UNDERSTANDING ND-GAR'S KALMAN FILTER: TESTING ON A TOY MONTE CARLO MODEL

UNIVERSITY OF OXFORD

Presents:
Federico Battisti

OUTLINE

- This presentation will cover a fairly complete description of our current Kalman Filter based reconstruction algorithm together with some early findings from a Toy Monte Carlo study developed to study the performance of said algorithm

- The presentation will be divided into 4 main sections:

  - Description of what Kinematic Fitting is
  - Description of what a Kalman Filter is
  - Description of our own algorithm (T. Junk, DUNE-Doc-13933 https://docs.dunescience.org/cgi-bin/private/ShowDocument?docid=13933), which is a convolution of a Kalman Filter and a Kinematic Fitting algorithm
  - Early findings from my Toy Monte Carlo study

KINEMATIC FITTING

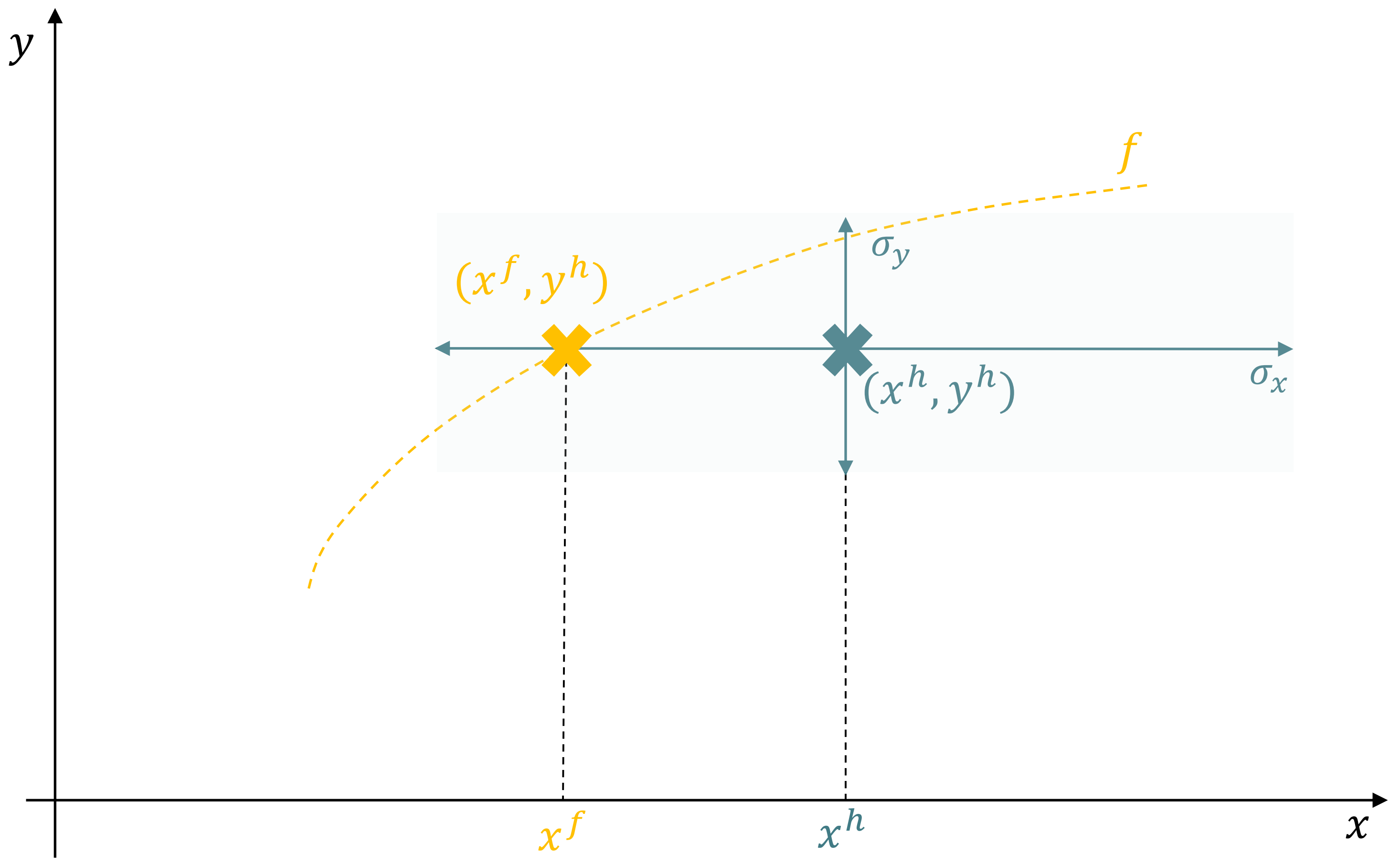
KINEMATIC FITTING

- Kinematic fitting is an algorithm that updates a measurement within error in the presence of a model constraint.

- An example of this sort of fit can be found in https://inspirehep.net/literature/1780811 where kinematic fitting has been used in the context of neutrino-induced charged-current (CC) π_0 production on carbon, to improve the neutral pion momentum reconstruction

- For more insights you can consult:
  - https://www.phys.ufl.edu/~avery/fitting/kinfit_talk1.pdf
  - http://www-hermes.desy.de/notes/pub/TALK/yaschenk.ColloqGlasgow.pdf

KINEMATIC FITTING



- A new value for the x coordinate x^f is calculated as an alternative to the measured value x^h
- x^f is taken such that the weighted distance (in terms of σ_x and σ_y) between the model f and the measurement (x^h, y^h) is minimal.
- In the limit case where σ_y is very large, this is equivalent to using the measured x^h
- In the limit case where σ_x is very large, this is equivalent to using the x from the model (this is the case shown in the diagram)

KALMAN FILTER

KALMAN FILTER IN GENERAL

- A Kalman filter is an iterative algorithm which uses a system's physical laws of motion, known control inputs and multiple sequential measurements to form an estimate of the system's varying quantities
- At each step of the iteration an estimate of the state of the system is produced as a weighted average of the system's predicted state and of the new measurement. The weights are calculated from the covariance.
- The models for state transition and measurement can be written as:

STATE VECTOR	$s_k = f(s_{k-1}, X_{k-1})$
MEASUREMENT VECTOR	m_k^h

- Where f is the function of the previous state, s_{k-1} , and the free parameter, X_{k-1}, that provides the current state s_k.
- Note that the measurement vector m_k^h and the state vector s_k do not necessarily have the same number of components

KALMAN FILTER IN GENERAL

1. Make a priori predictions for the current step's state and covariance matrix using the a posteriori best estimate of the previous step (i.e. updated using measurement)

STATE VECTOR $s_k^- = f(s_{k-1}^+, X_{k-1})$

COVARIANCE MATRIX $P_k^- = F_{k-1} P_{k-1}^+ F_{k-1}^T + Q$

$$\boxed{F_{k-1} = \left.\frac{\partial f}{\partial s}\right|_{s_{k-1}^+, X_{k-1}}}$$

JACOBIAN

$$\boxed{Q}$$

PROCESS NOISE COVARIANCE

Note: In the first iteration step we use step 0 estimates for the state vector and the covariance matrix (s_0, P_0), which can be made very roughly

KALMAN FILTER IN GENERAL

2. Calculate the measurement residual and the Kalman Gain

RESIDUAL	$\tilde{y}_k = m_k^h - H(s_k^-)$
KALMAN GAIN	$K_k = P_k^- H^T (R + H P_k^- H^T)^{-1}$

$\boxed{R}$

MEASUREMENT NOISE COVARIANCE

$\boxed{H}$

CONVERSION MATRIX

3. Update the estimate

STATE VECTOR	$s_k^+ = s_k^- + K_k \tilde{y}$
COVARIANCE MATRIX	$P_k^+ = (1 - K_k H) P_k^-$

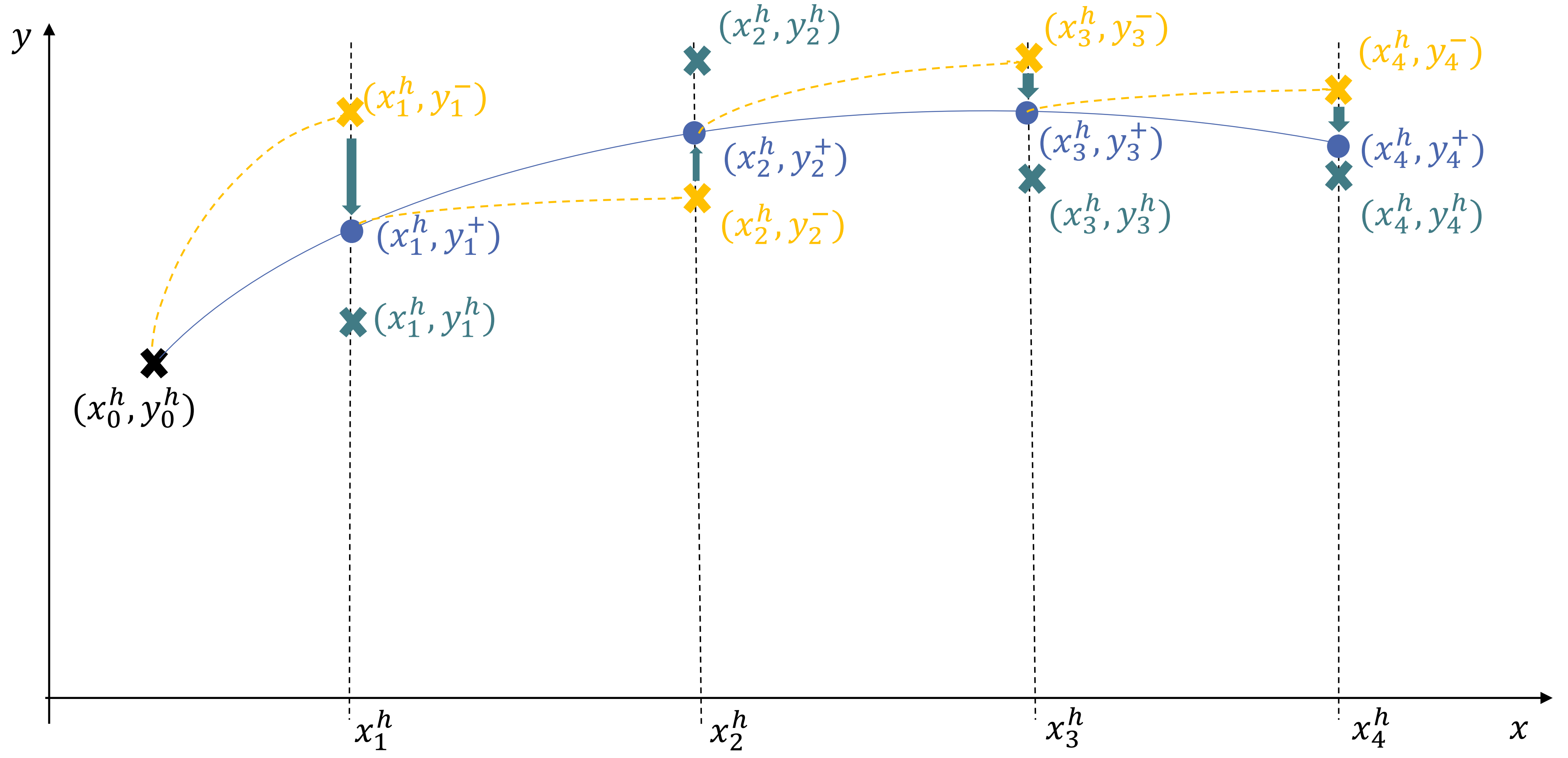
Note: in the case where R is a null matrix $s_k^+ = s_k^h$ and $P_k^+ = 0$

Note: the conversion matrix is needed to make the dimensions of vectors and matrixes turn out right. For exemple if s_k^h is a 2-D vector and s_k^- is 5-D, then H would be a 2×5 matrix:

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

FEDERICO BATTISTI

KALMAN FILTER IN GENERAL

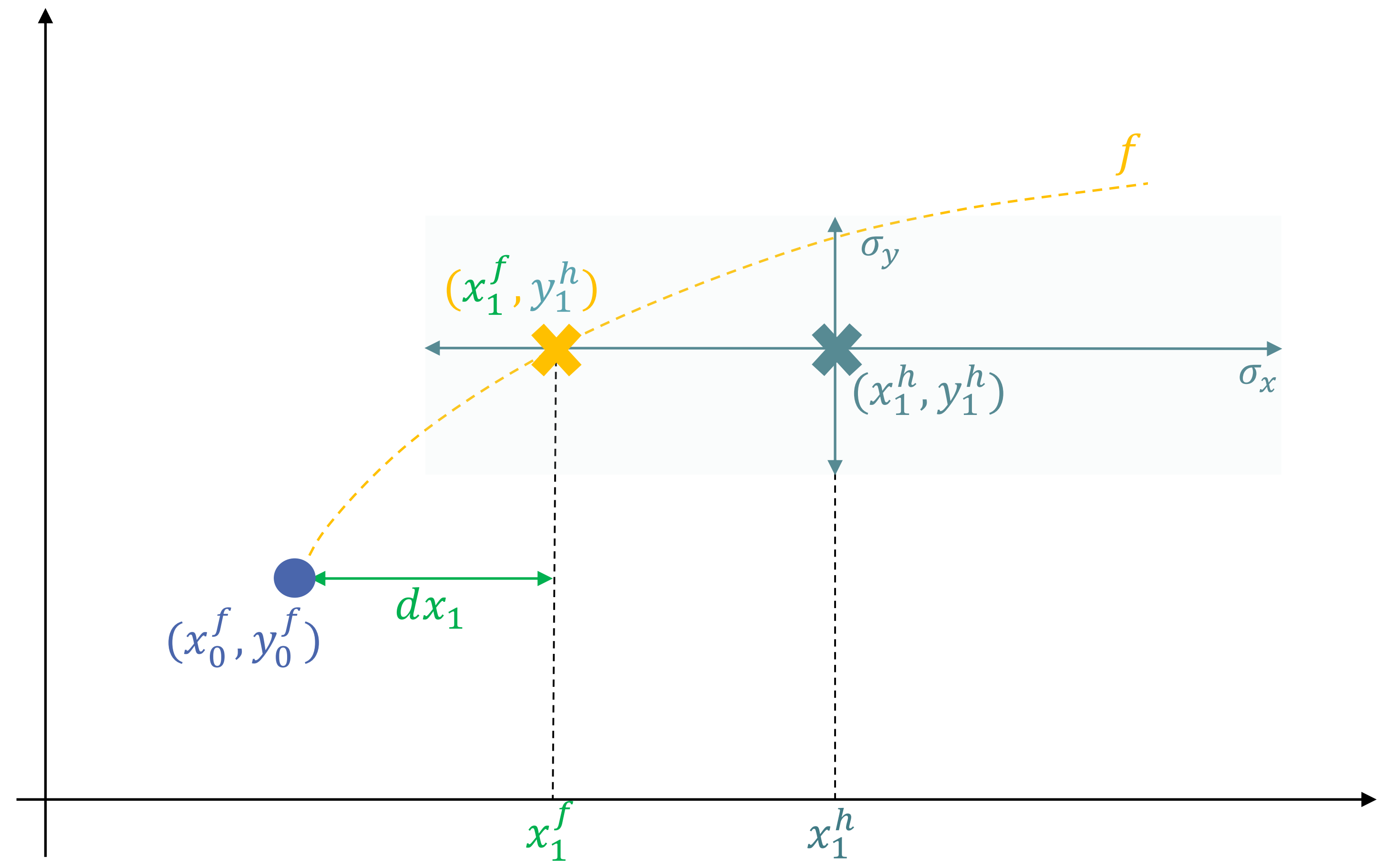


FEDERICO BATTISTI

OUR RECONSTRUCTION ALGORITHM™

T. Junk, DUNE-Doc-13933: https://docs.dunescience.org/cgi-bin/private/ShowDocument?docid=13933

KINEMATIC FITTING



- In our case we use kinematic fitting to determine the free parameter step dx_k
- dx_k is taken such that a weighted distance Δ (in terms of σ_x and $\sigma_y = \sigma_z = \sigma_{yz}$) between the model f and the measurement (x_k^h, y_k^h) is minimal.
- In the limit case where σ_{yz} is very large, this is equivalent to using the measured x^h
- In the limit case where σ_x is very large, this is equivalent to using the x from the model (this is the case shown in the diagram)

KINEMATIC FITTING

- The quantity minimize in our kinematic fitting is the weighted distance Δ with $(\sigma_x, \sigma_y) = (0.5cm, 1cm)$:

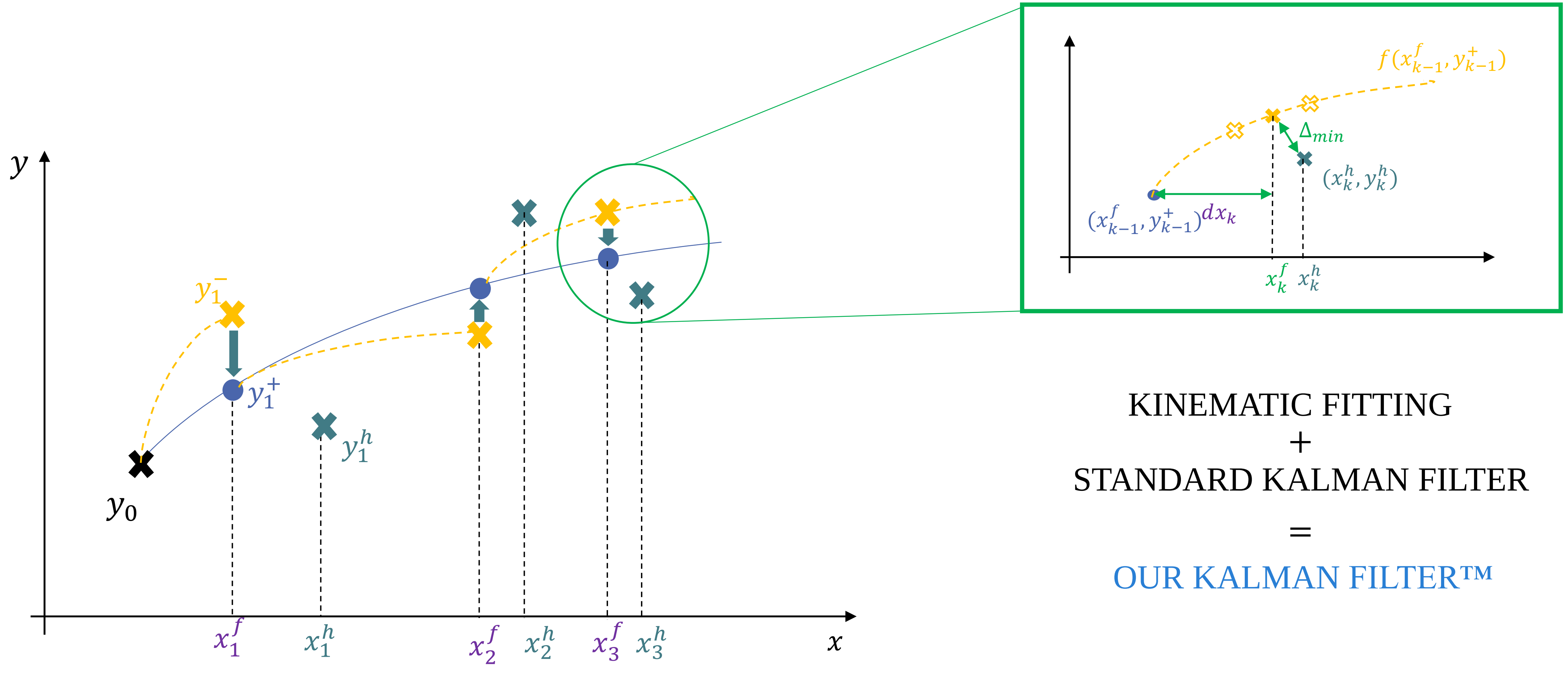
$$\Delta(dx_k; x_k^h, y_k^h, z_k^h, x_{k-1}^f, y_{k-1}^+, z_{k-1}^+, \lambda_{k-1}^+, \phi_{k-1}^+) = \left[\frac{(x_k^h - x_k^f)}{\sigma_x}\right]^2 + \left[\frac{(y_k^h - y_k^-)}{\sigma_{yz}}\right]^2 + \left[\frac{(z_k^h - z_k^-)}{\sigma_{yz}}\right]^2$$

$$= \left[\frac{(x_k^h - x_{k-1}^f - dx_k)}{\sigma_x}\right]^2 + \left[\frac{(y_k^h - y_{k-1}^+ - dx_k \times \cot\lambda_{k-1} \times \sin\phi_{k-1}^+)}{\sigma_{yz}}\right]^2 + \left[\frac{(z_k^h - z_{k-1}^+ - dx_k \times \cot\lambda_{k-1} \times \cos\phi_{k-1}^+)}{\sigma_{yz}}\right]^2$$

- We determine dx_k by imposing that the derivative of Δ is equal to zero

$$\frac{\mathrm{d}\Delta}{\mathrm{d}(dx_k)} = 0 \quad \Rightarrow \quad dx_k = \frac{\left\{\frac{\cot\lambda_{k-1}^+}{\sigma_{yz}^2}\left[(y_k^h - y_{k-1}^+)\sin\phi_{k-1}^+ + (z_k^h - z_{k-1}^+)\cos\phi_{k-1}^+\right]\right\} + \frac{(x_k^h - x_{k-1}^f)}{\sigma_x^2}}{\cot^2\lambda_{k-1}^+ / \sigma_{yz}^2 + 1/\sigma_x^2}$$

OUR KALMAN FILTER: VISUALIZATION



KINEMATIC FITTING

+

STANDARD KALMAN FILTER

=

OUR KALMAN FILTER™

TOY MONTE CARLO STUDY

TOY MONTE CARLO OUTLINE

- We constructed a Toy Monte Carlo study as a sanity check for our current Kalman filter™

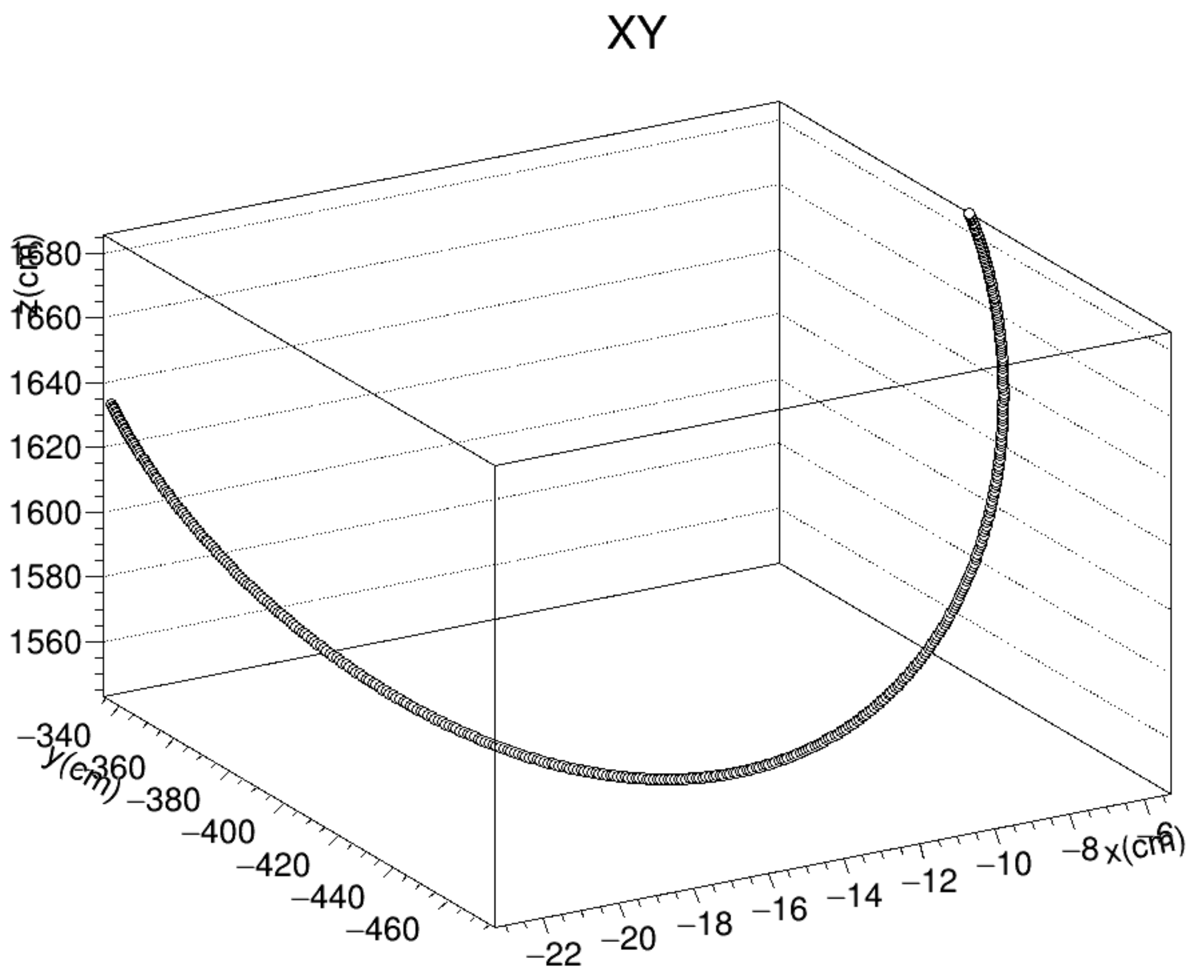
- The study will eventually develop in multiple steps:

  - Use a perfect helix model with fixed step length to check the a priori model
  - Use a perfect helix model with randomized step length (but no smearing)
  - Add Gaussian error on xyz (1D at a time, then 2D at a time, then fully 3D) to validate covariance and calibrate χ^2
  - Use Cauchy smearing instead to test sensitivity to noise model
  - Simulate Energy Loss etc..

