# Swimming, a data driven acceptance correction algorithm

**M. Cattaneo, CERN**

**On behalf of the LHCb collaboration**

**CHEP 2012**

**22nd May 2012**

# Introduction

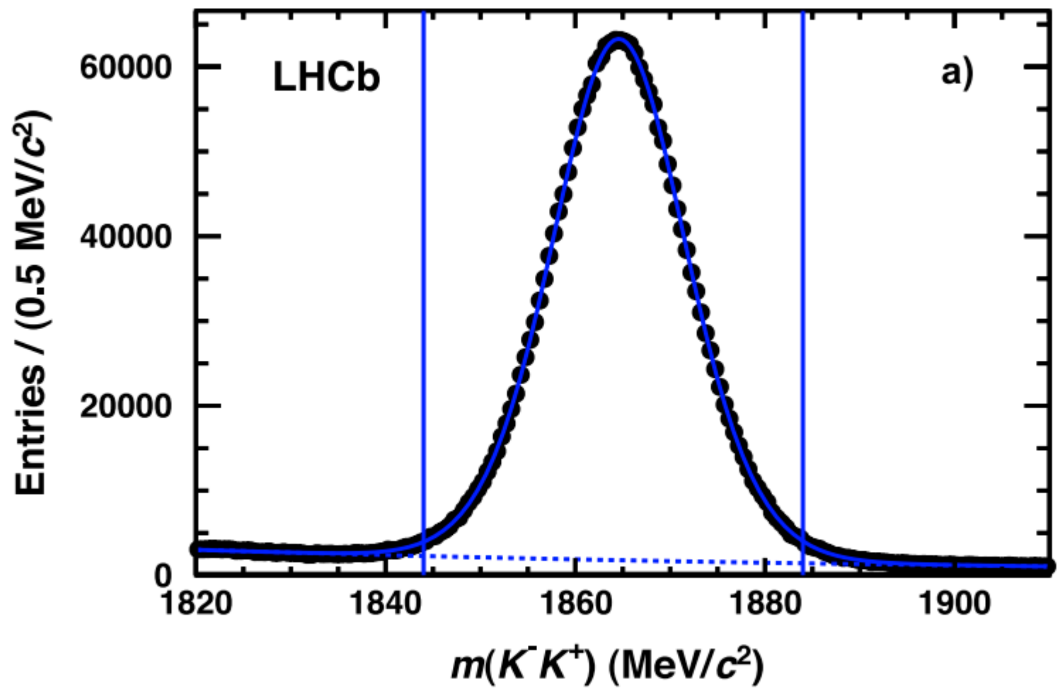
The LHC may not have yet produced the Higgs, but it is producing some of the biggest signal samples in the world

See $D^0 \to KK$ in the picture as an example : millions of signal events in a fraction of the data taken in 2011

Such large samples necessitate careful control of measurement biases

In particular biases introduced by the real-time event selection (trigger) which chooses which events to save for offline analysis

Here we will present a method for measuring these biases on data alone in the context of the measurements of meson lifetimes



# The history of swimming

Method first used at NA11 for measurements of charm meson lifetimes

Subsequently the method was developed and used at DELPHI and CDF for the measurement of beauty meson lifetimes

- Somewhat complicated at CDF owing to need to emulate a hardware lifetime biasing trigger offline, and non-uniform tracking efficiency as a function of meson lifetimes
- LHCb learned a lot from the challenges CDF faced making these measurements and many key design decisions were guided by their experiences

LHCb is the first experiment where the DAQ and analysis model have been built with the swimming in mind; indeed a large fraction of our physics programme relies on this method now.

[2] R. Bailey et al., *Measurement of the lifetime of charged and neutral D mesons with high resolution silicon strip detectors*, *Z. Phys.* **C28** (1985) 357–363.

[3] DELPHI, W. Adam et al., *Lifetimes of charged and neutral B hadrons using event topology*, *Z. Phys.* **C68** (1995) 363–374.

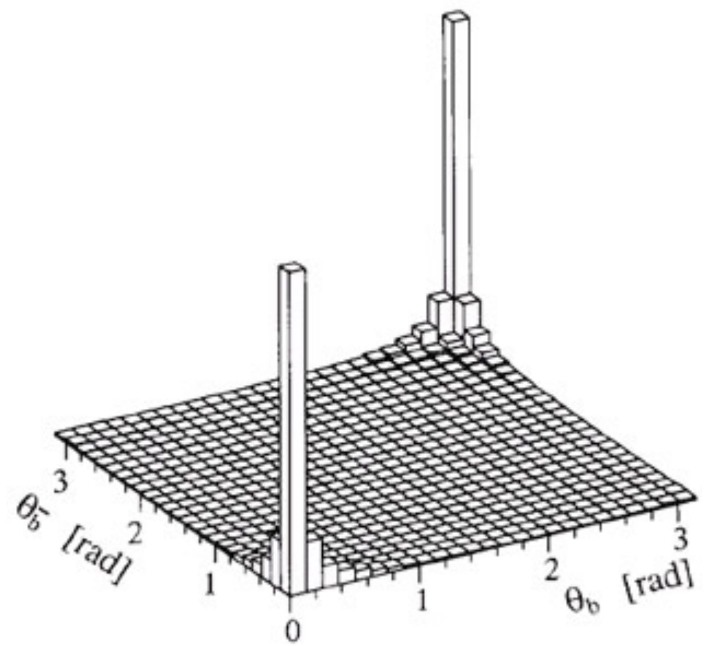
[4] J. Rademacker, *Reduction of statistical power per event due to upper lifetime cuts in lifetime measurements*, *Nucl. Instrum. Meth.* **A570** (2007) 525.

