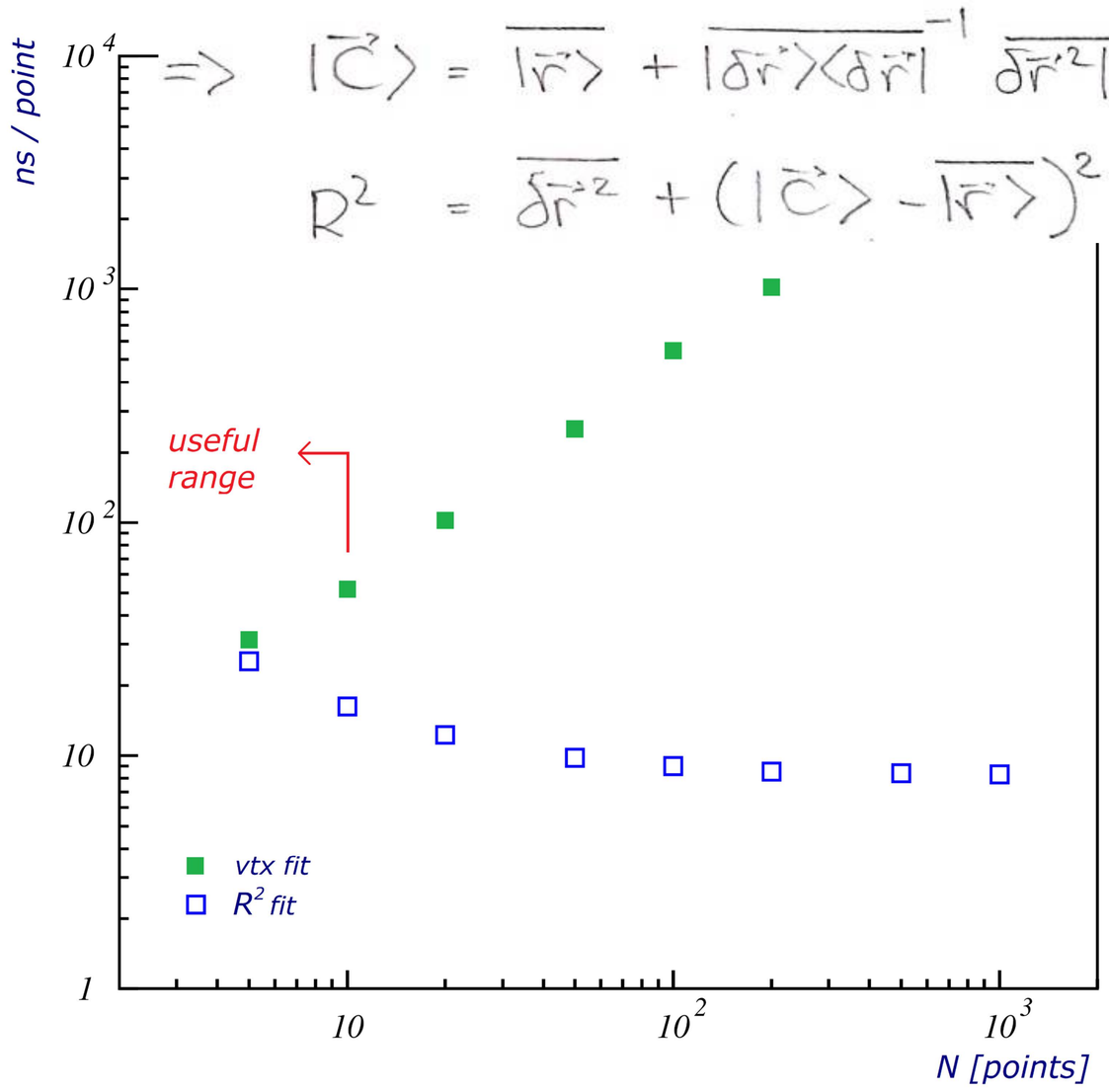
Helix fitting is very CPU intensive. Alternatively: perform a circle fit (in the transversal plane of the helix), from which find the s/2ρ parameter for each point, then verify that the longitudinal view is linear in this parameter. This method is much faster than a helix fit.