- The current presentation will cover the first two steps in the study

TOY MONTE CARLO

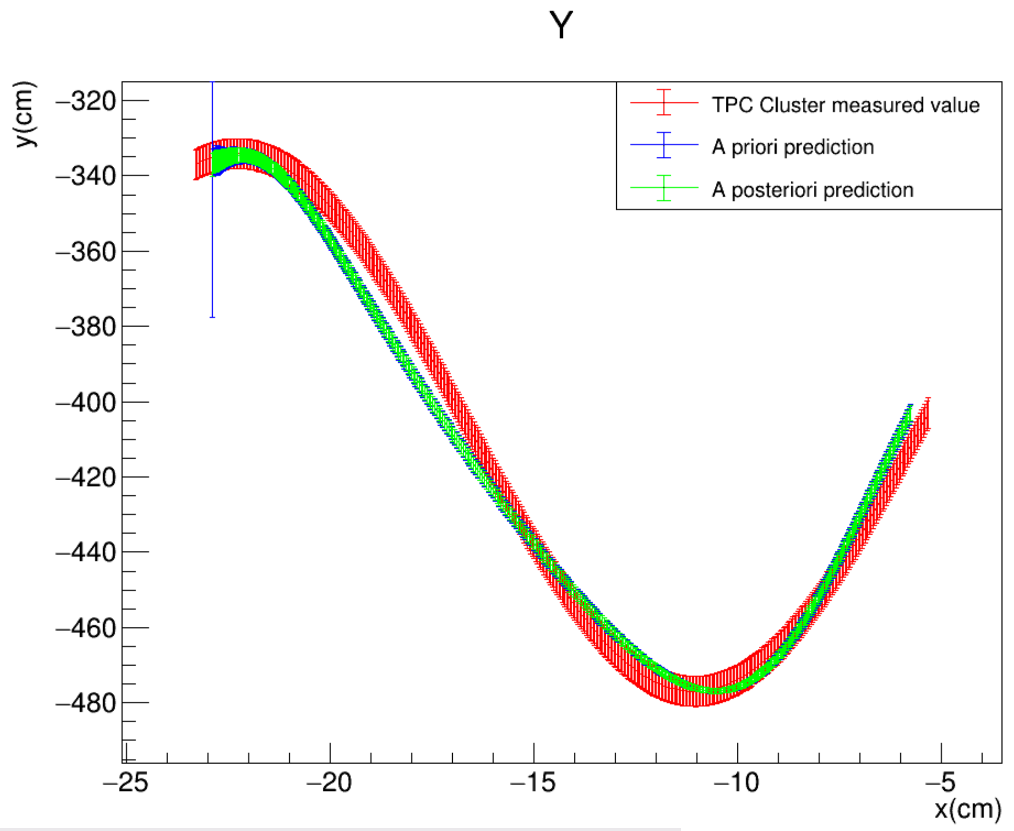
- We applied our Kalman Filter™ to ideal measurements following perfectly an helix with initial coordinates: $s_0^T =$ $(y_0 \quad z_0 \quad 1/r_0 \quad \phi_0 \quad \lambda_0) =$ $(y_0^h \quad z_0^h \quad 0.014\ cm^{-1} \quad 6\ \text{rad} \quad -0.05\ \text{rad})$ and the free parameter $x_0 = x_0^h$ and the step $dx = 0.04$ cm

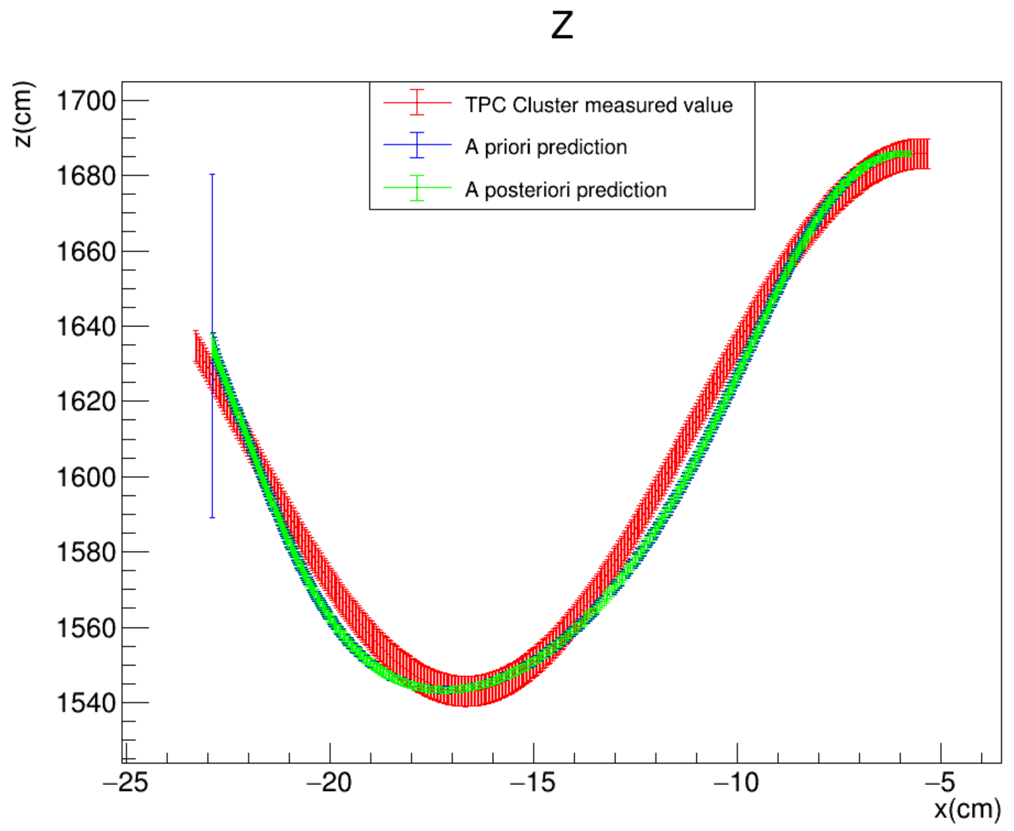


Note: even though we have unsmeared idealized measurement for this first study we kept the 'R (noise) matrix' unmodified with uncertainties $\Sigma_{xy} = 4cm$

TOY MONTE CARLO

- Plots describing the evolution of the estimate of the measured quantities (y, z) as a function of the free parameter x for the perfect helix. Note that the TPC points have on the x coordinate the measured value, while for the estimates we use the estimated x from the kinematic fitting (i.e. $x_k = x_0 + k \times dx_k$)





Note: the 'TPC Cluster measured values' are from a Toy MC, so the red error bars are the dummy values from the R matrix

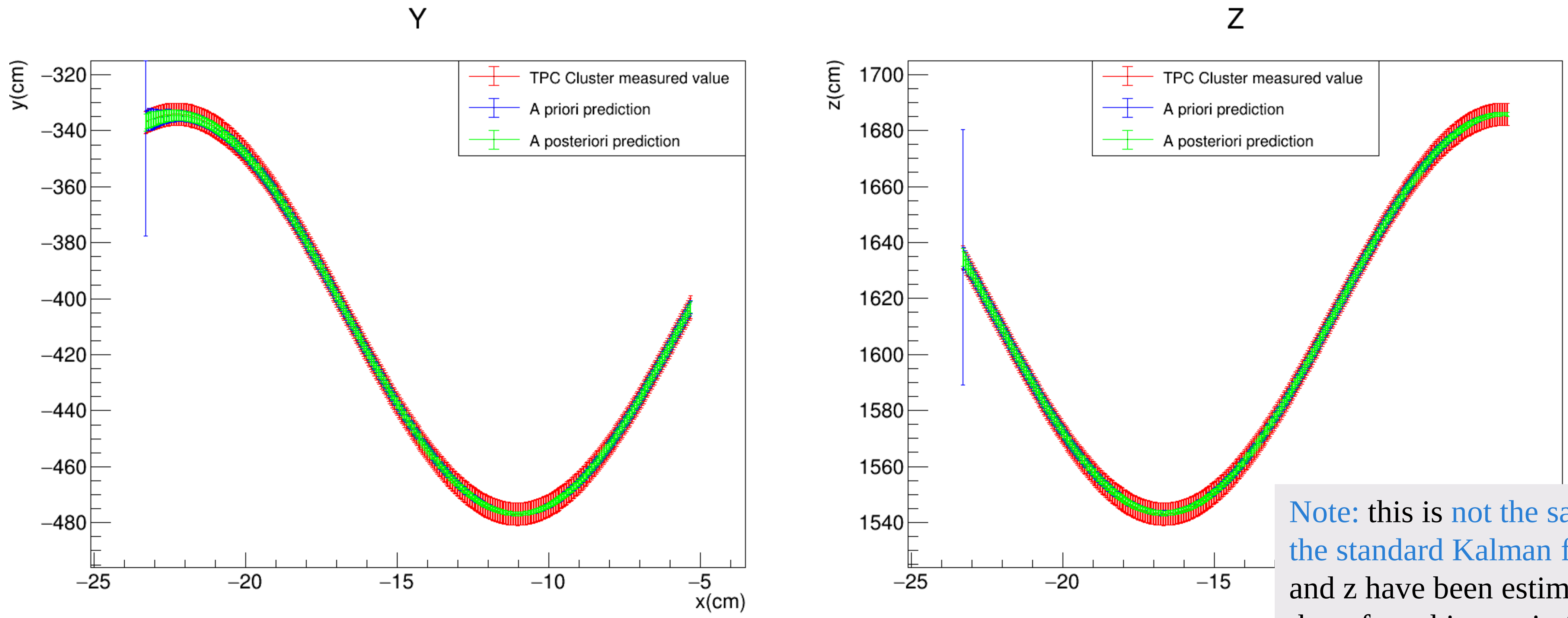
FILTER PROPAGATION DIRECTION

KFTM: $(\sigma_x, \sigma_{yz}) = (0.5cm, 1cm)$

FEDERICO BATTISTI

TOY MONTE CARLO

- We now replot the results from the previous slide, all identical except replacing the x coordinate by x^h instead of x^f. The plots show that the predicted y and z components coincide with the measured ones



Note: this is not the same as using the standard Kalman filter since y and z have been estimated using the x from kinematic fitting, as in the previous slide

KF™: $(\sigma_x, \sigma_{yz}) = (0.5cm, 1cm)$

FILTER PROPAGATION DIRECTION

TOY MONTE CARLO: UNDERSTANDING STEP DETERMINATION

- With the current sigma values ($\sigma_x = 0.5cm$ and $\sigma_{yz} = 1cm$) the ratio σ_x/σ_{yz} is large enough that the evolution of y and z is essentially decoupled from the evolution of the free parameter x.
- This gives us $y_k^- \simeq y_k^h$ and $z_k^- \simeq z_k^h$ and an x^f that is updated completely disregarding the measured x^h values
- A way to see this is to divide the dx_k update formula into two parts: one that depends on the evolution of the x free parameter, and one on y and z:

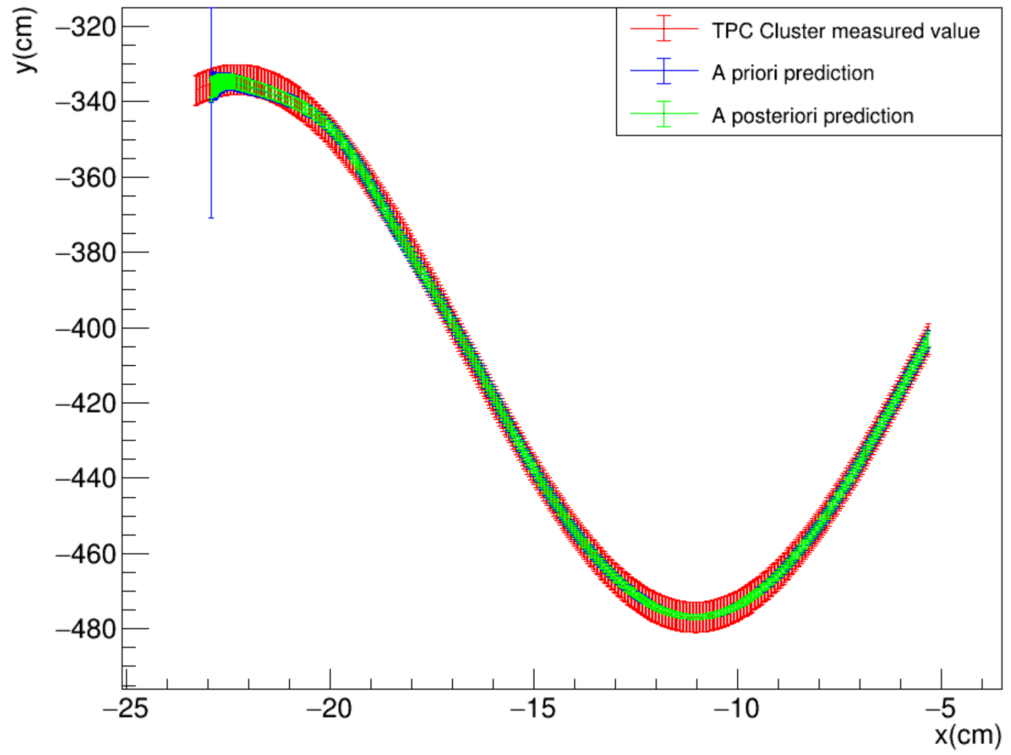
$$dx_k = \frac{\frac{\cot\lambda_{k-1}^+}{\sigma_{yz}^2}\left[\left(y_k^h - y_{k-1}^+\right)\sin\phi_{k-1}^+ + \left(z_k^h - z_{k-1}^+\right)\cos\phi_{k-1}^+\right]}{\cot^2\lambda_{k-1}^+ / \sigma_{yz}^2 + 1/\sigma_x^2} + \frac{\frac{\left(x_k^h - x_{k-1}^f\right)}{\sigma_x^2}}{\cot^2\lambda_{k-1}^+ / \sigma_{yz}^2 + 1/\sigma_x^2} = dx_{yz} + dx_x$$

- With the current sigma values ($\sigma_x = 0.5cm$ and $\sigma_{yz} = 1cm$) dx_{yz} completely dominates, being often 2 or 3 orders of magnitude larger than dx_x: this gives us fairly predictions for y and z very close to the measured values, but completely wrong ones for x, becuase the fit can never recover from a bad initial estimate for the free parameter