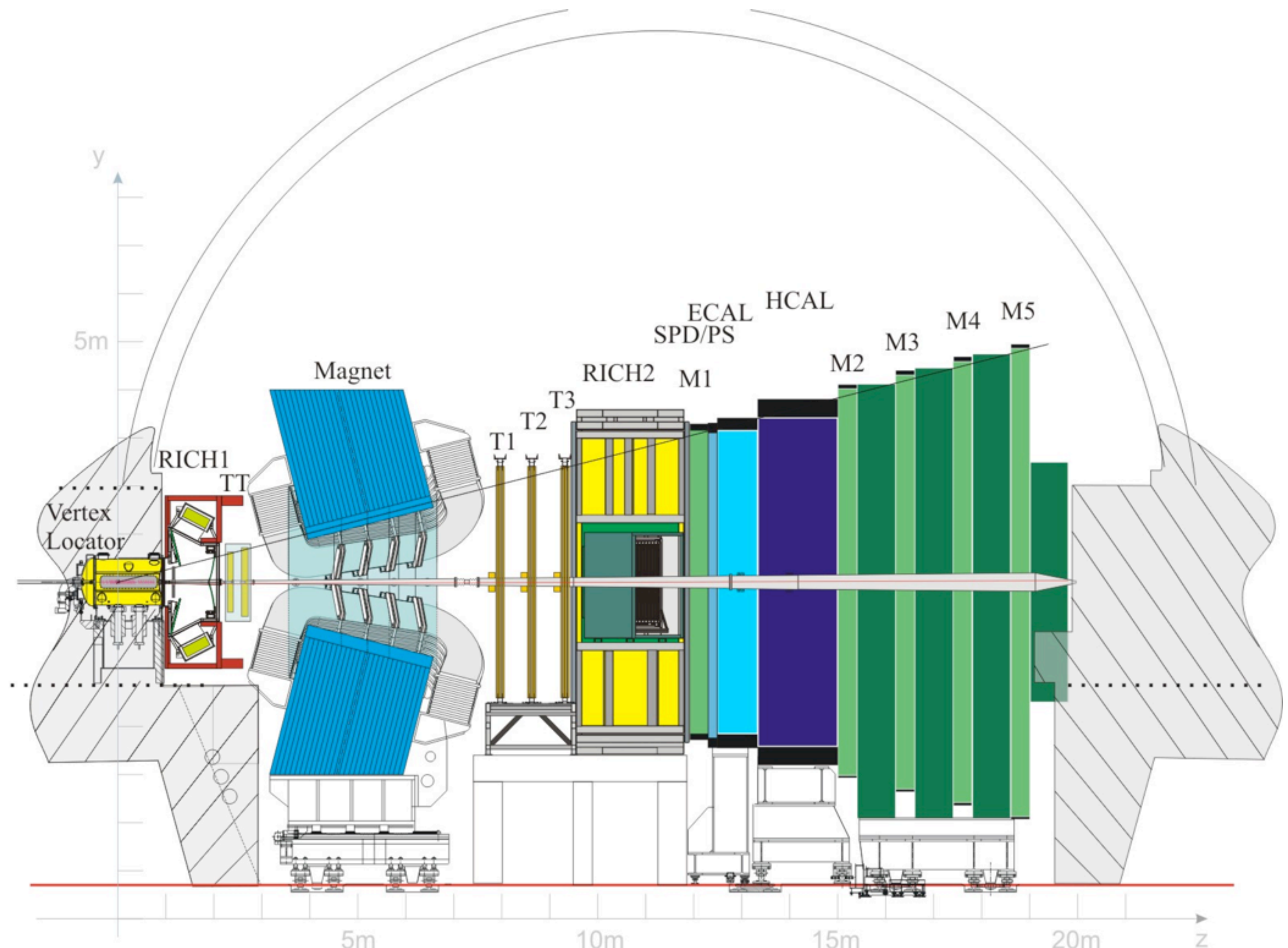
[5] CDF, T. Aaltonen et al., *Measurement of the $B^-$ lifetime using a simulation free approach for trigger bias correction*, *Phys. Rev.* **D83** (2011) 032008.

[6] V. V. Gligorov, *Measurement of the CKM angle gamma and B meson lifetimes at the LHCb detector.* PhD thesis, 2008. CERN-THESIS-2008-044.

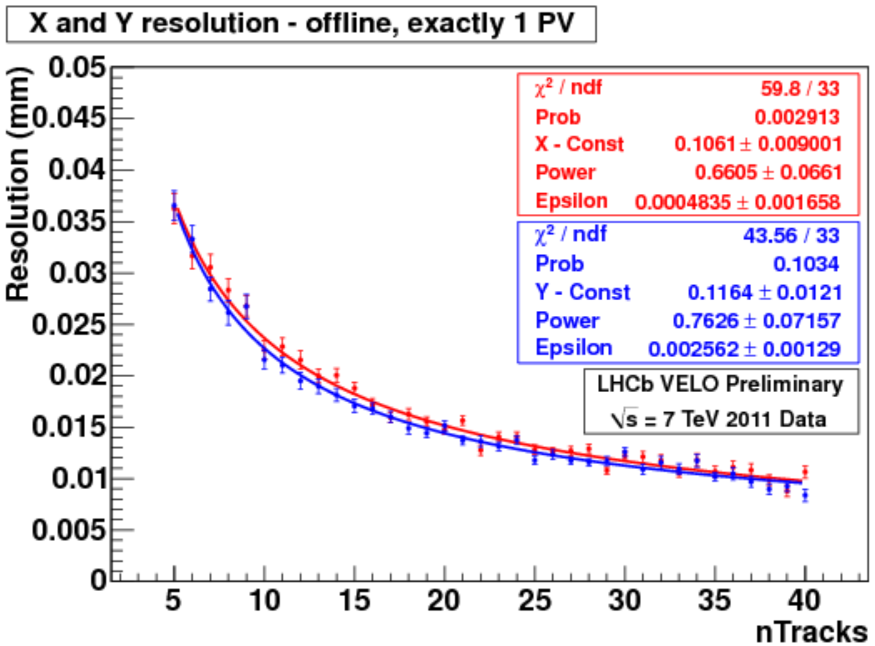
[7] M. Gersabeck, *Alignment of the LHCb Vertex Locator and Lifetime Measurements of Two-Body Hadronic Final States.* PhD thesis, 2009. CERN-THESIS-2009-118.