CIRCLE FIT 2 *– minimises residuals for R^2*

$$\sum \left[(\vec{r}_i - \vec{C})^2 - R^2 \right]^2 = \min.$$

$$\Rightarrow \quad |\vec{C}\rangle = \overline{|\vec{r}\rangle} + \overline{|\delta\vec{r}\rangle\langle\delta\vec{r}|}^{-1} \overline{\delta\vec{r}^2 |\delta\vec{r}\rangle}$$

$$R^2 = \overline{\delta\vec{r}^2} + \left(|\vec{C}\rangle - \overline{|\vec{r}\rangle}\right)^2$$

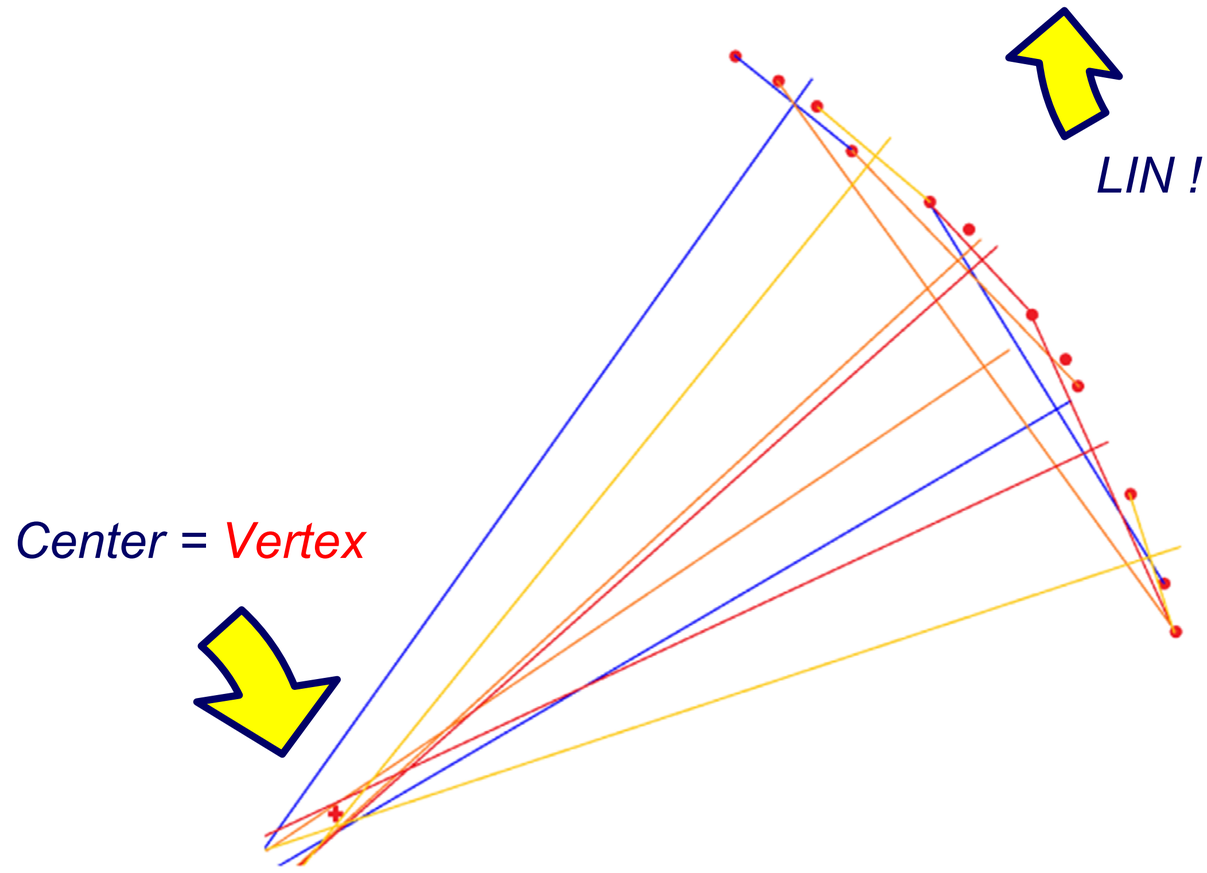


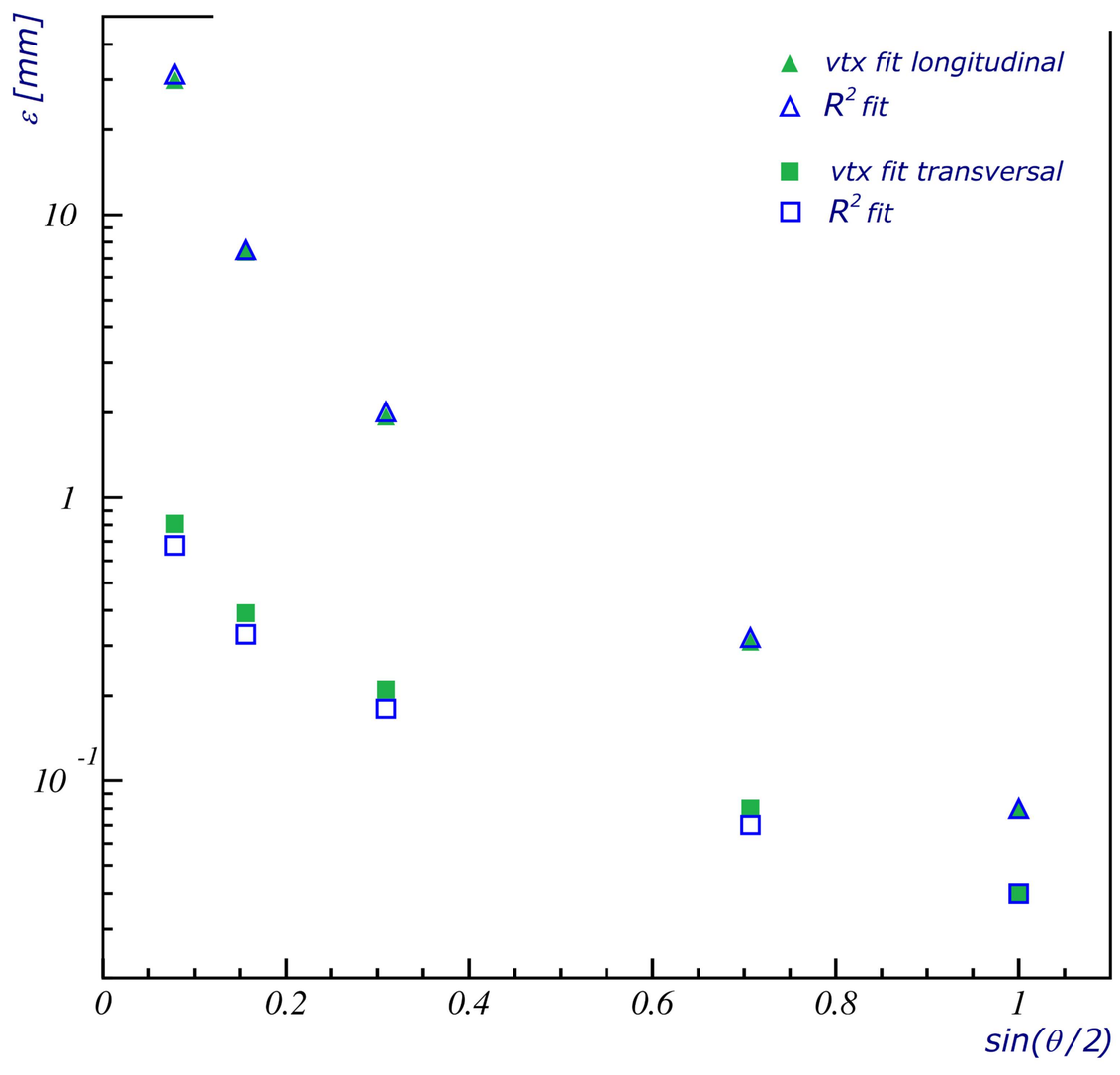
CIRCLE FIT 1 *(personal) – vertexing problem*

$$\vec{r}_{\text{vtx}} = \left[\mathbf{1} - \left\langle \frac{\vec{n}\vec{n}}{\vec{n}^2} \right\rangle \right]^{-1} \left\langle \left[\mathbf{1} - \frac{\vec{n}\vec{n}}{\vec{n}^2} \right] \vec{r}_0 \right\rangle$$

LIN !

Center = Vertex





CONCLUSIONS *– linearity is main advantage, error calc.*

– CPU disadvantage, C^2_N

– resolution comparable

– simple method to code

C++ gcc-9.3.1 20200408

Intel Broadwell, 2.9 GHz

FLNP65