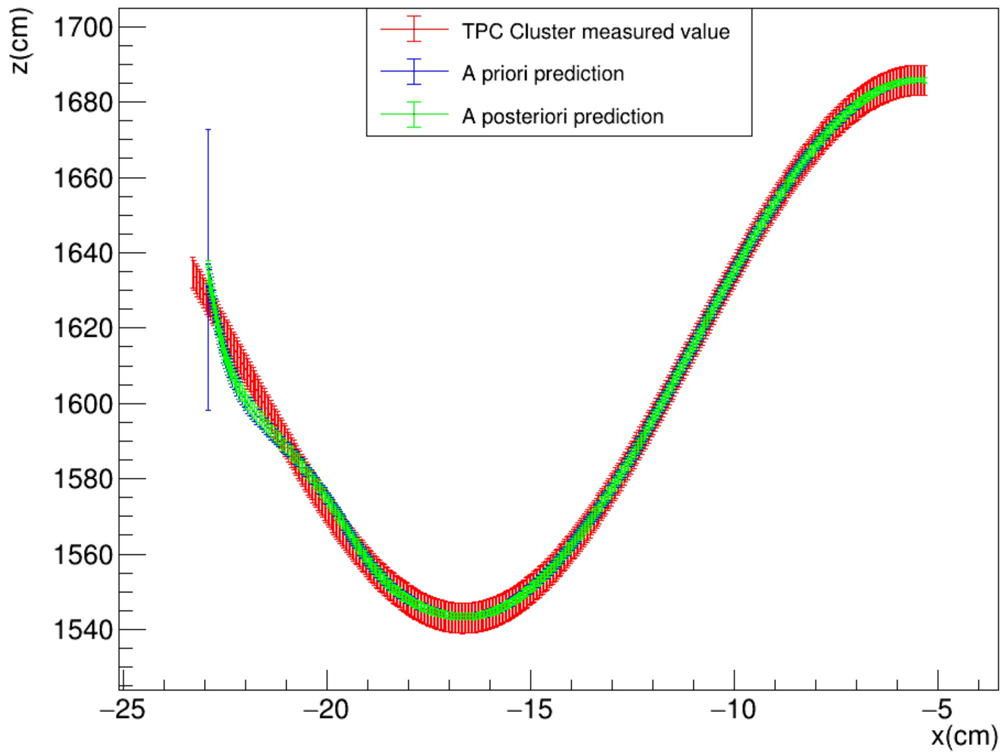
TOY MONTE CARLO: UNDERSTANDING STEP DETERMINATION

- In order for the x parameter to have more weight in the update of dx_k, changed the value of σ_{yz} from 1cm to 4cm, leaving $\sigma_x = 0.5cm$
- The fit initially concentrates on fixing the x prediction until that becomes accurate, and then it focuses on yz, reaching similar levels of precision by the end

Y



Z

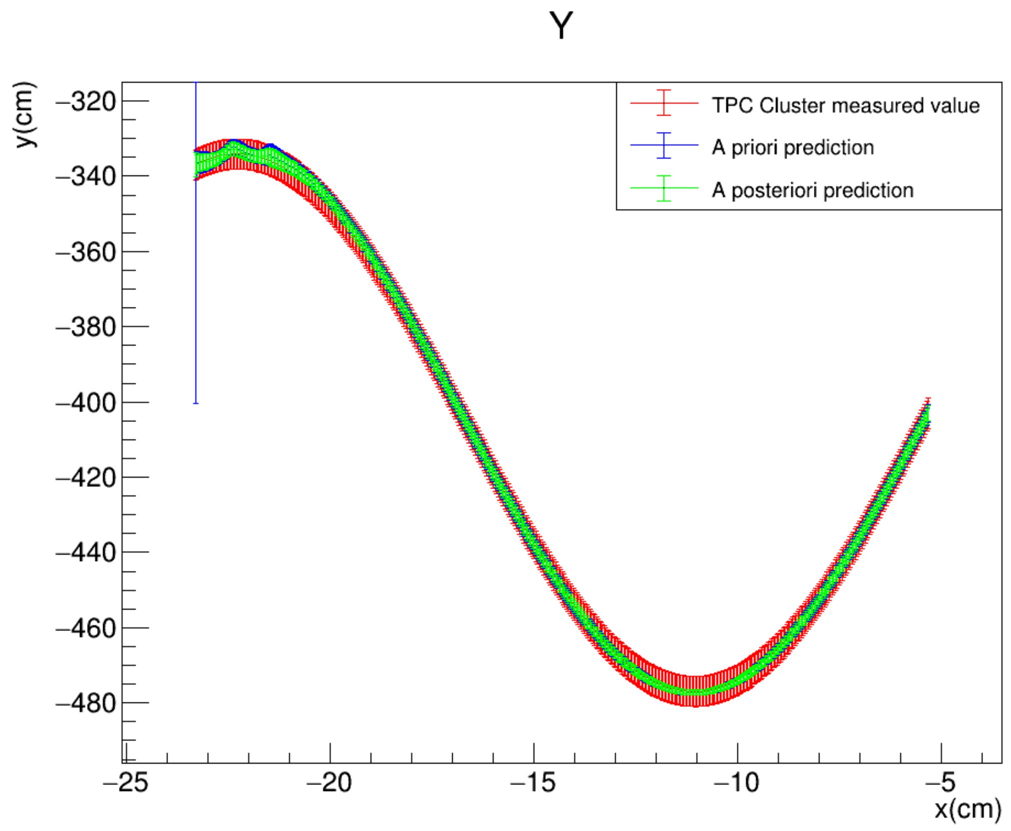


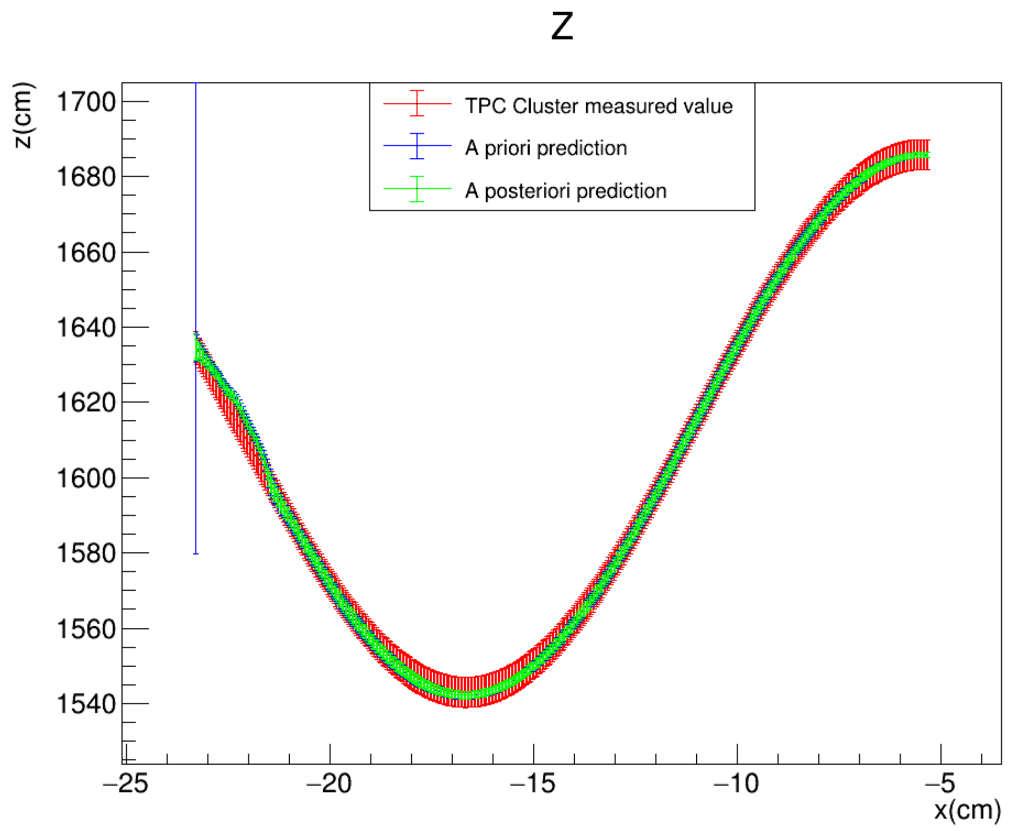
KF™: $(\sigma_x, \sigma_{yz}) = (0.5cm, 4cm)$

FILTER PROPAGATION DIRECTION

TOY MONTE CARLO: UNDERSTANDING STEP DETERMINATION

- We now test an extreme case were σ_{yz}/σ_x is very large: σ_{yz} is changed from 1cm to 1000cm, leaving $\sigma_x = 0.5cm$
- This becomes very close to be using x^h instead of x^f



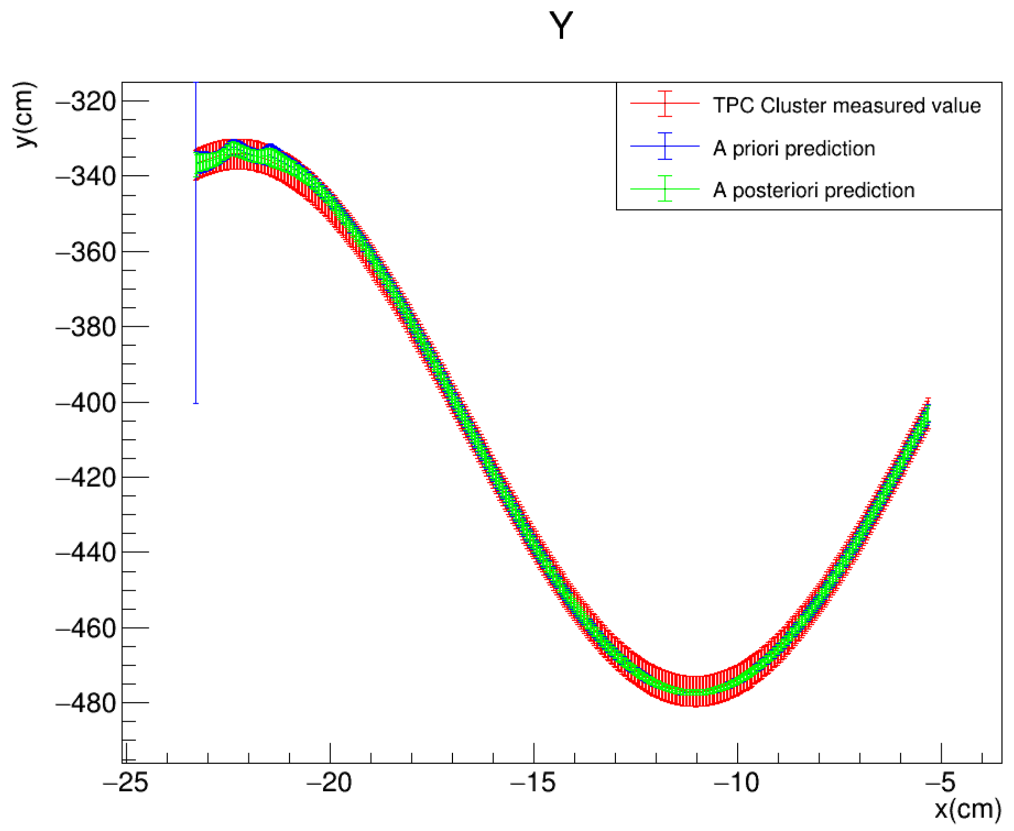


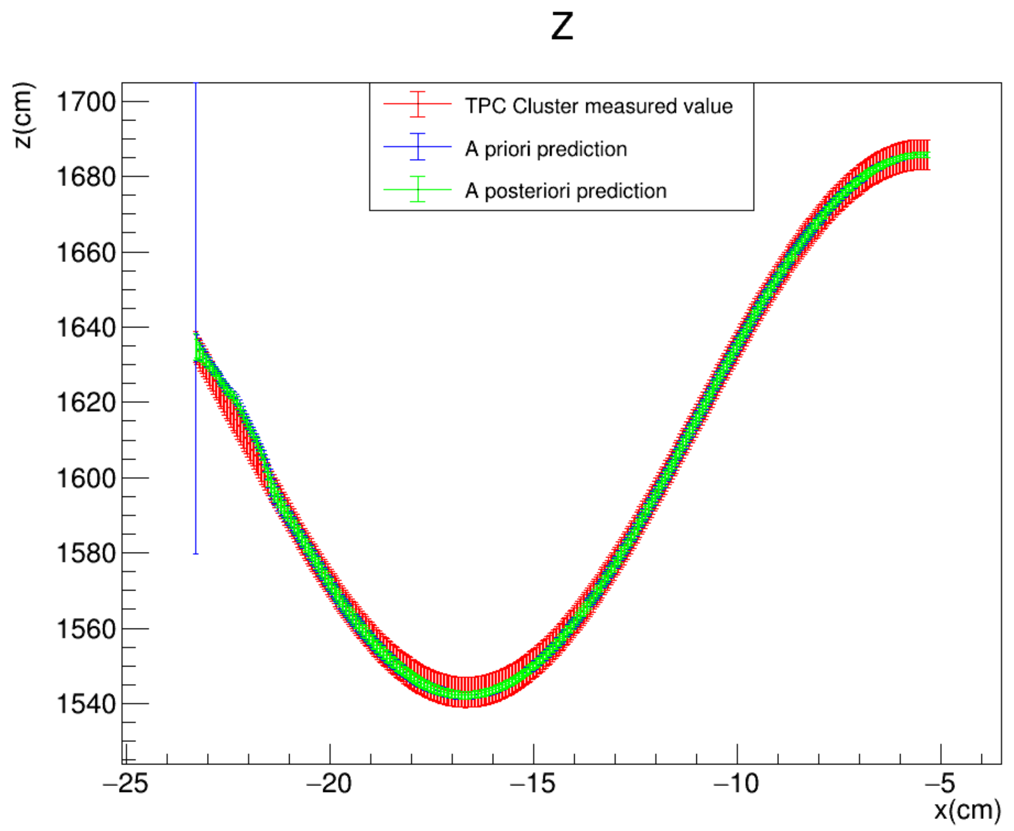
KFTM: $(\sigma_x, \sigma_{yz}) = (0.5cm, 1000cm)$

FILTER PROPAGATION DIRECTION

TOY MONTE CARLO: UNDERSTANDING STEP DETERMINATION

- We try applying a standard Kalman Filter to our Toy Monte Carlo, such that it is identical to our Kalman Filter™ but it avoids the kinematic fitting algorithm completely
- We get essentially the same result as in the last slide: having σ_{yz}/σ_x very large is equivalent to a standard Kalman Filter