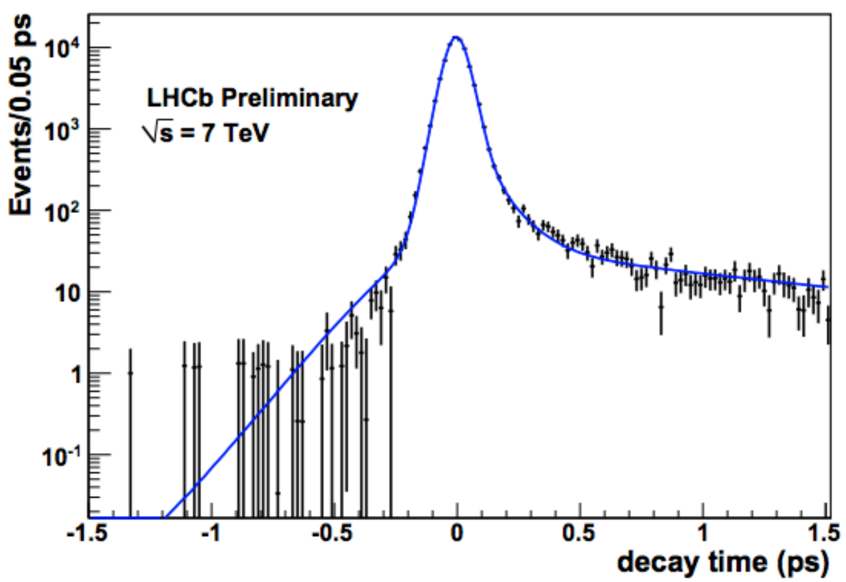
# The LHCb detector

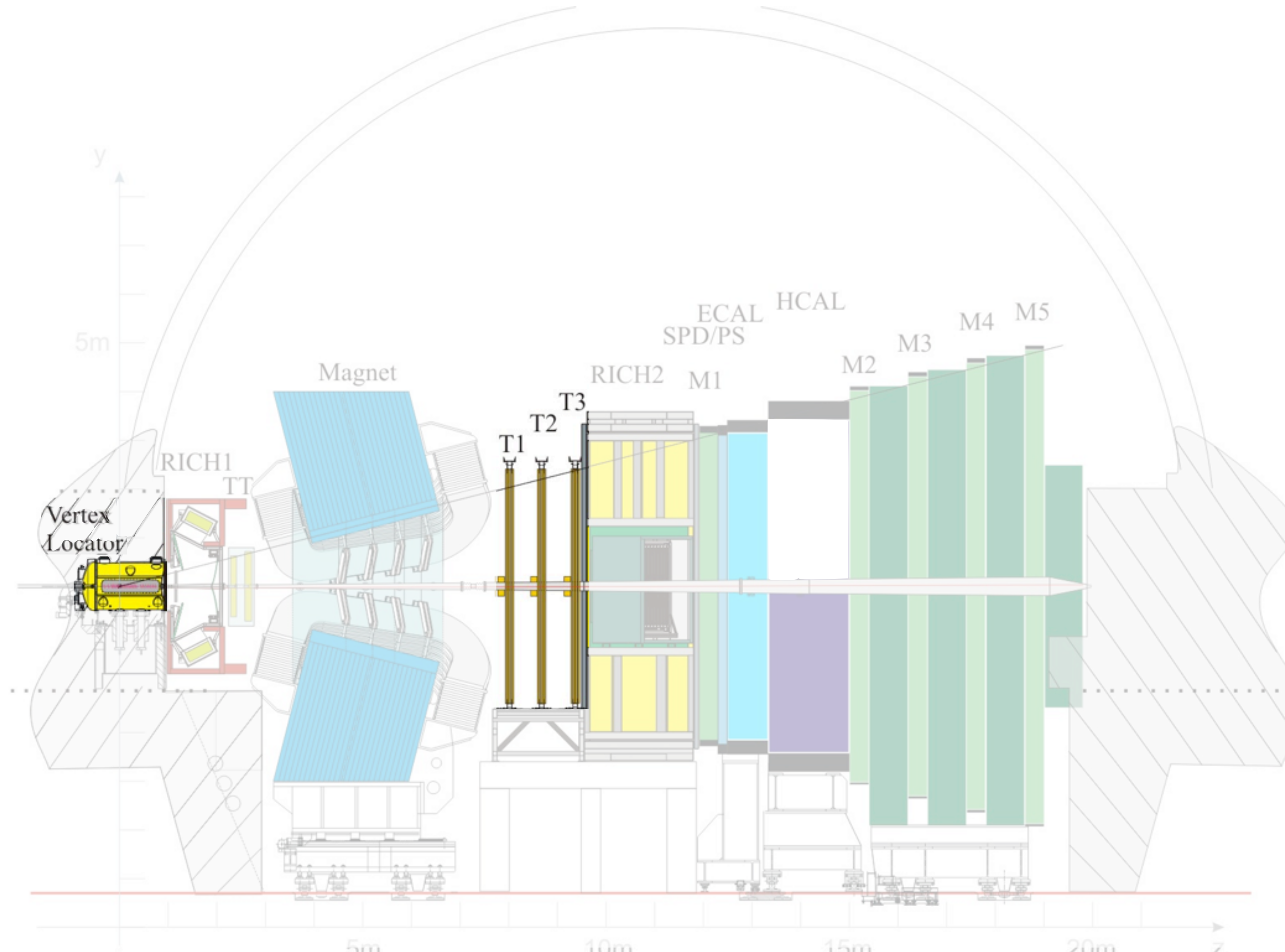


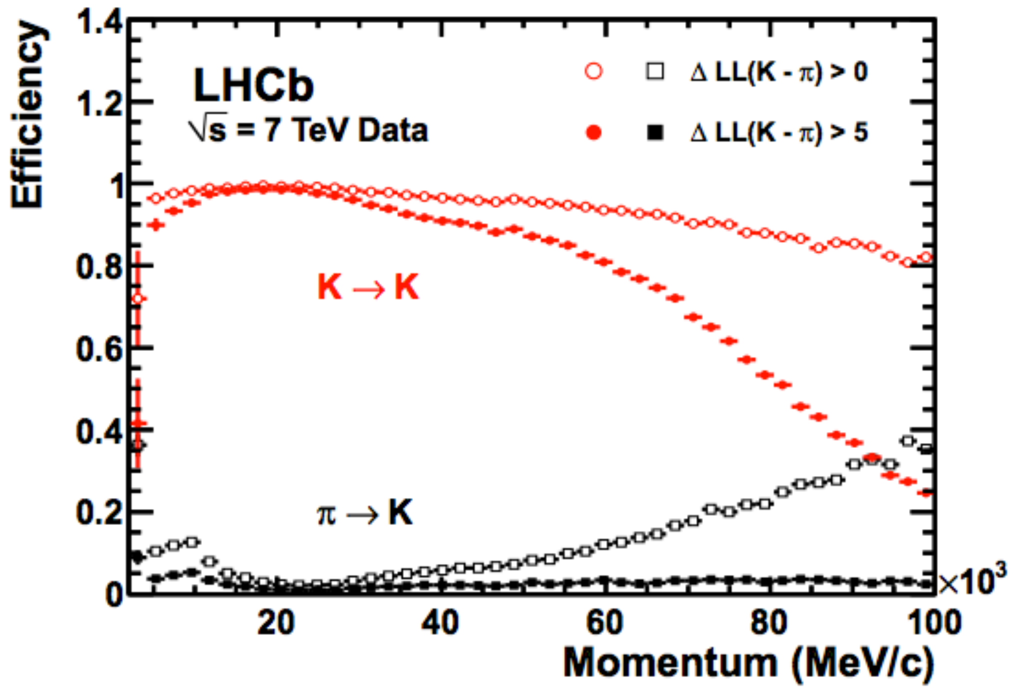
# The LHCb detector

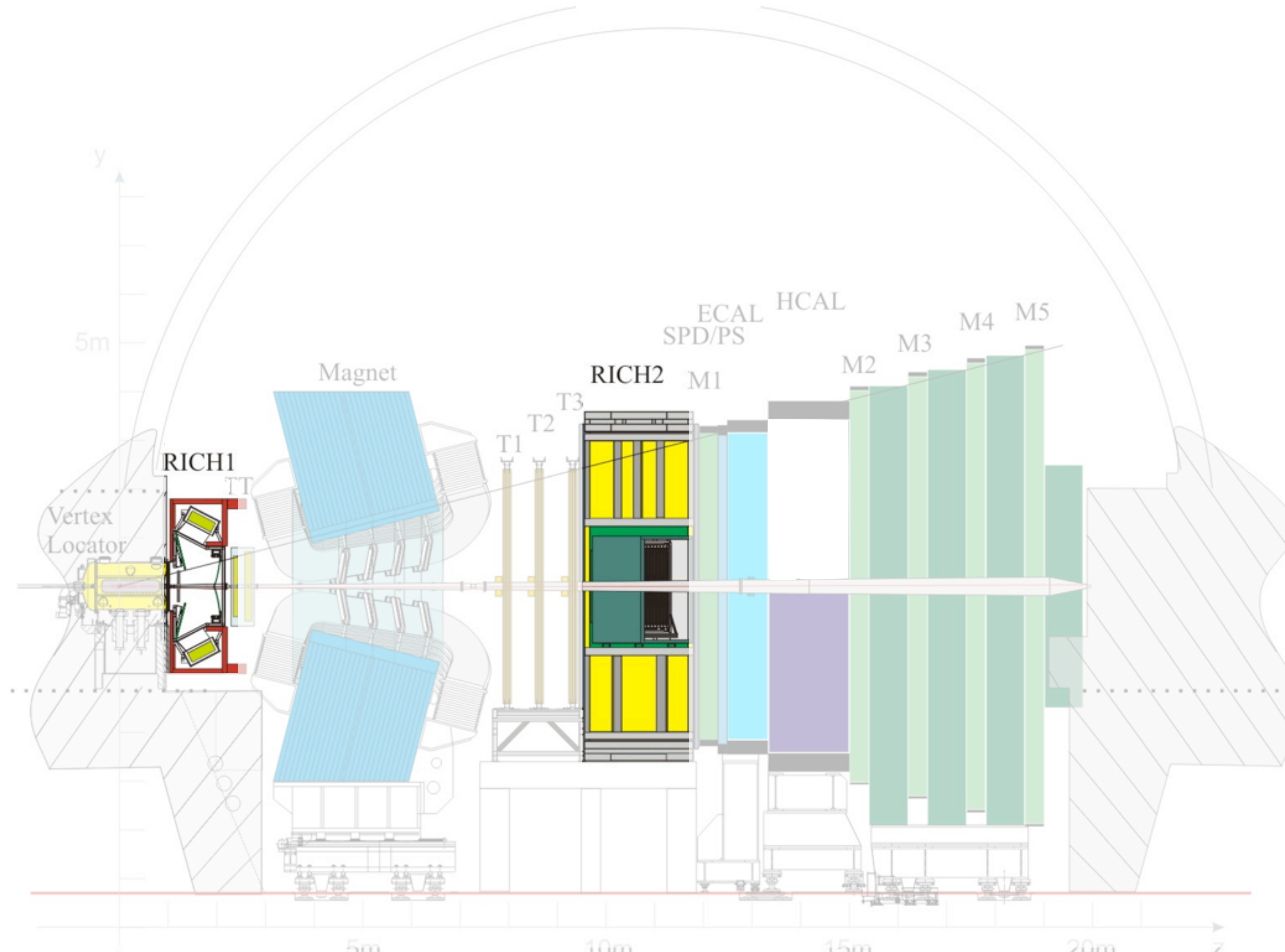


# The LHCb detector



# The LHCb DAQ/trigger

15 MHz pp interactions

1 MHZ Detector readout

| 450 Hz | 350 Hz | 120 Hz | 80 Hz |
|---|---|---|---|
| $h^{\pm}$ | $\mu$ | $e/\gamma$ | $\mu\mu$ |

Software trigger :

- 29000 Logical CPU cores
- Access to the full event information
- Use offline reconstruction software tuned for HLT time constraints

3 kHz data output

We can only read out the full detector information at 1 MHz, hence need a hardware trigger

- Trigger on high transverse momentum and energy deposits in the calorimeters and muon chambers

Subsequently, events are processed by the High Level Trigger (HLT), implemented in software

- This achieves a further factor 330 rate reduction, but in doing so introduces biases in the event selection which must later be corrected for

See also "The Software Architecture of the LHCb High Level Trigger"
Talk id 146 by Mariusz WITEK on 21 May 2012 from 17:50 to 18:15 (track 1)

# HLT architecture concept

**Implemented in an "off the shelf" CPU farm, with the following principles**

**-- All detector information is in principle available at all times**

**-- Uses the offline software, reconstruction, and selection code out of the box wherever possible.**

**-- Where "special" reconstruction and/or selection must be done for reasons of time, it is implemented by setting options within the offline software, not by writing custom code.**

**-- Trigger decisions must be reproducible offline, including downscaling of trigger lines. If we take a triggered event and rerun with the same trigger configuration, we must get the same answer.**

# Reproducing the HLT

**In order to rerun the HLT on a given event, it is enough to know**

- **-- The version of the trigger software which ran during data taking;**
- **-- The trigger configuration key (TCK) which fully defines the order of trigger algorithms and their properties;**
- **-- The DDDB tag which defines the geometrical description of the LHCb detector propagated to the trigger at the time of data taking;**
- **-- The LHCBCOND tag which defines condition of the detector at the time of data taking, for example the alignment constants.**

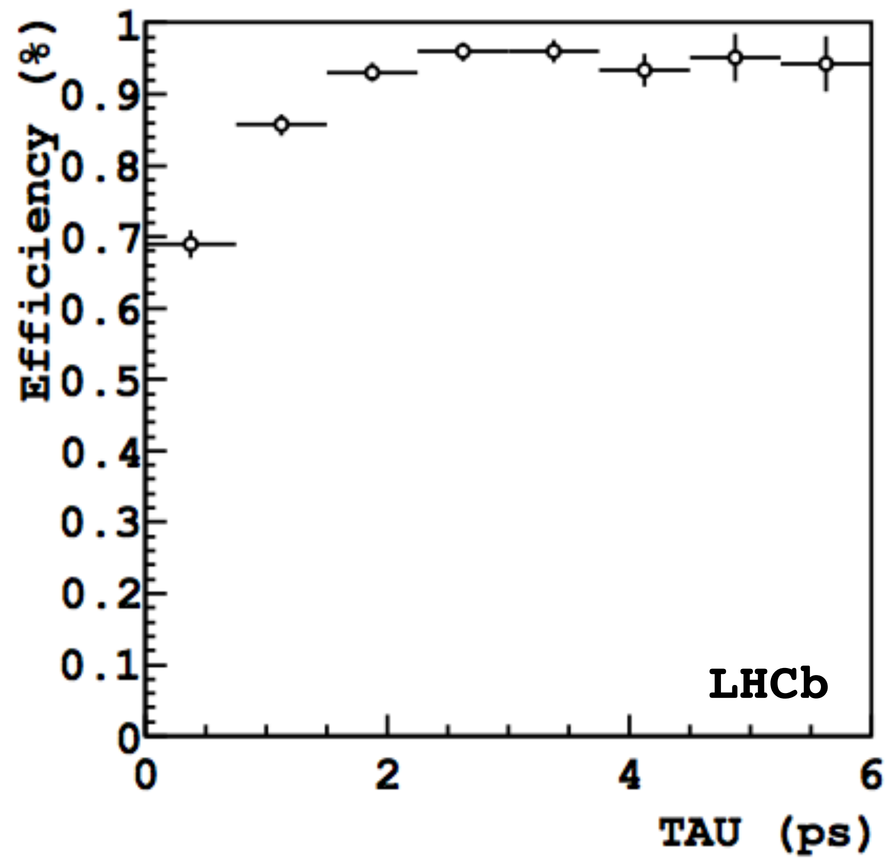
# Example of HLT biases

**The figure shows the efficiency of the HLT to select a B→Dπ decay as a function of the decay time of the B**

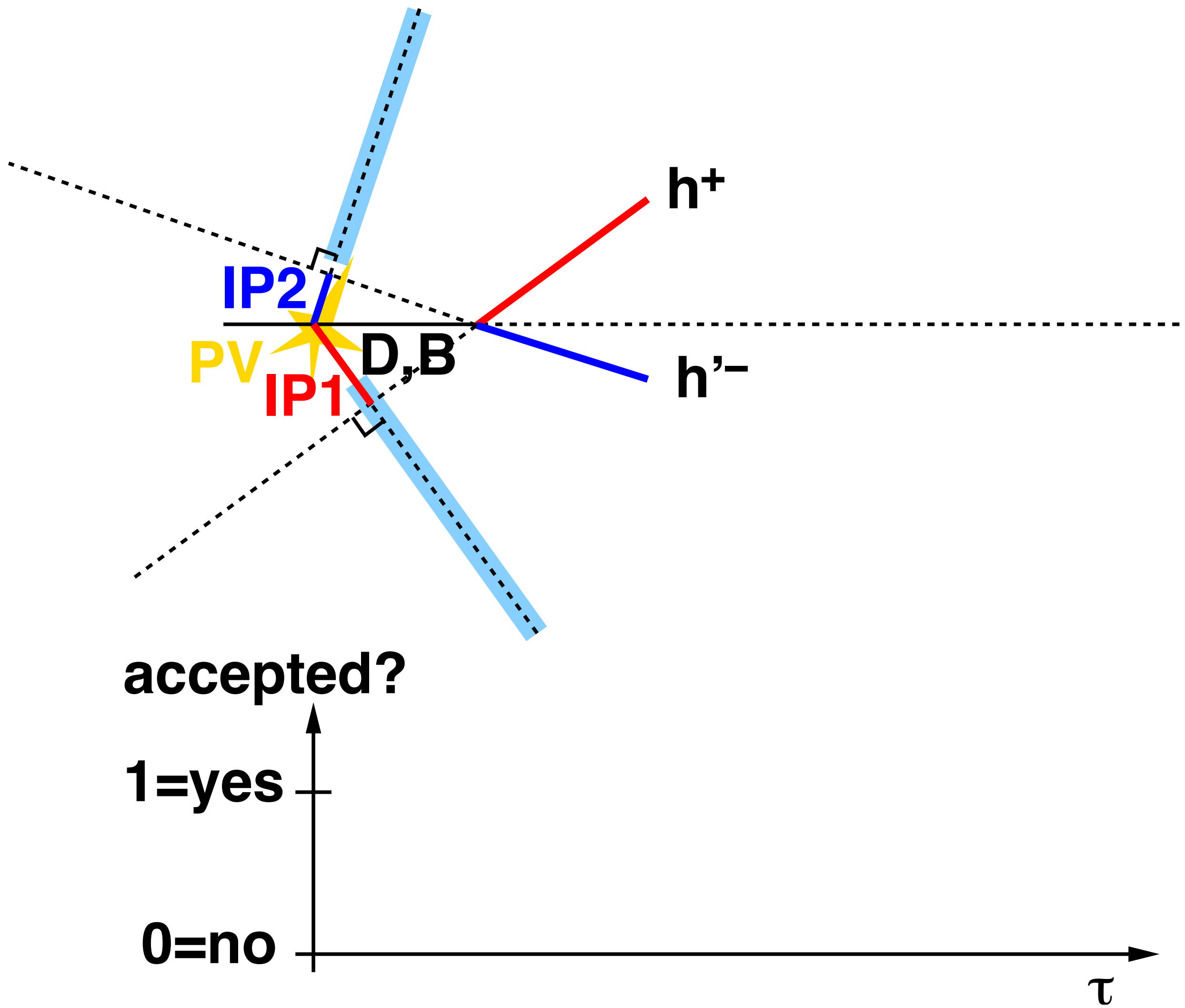
**The shape of this function must be measured if the events are to be used to measure the B lifetime**