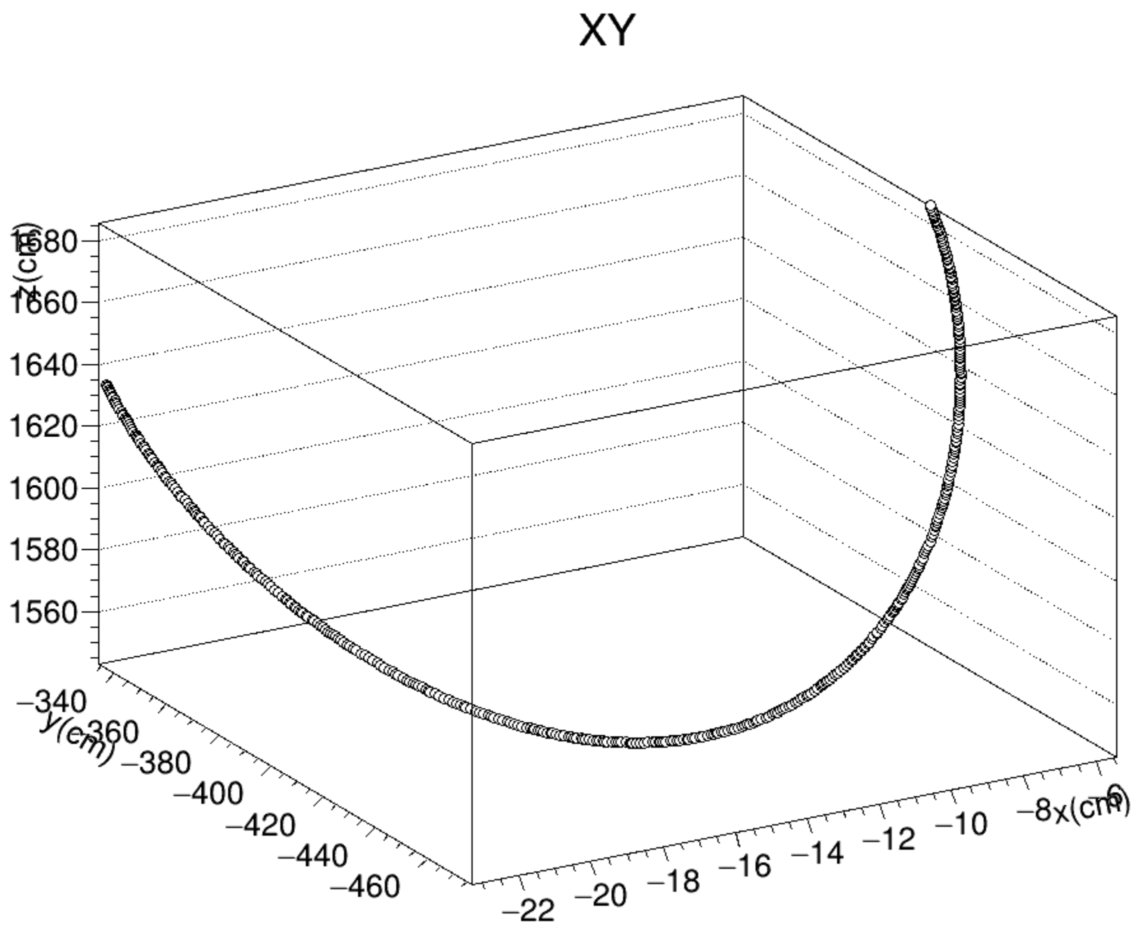
FILTER PROPAGATION DIRECTION

FEDERICO BATTISTI

RANDOMIZED X STEP

- Now we try applying our Kalman Filter to ideal measurements following perfectly an helix with initial coordinates: $x_i^T = (y_i \quad z_i \quad 1/r_i \quad \phi_i \quad \lambda_i) = (y_1^h \quad z_1^h \quad 0.014\ cm^{-1} \quad 6 \text{ rad} \quad -0.05 \text{ rad})$ and the free parameter $x_i = x_1^h$ but with a randomized step dx uniformely distributed between 0.02cm and 0.06cm.

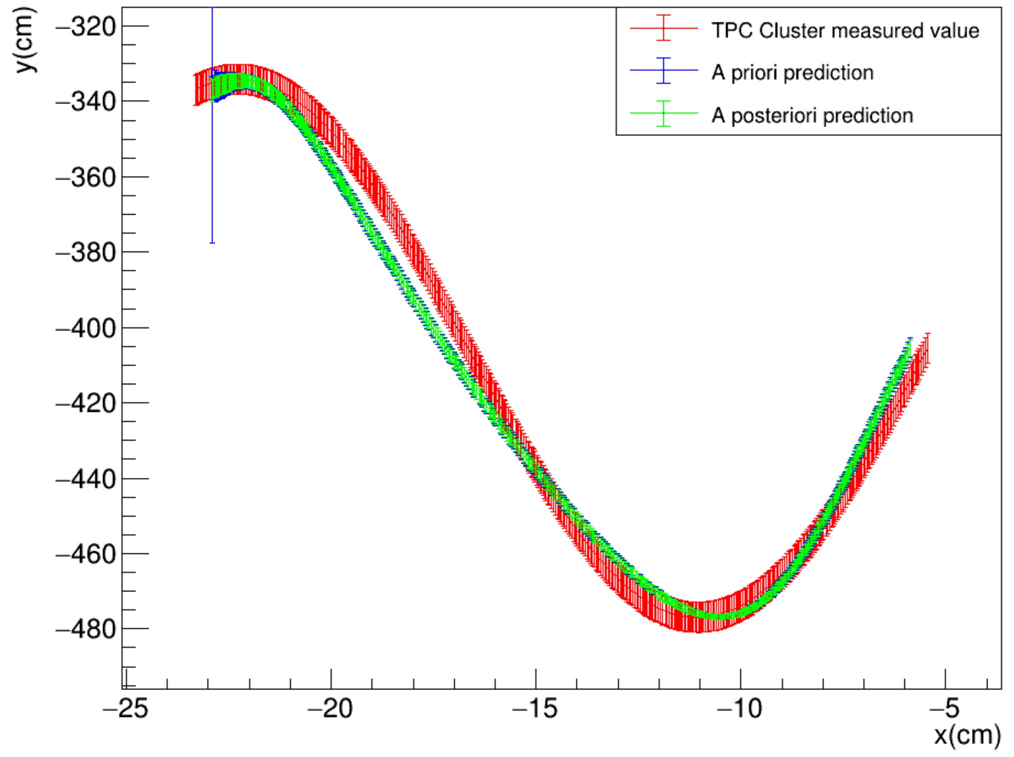
Note: the step progression is randomized, but we are still following a perfect helix i.e. No smearing on the xyz coordinates is involved



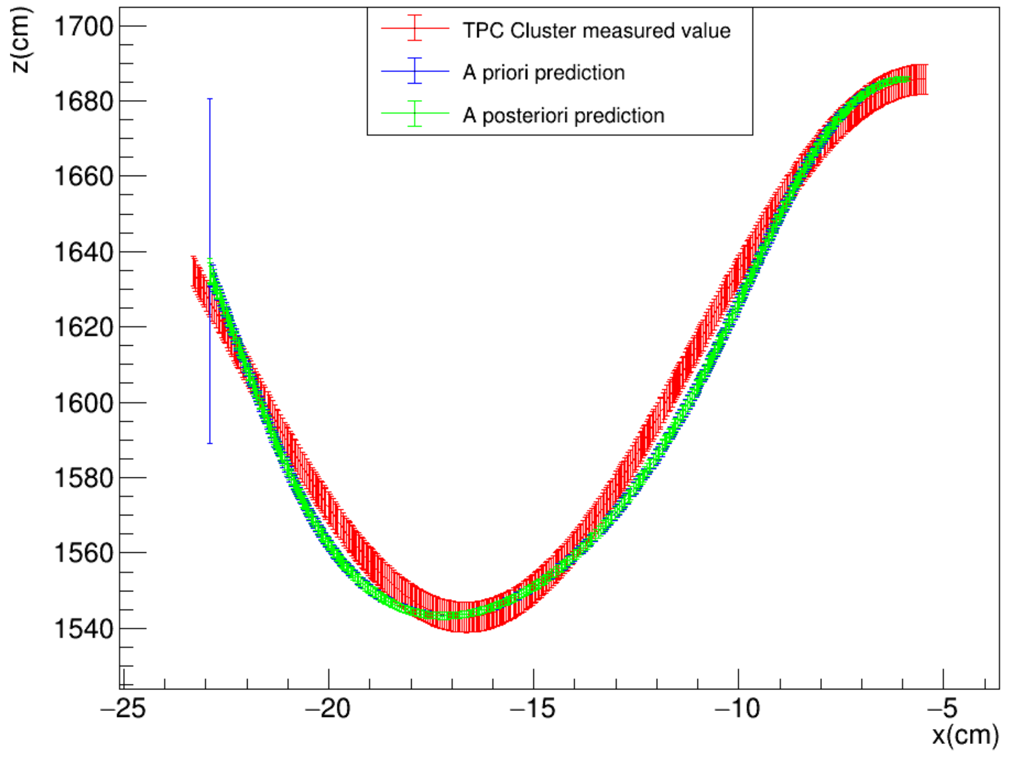
RANDOMIZED X STEP

- Plots describing the evolution of the estimate of the measured quantities (y, z) as a function of the free parameter x for the perfect helix. Note that the the TPC points have on the x coordinate the measured value, while for the estimates we use the estimated x (i.e. $x_k = x_0 + k \times dx_k$)

Y



Z

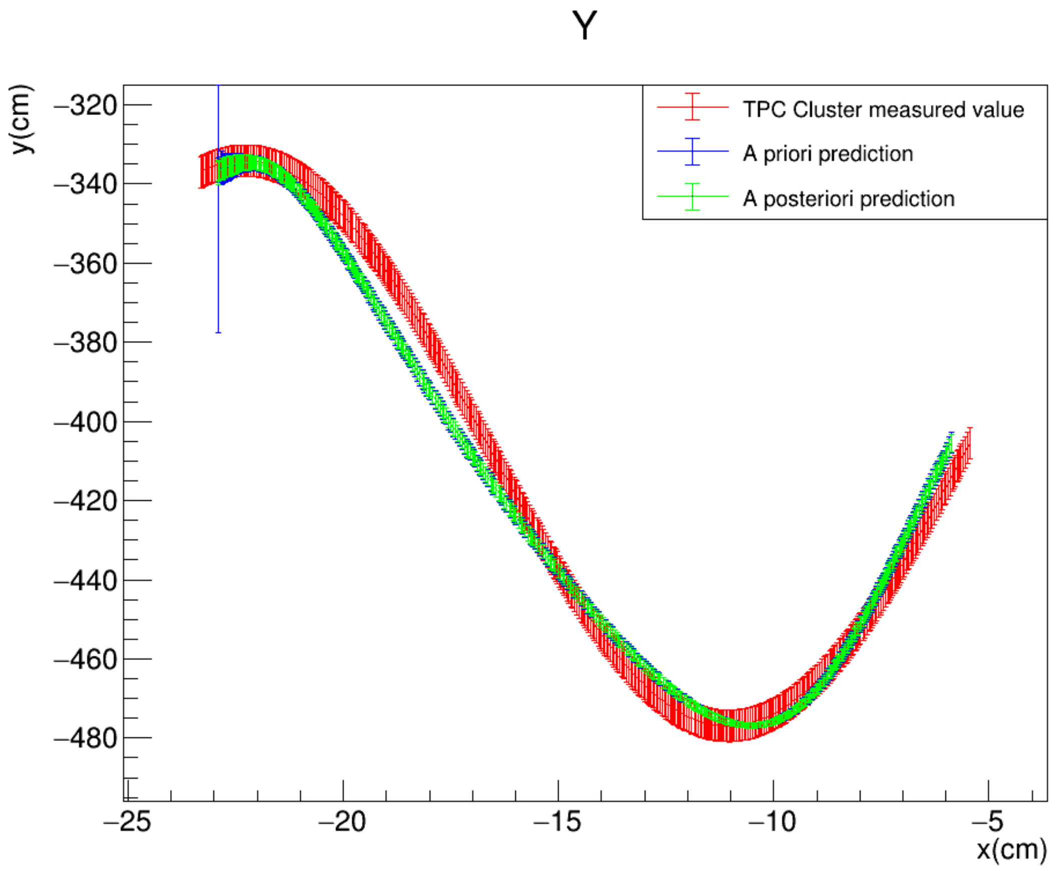


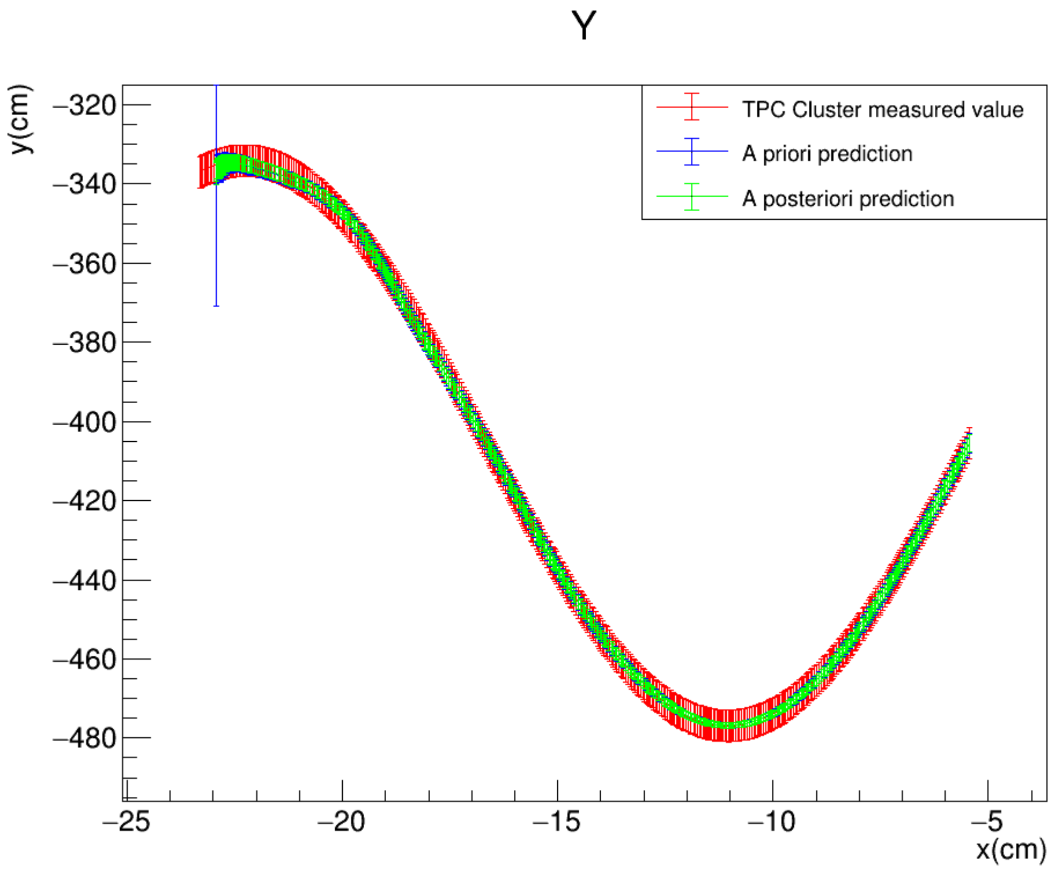
KF™: $(\sigma_x, \sigma_{yz}) = (0.5cm, 1cm)$