**However, we do not want to take this efficiency from simulated events, as we cannot prove that the simulation is accurate at the level of precision required**

**Similarly there are no large enough control samples : we must obtain the efficiency from the signal itself.**



# Swimming in action

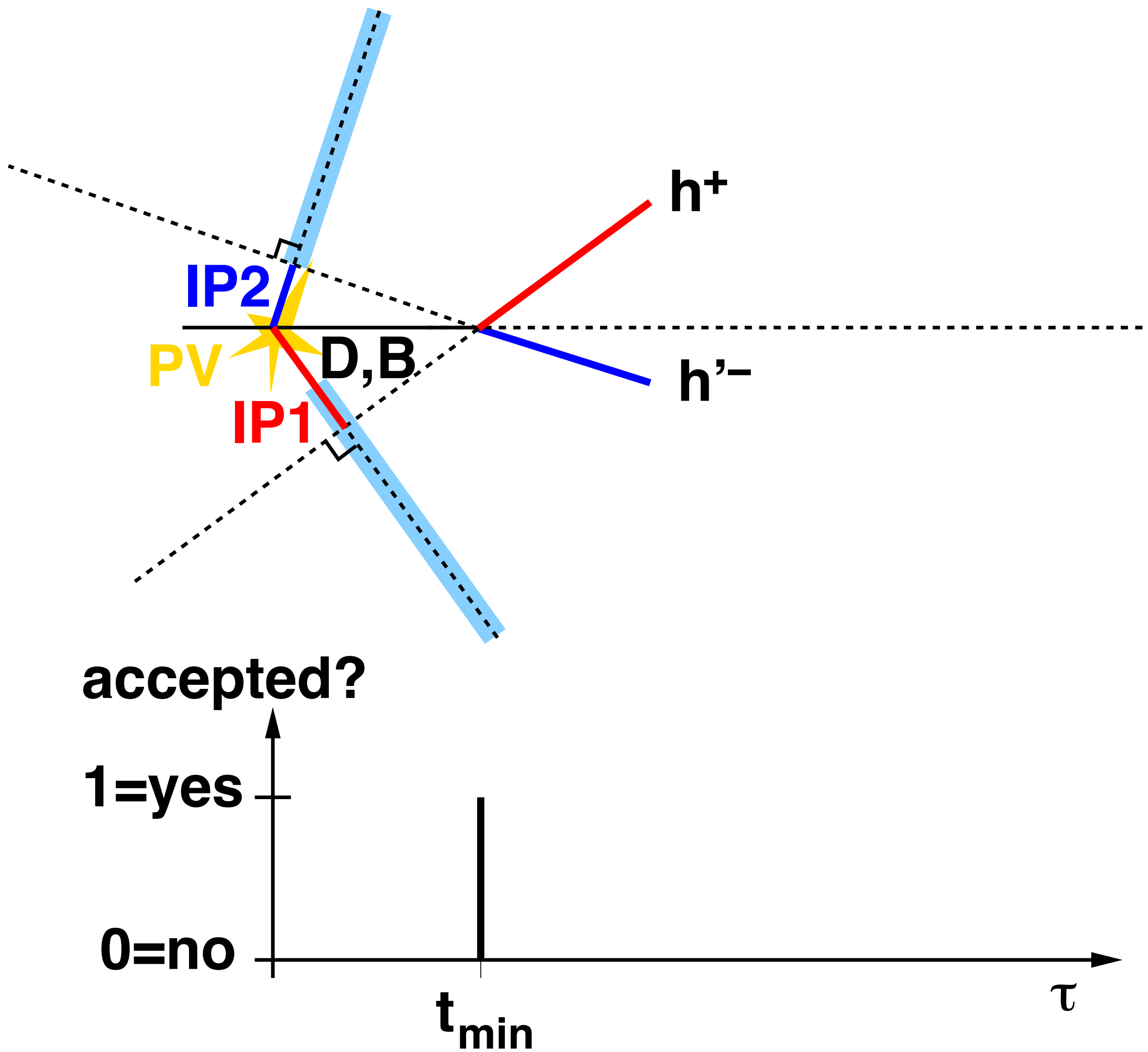



Because we can reproduce the trigger decisions offline, we can measure lifetime biases in a data driven way offline

Get an event-by-event acceptance by replaying the trigger decision for the full range of possible B/D lifetimes

No trigger emulation needed, correct alignment and detector conditions automatically taken into account.

# Swimming in action

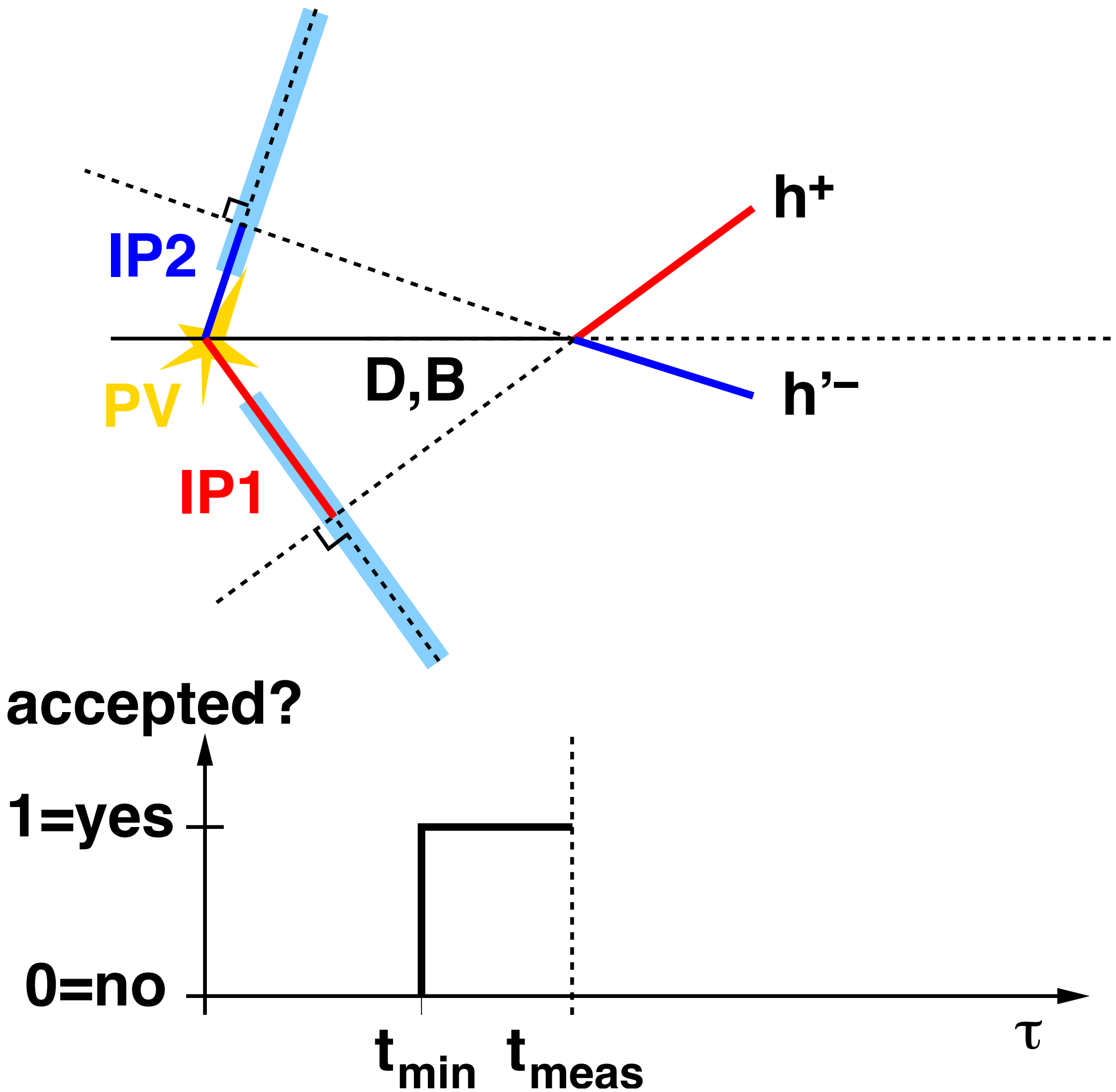



Because we can reproduce the trigger decisions offline, we can measure lifetime biases in a data driven way offline

Get an event-by-event acceptance by replaying the trigger decision for the full range of possible B/D lifetimes

No trigger emulation needed, correct alignment and detector conditions automatically taken into account.