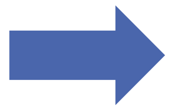
FILTER PROPAGATION DIRECTION

RANDOMIZED X STEP

- Reapplying the fit with the new $\sigma_{yz} = 4$cm we see that the 3D predictions are more in line with the Monte Carlo truth, past the first few steps





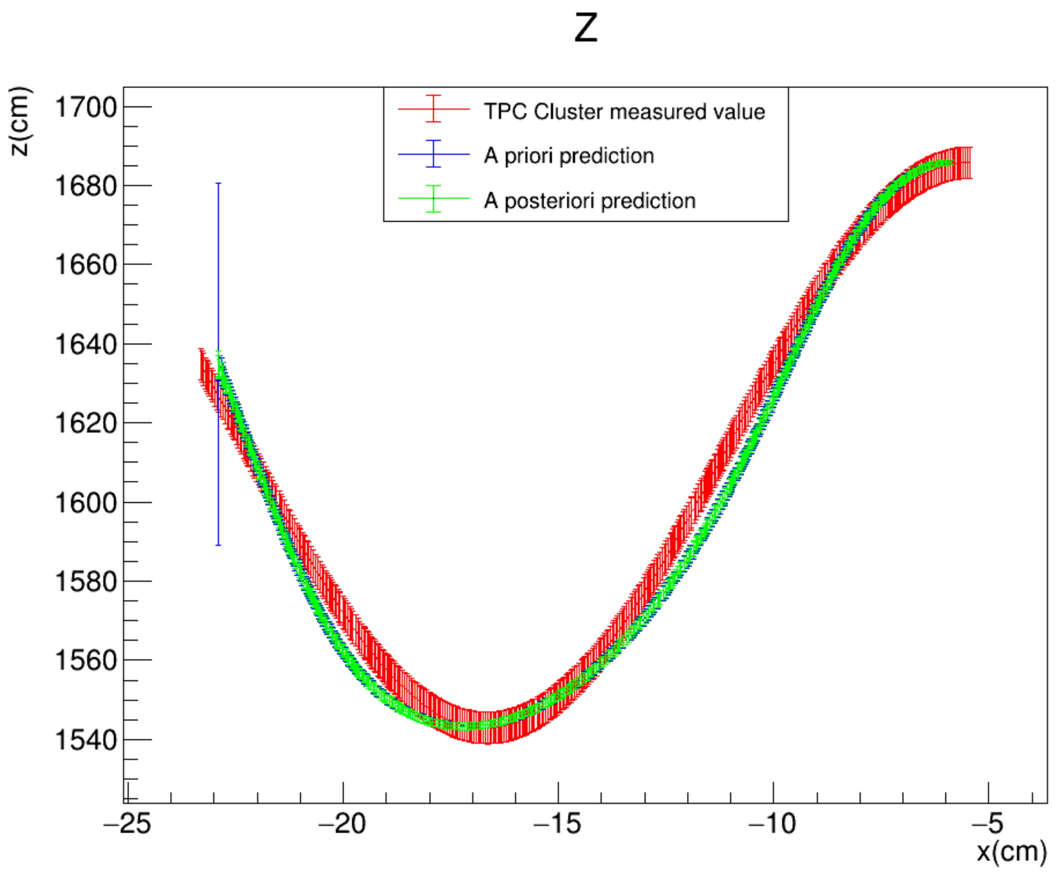


KF™: $(\sigma_x, \sigma_{yz}) = (0.5cm, 1cm)$

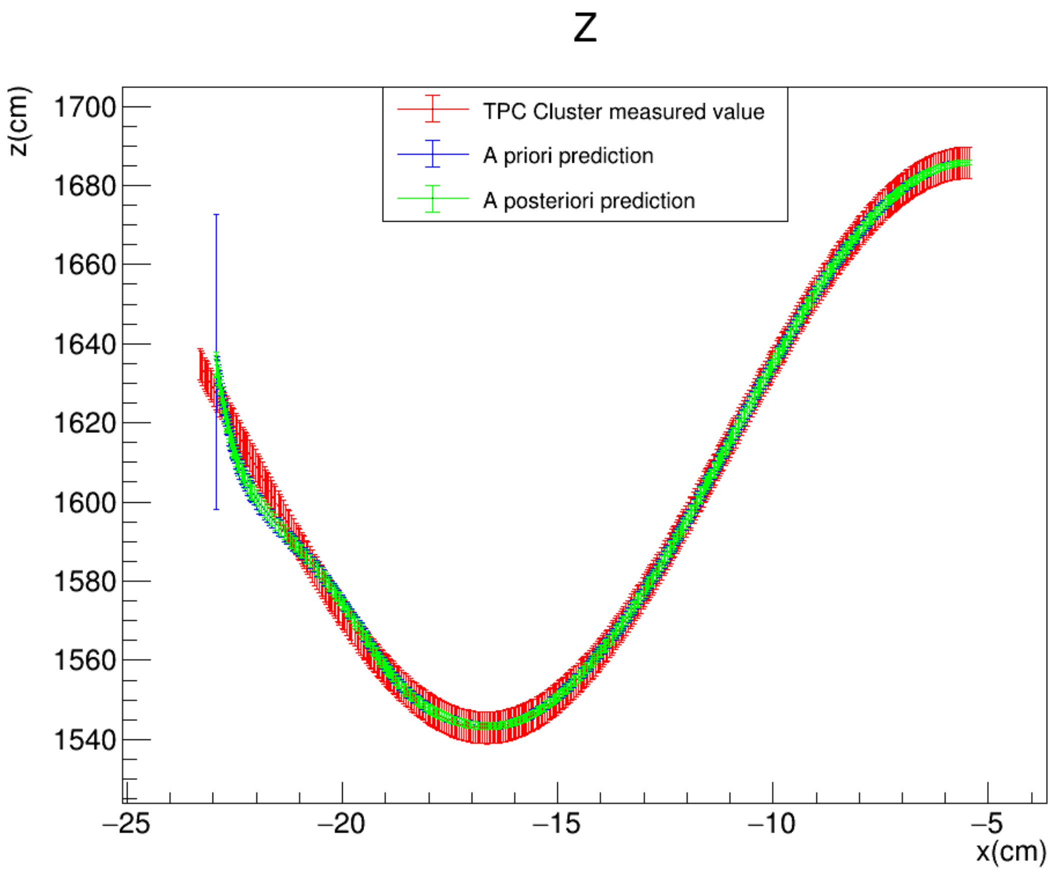
KF™: $(\sigma_x, \sigma_{yz}) = (0.5cm, 4cm)$

RANDOMIZED X STEP

- Reapplying the fit with the new $\sigma_{yz} = 4\text{cm}$ we see that the 3D predictions are more in line with the Monte Carlo truth, past the first few steps



KF™: $(\sigma_x, \sigma_{yz}) = (0.5cm, 1cm)$

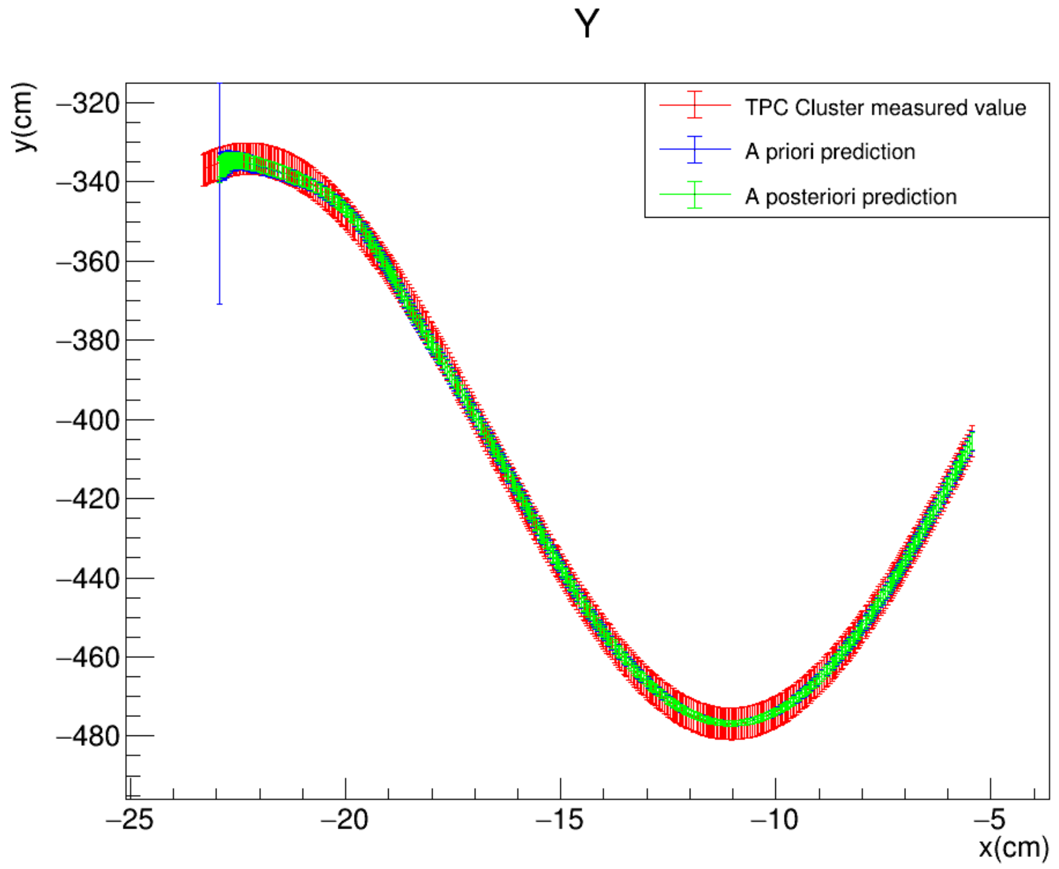


KF™: $(\sigma_x, \sigma_{yz}) = (0.5cm, 4cm)$

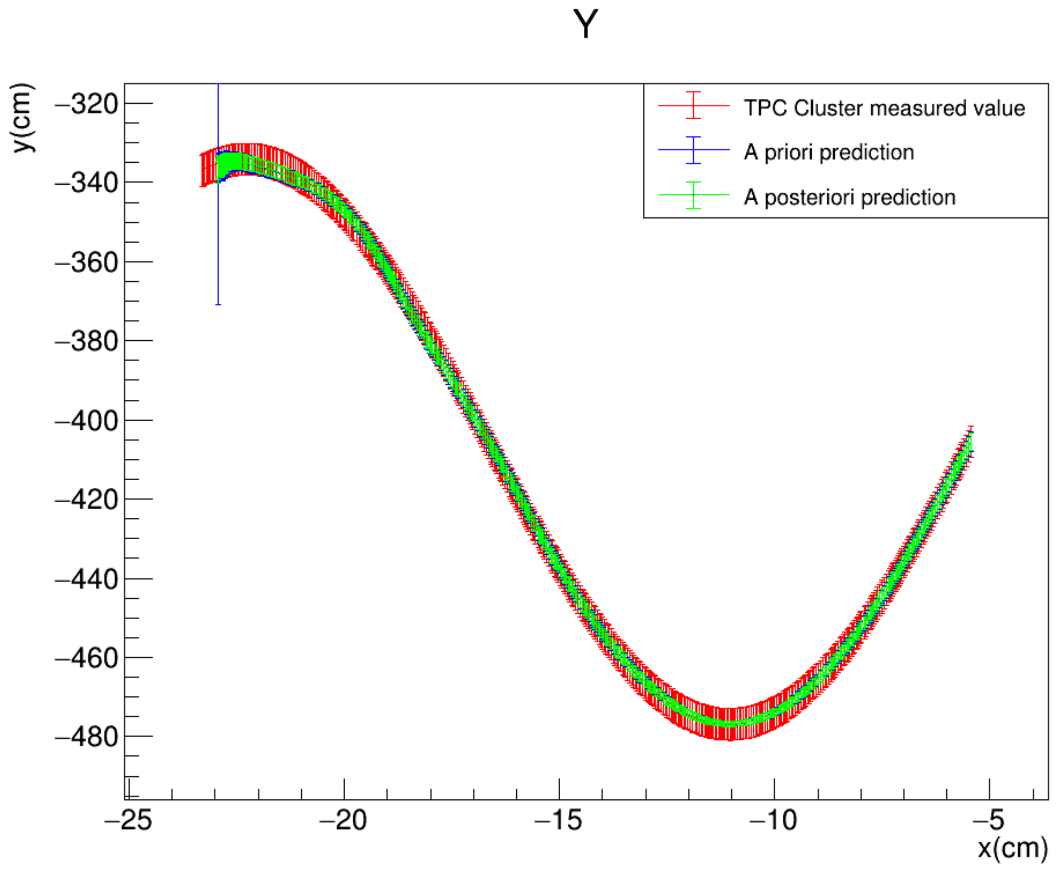
FILTER PROPAGATION DIRECTION

RANDOMIZED X STEP: TESTING THE RATIO

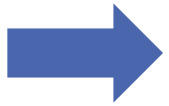
- Since σ_x and σ_{yz} are essentially weights, only their ratio σ_x/σ_{yz} should matter. To test this we try the pair $(\sigma_x, \sigma_{yz}) = (1cm, 8cm)$ which has the same ratio as $(\sigma_x, \sigma_{yz}) = (0.5cm, 4cm)$