# Swimming in action

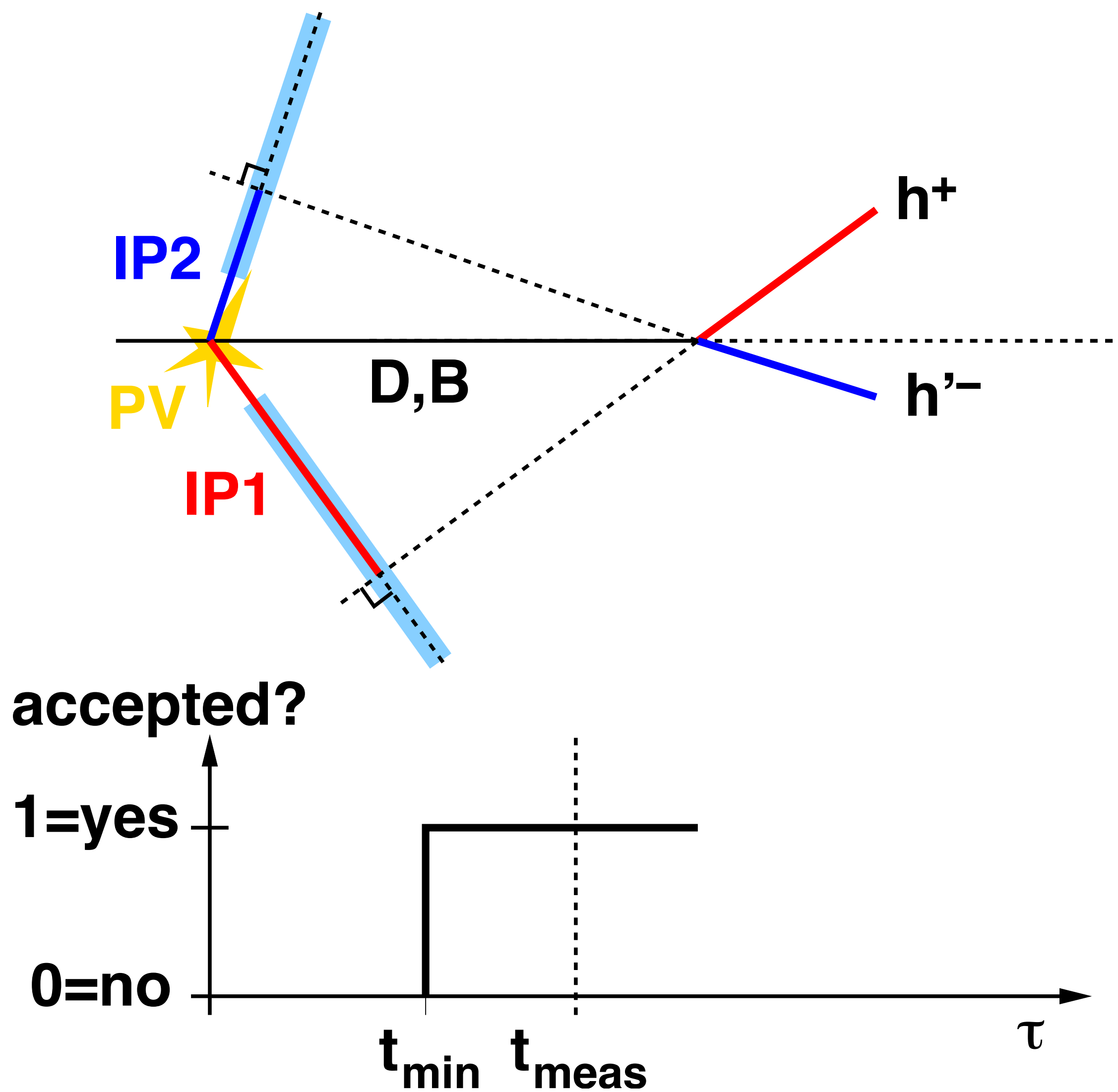



**Because we can reproduce the trigger decisions offline, we can measure lifetime biases in a data driven way offline**

**Get an event-by-event acceptance by replaying the trigger decision for the full range of possible B/D lifetimes**

**No trigger emulation needed, correct alignment and detector conditions automatically taken into account.**

# Swimming in action

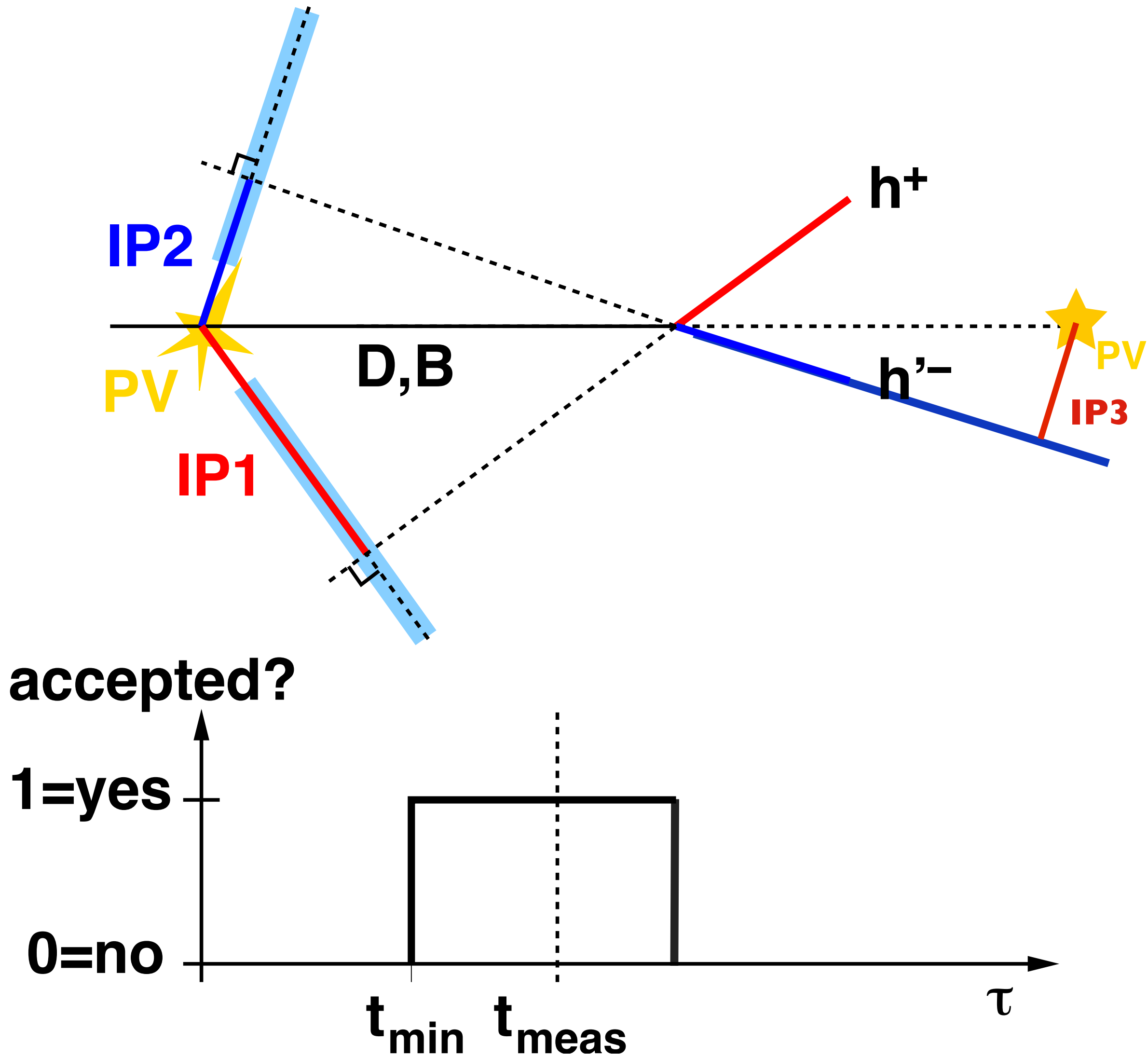



**Because we can reproduce the trigger decisions offline, we can measure lifetime biases in a data driven way offline**

**Get an event-by-event acceptance by replaying the trigger decision for the full range of possible B/D lifetimes**

**No trigger emulation needed, correct alignment and detector conditions automatically taken into account.**

# Swimming in action




**Because we can reproduce the trigger decisions offline, we can measure lifetime biases in a data driven way offline**

**Get an event-by-event acceptance by replaying the trigger decision for the full range of possible B/D lifetimes**

**No trigger emulation needed, correct alignment and detector conditions automatically taken into account.**

# Finding the turning points

The general acceptance function is a series of top hat and/or step functions

The point at which the acceptance changes from 0 to 1 or vice versa is known as a "turning point"

We do not know the positions or number of turning points in each event

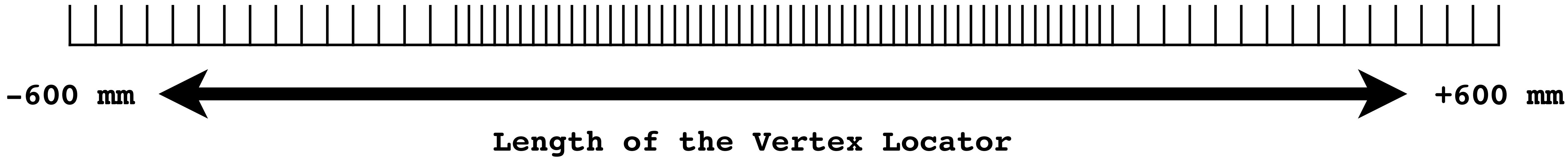
However we do know the actual decay time at which the parent meson was reconstructed, and this must lie within an "accept" interval by definition

Two stage search :

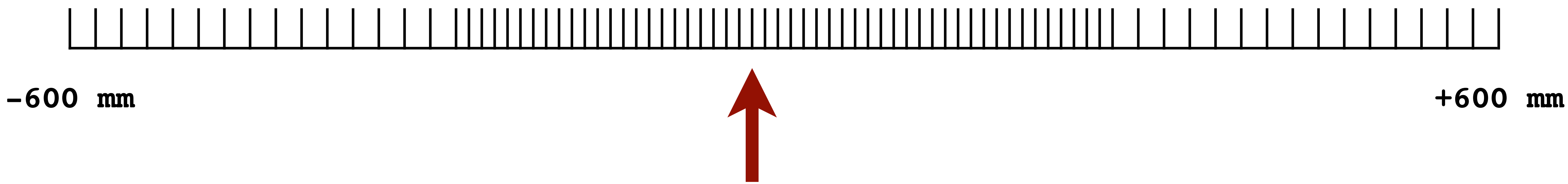
1) Replay the decision varying the decay time by a fixed amount and find all turning points with a coarse precision

2) Refine the position of each turning point by reducing the swimming granularity and repeating the turning point search in the region around each coarsely determined turning point

# Swimming granularity

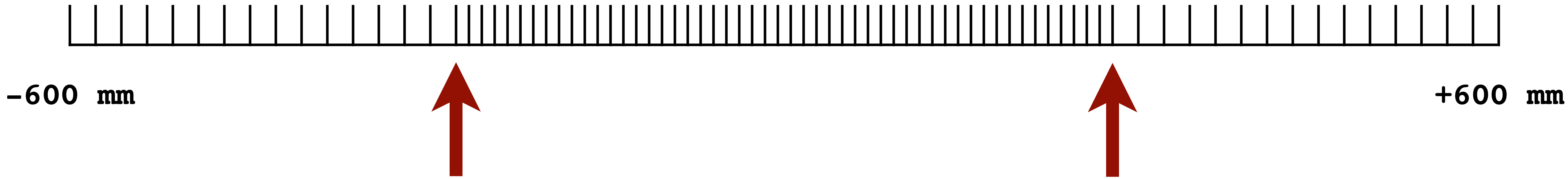


# Swimming granularity



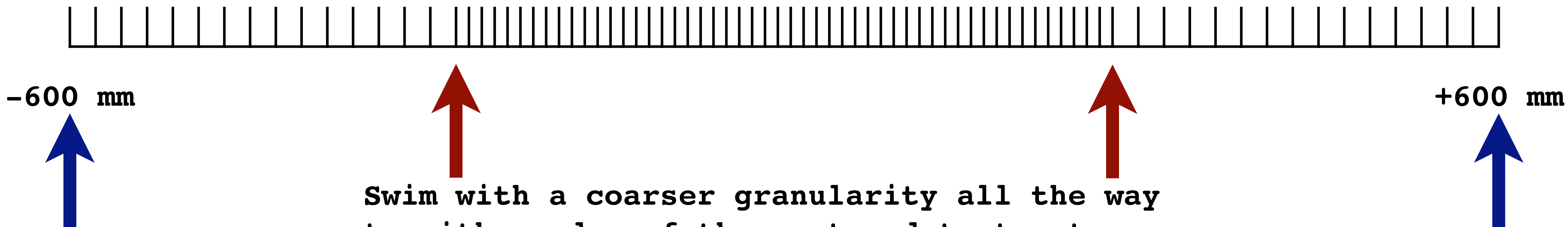

**Always swim the "zero" point, at which the particle actually decayed. This decision must always be 1 and provides a good check that the trigger is being replayed correctly**