KF™: $(\sigma_x, \sigma_{yz}) = (0.5cm, 4cm)$

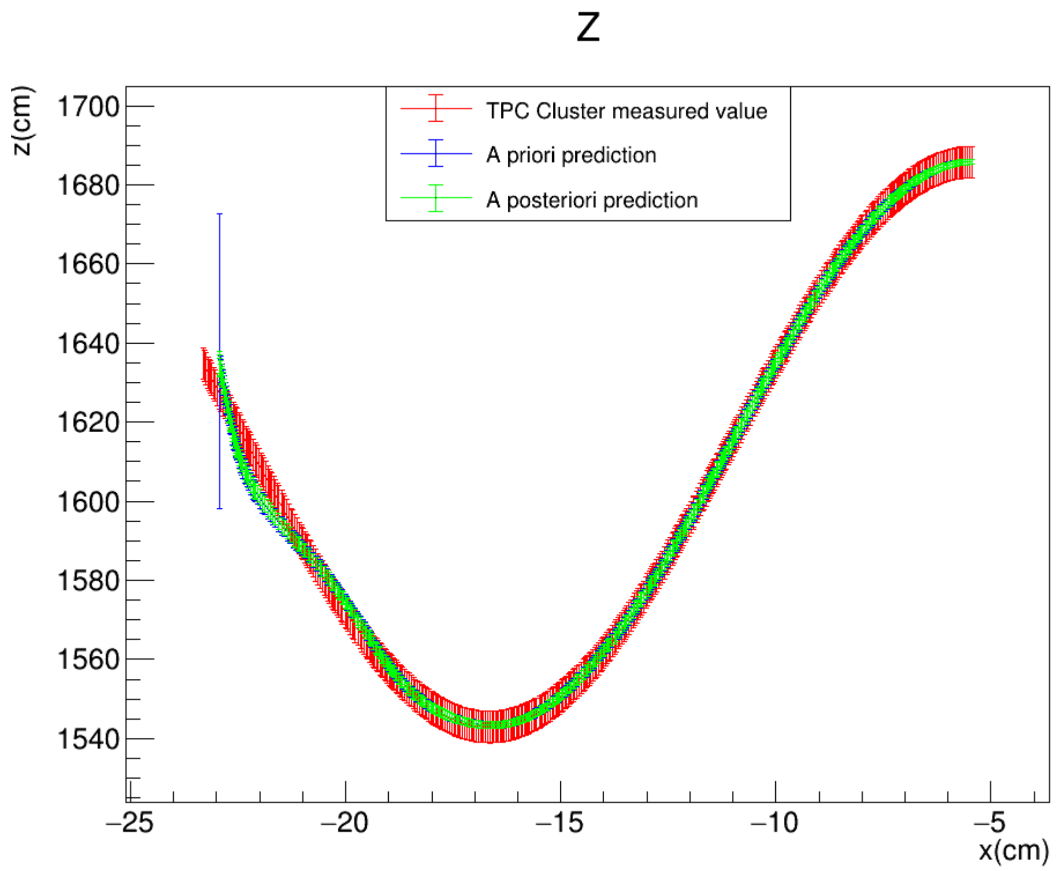


KF™: $(\sigma_x, \sigma_{yz}) = (1cm, 8cm)$

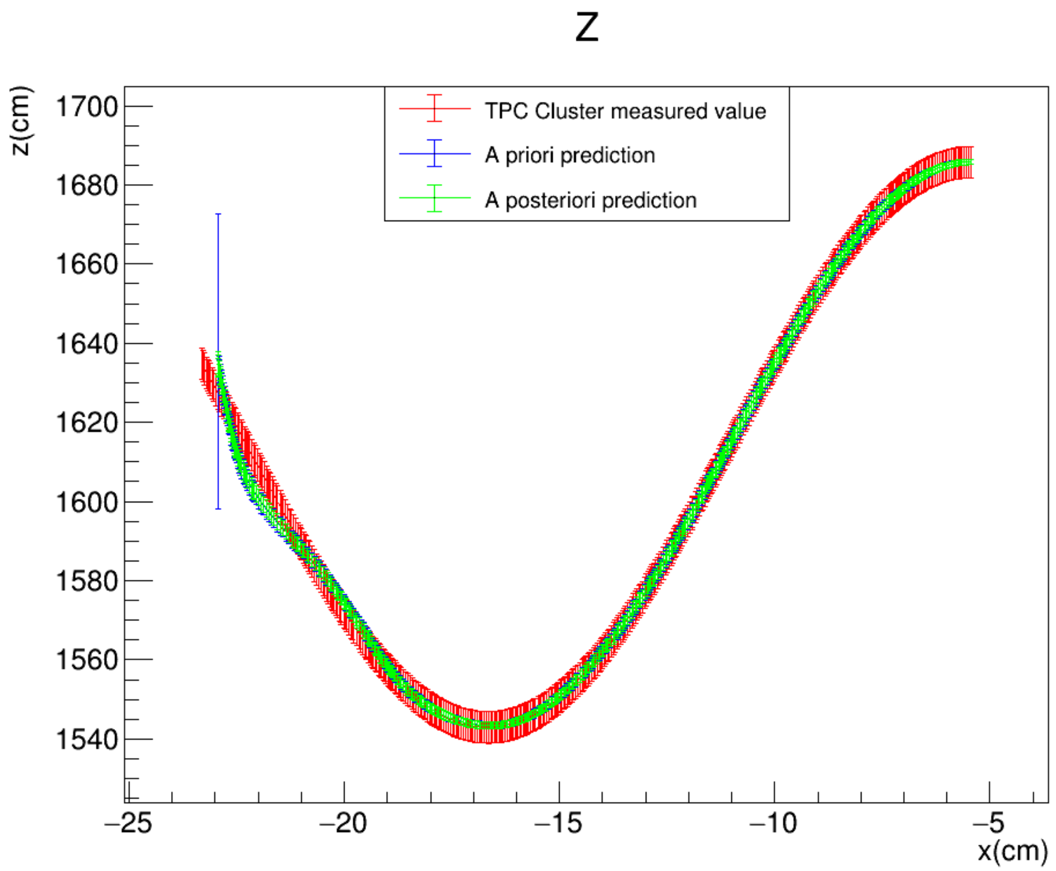


RANDOMIZED X STEP: TESTING THE RATIO

- Since σ_x and σ_{yz} are essentially weights, only their ratio σ_x/σ_{yz} should matter. To test this we try the pair $(\sigma_x, \sigma_{yz}) = (1cm, 8cm)$ which has the same ratio as $(\sigma_x, \sigma_{yz}) = (0.5cm, 4cm)$



KF™: $(\sigma_x, \sigma_{yz}) = (0.5cm, 4cm)$

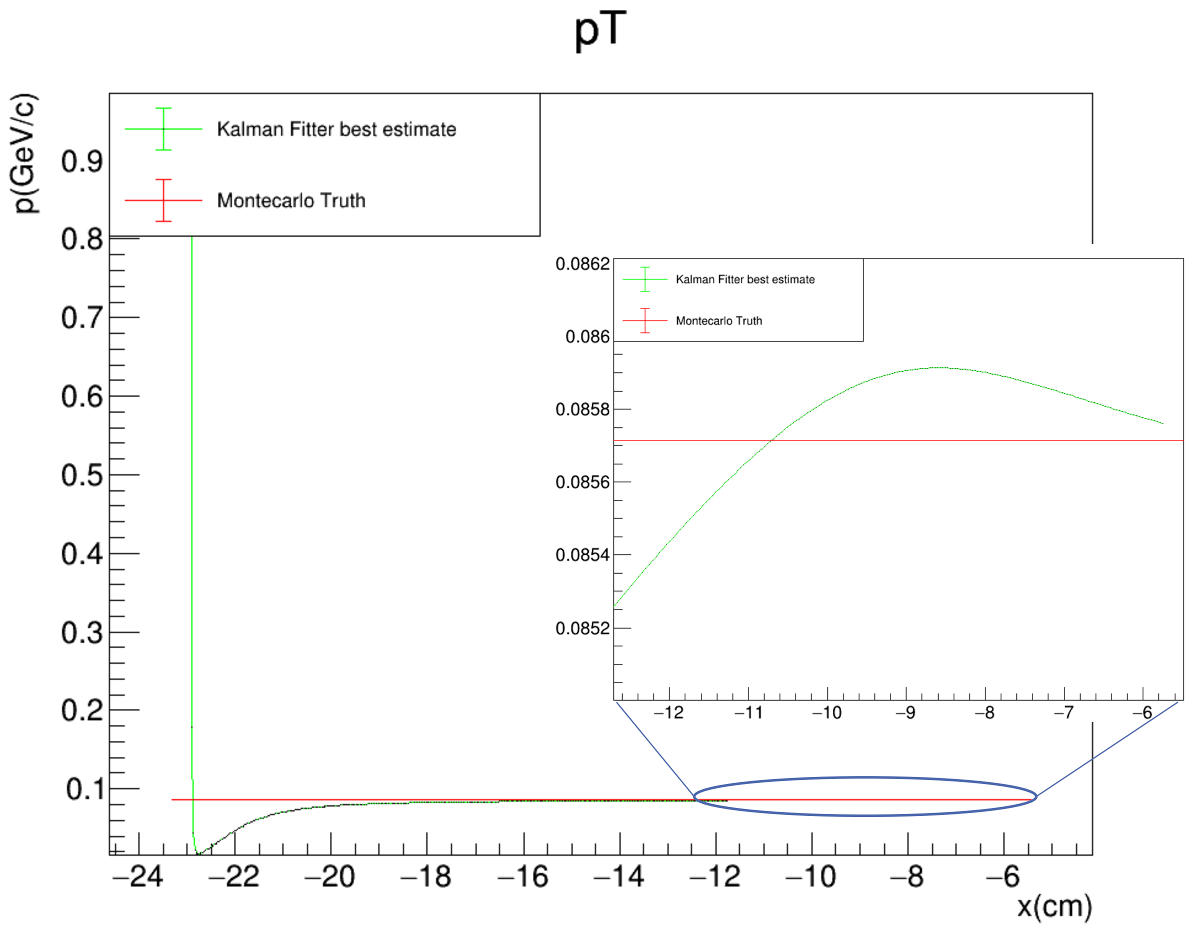


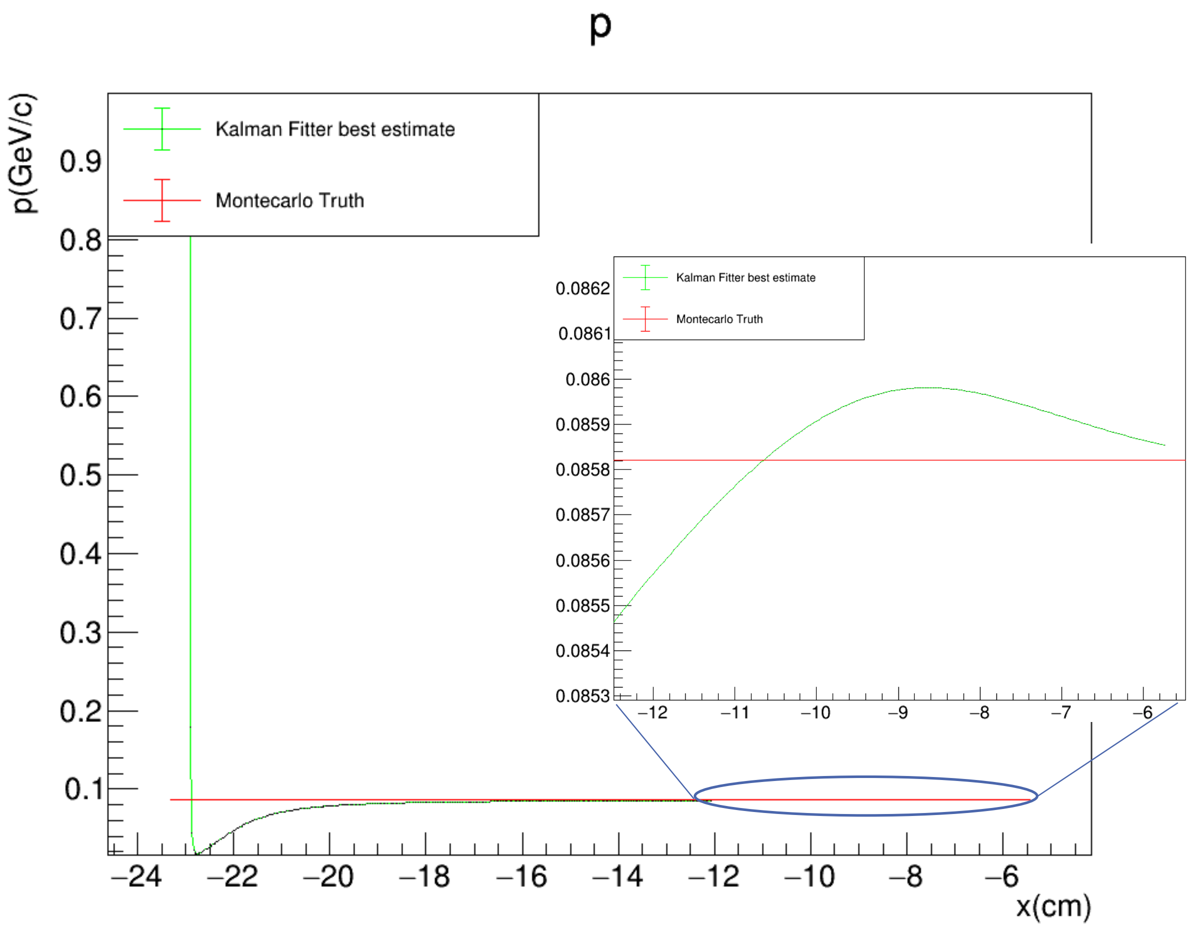
KF™: $(\sigma_x, \sigma_{yz}) = (1cm, 8cm)$