# Swimming granularity



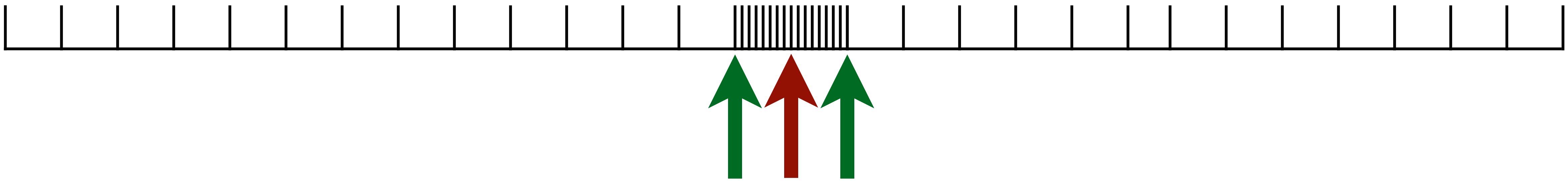

**Swim with a finer initial granularity closer to the measured decay time, where we expect the turning points to be**

# Swimming granularity




**Swim with a coarser granularity all the way to either edge of the vertex detector to look for effects of the detector geometry and finite size on the acceptance**

# Swimming refinement



**Once a turning point is found, swim with a finer granularity around it to refine its position.**

**Final granularity chosen to measure the turning points with ~10 micron precision, comparable to the propertime resolution; balance performance and speed.**

# Large scale implementation

Swimming an event takes ~50 HS06.s

- On average each event is replayed ~150 times

The amount of events to be swum varies, but in total it was around 100 million for the 2011 LHCb dataset (about 1% of all events written to disk)

- Impractical to implement this as a series of jobs by individual users

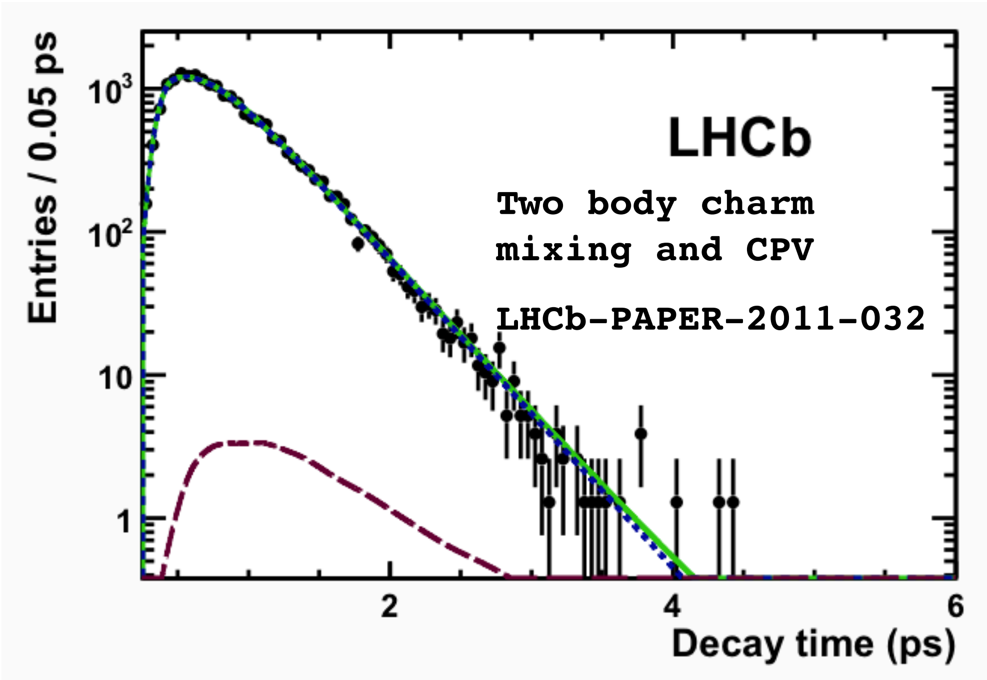
As a result the swimming was implemented in an automated way through the DIRAC software used to manage all large-scale LHCb productions

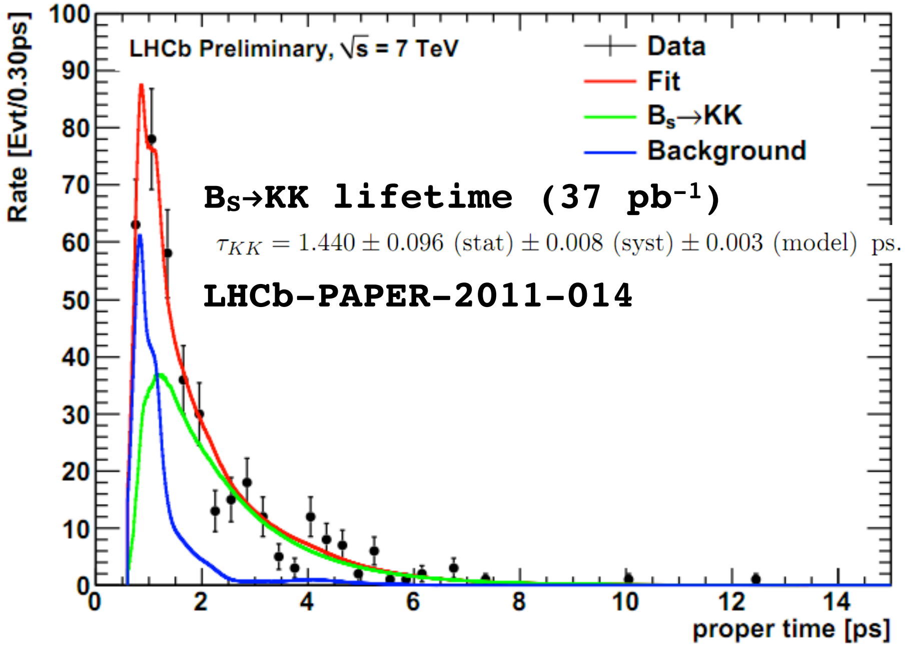
- This also required modifications in the way jobs were assigned input files, because each file had to be assigned to the correct job based on the trigger software version, trigger configuration key, and database tags associated with it



See also "LHCbDIRAC: distributed computing in LHCb"
Poster id 145 by Federico STAGNI on 22 May 2012 from 13:30 to 18:15 (track 3)

# Outlook and conclusions




LHCb has already published several papers using the swimming technique and more are on the way

Reproducible software trigger is at the core of this

- Reproducibility presents long term challenges as compilers, hardware, and the underlying codebase evolve
- Have already discovered reproducibility problems at the $10^{-4}$ level linked to floating point precision in different compilers, to be seen if this has an impact on the physics