FILTER PROPAGATION DIRECTION

RANDOMIZED X STEP

- Plots comparing the reconstructed tansverse momentum p_T and the reconstructed total momentum p_T from our Kalman filter algorithm™ to the MC truth as a function of the free parameter x



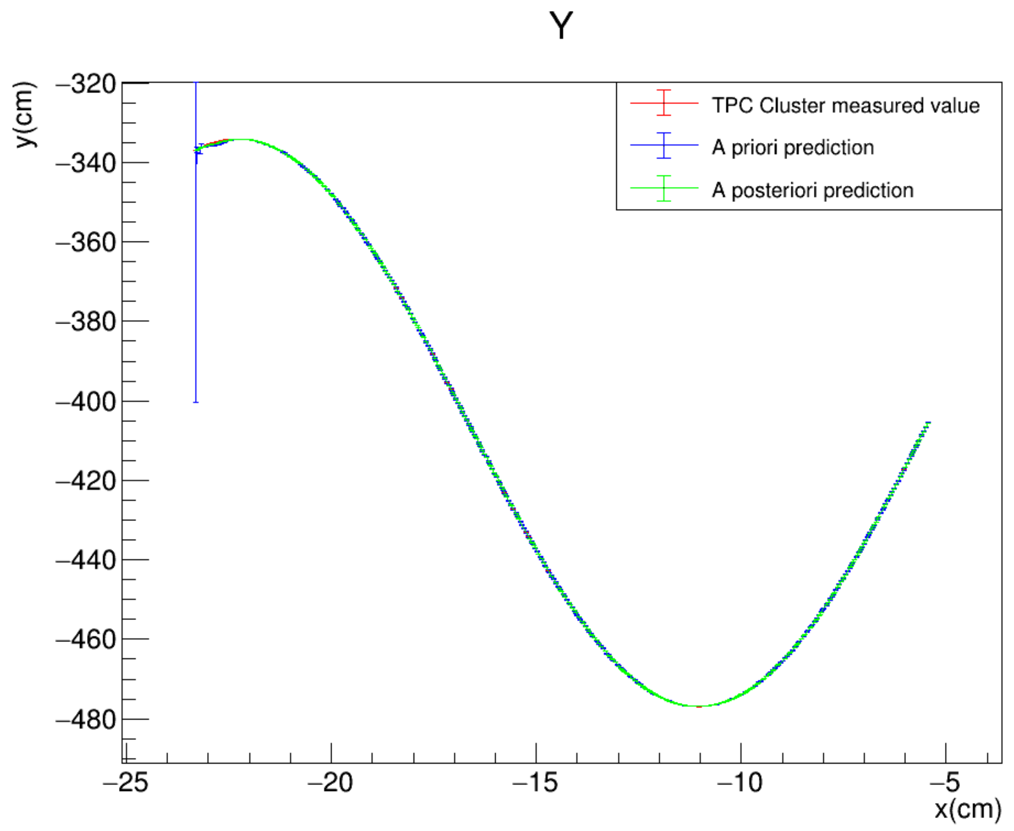


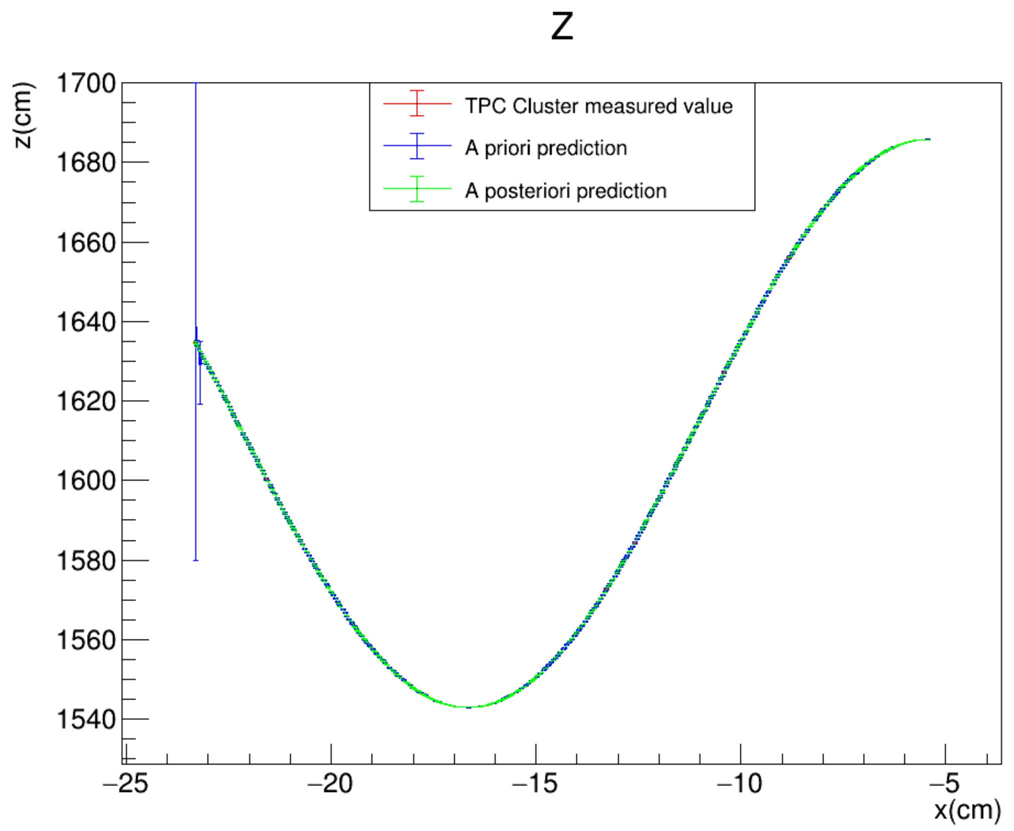
KF™: $\left(\sigma_x, \sigma_{yz}\right) = (0.5cm, 1cm)$

FORWARD FIT

APPLYING A VERY SMALL R A STANDARD KALMAN FILTER

- We try applying a standard Kalman Filter to our Toy Monte Carlo, substituting the values in the R matrix with $\Sigma_{xy} = 10^{-6}$cm , where before we had $\Sigma_{xy} = 4$cm: with such a small R the Kalman filter follows the measurements almost exactly

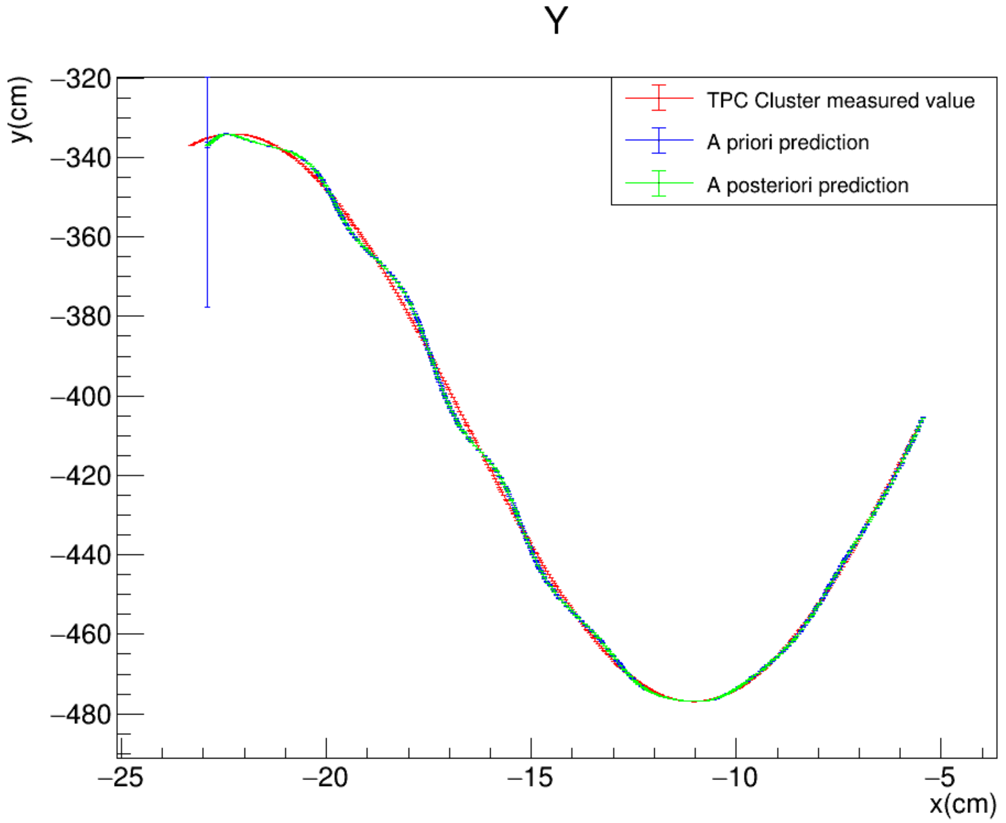


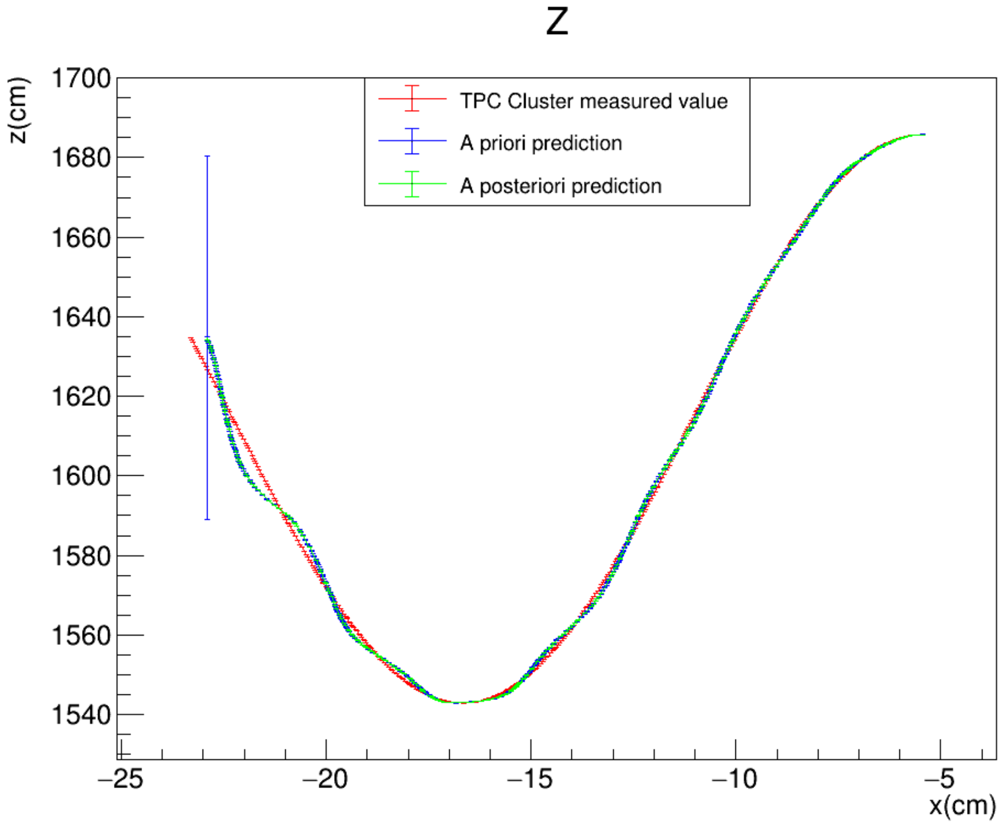


FITTER PROPAGATION DIRECTION

APPLYING A VERY SMALL R TO OUR KALMAN FILTER

- We try applying our Kalman Filter™ to our Toy Monte Carlo, but substituting the values in the R matrix with $\Sigma_{xy} = 10^{-6}$cm , where before we had $\Sigma_{xy} = 4$cm: with such a small R the Kalman filter should converge with the measurement very quickly (this happens immediately with a Standard Kalman filter, as you can see in the previous slide)





KF™: $(\sigma_x, \sigma_{yz}) = (0.5cm, 4cm)$

FITTER PROPAGATION DIRECTION

SUMMARY AND FUTURE STEPS

- The Kalman Filter now in use is convoluted with Kinematic Fitting which determines the evolution of the free parameter, minimizing the distance between the measured value and the a priori prediction

- The Kinematic Fitting is regulated by two weights σ_{yz} and σ_x whose ratio determines weather the fit is dominated by corrections related to the yz or the x measurements and predictions

- With the previous values of $\left(\sigma_x, \sigma_{yz}\right) = (0.5cm, 1cm)$ the Kalman Fit disregarded the x measurements in the update
  - Increasing the value of σ_{yz} ,bringing the ratio from $\sigma_{yz}/\sigma_x = 2$ to $\sigma_{yz}/\sigma_x = 8$ the fit better takes into account the x measurements and follows the Monte Carlo truth more closely

- Future steps:
  - Weights with a larger σ_{yz}/σ_x ratio (i.e. following the x measurement more closely) should maybe be used
  - A smeared helix Toy Monte Carlo needs to be investigated
  - Other options for the free parameter need to be studied

BACKUP

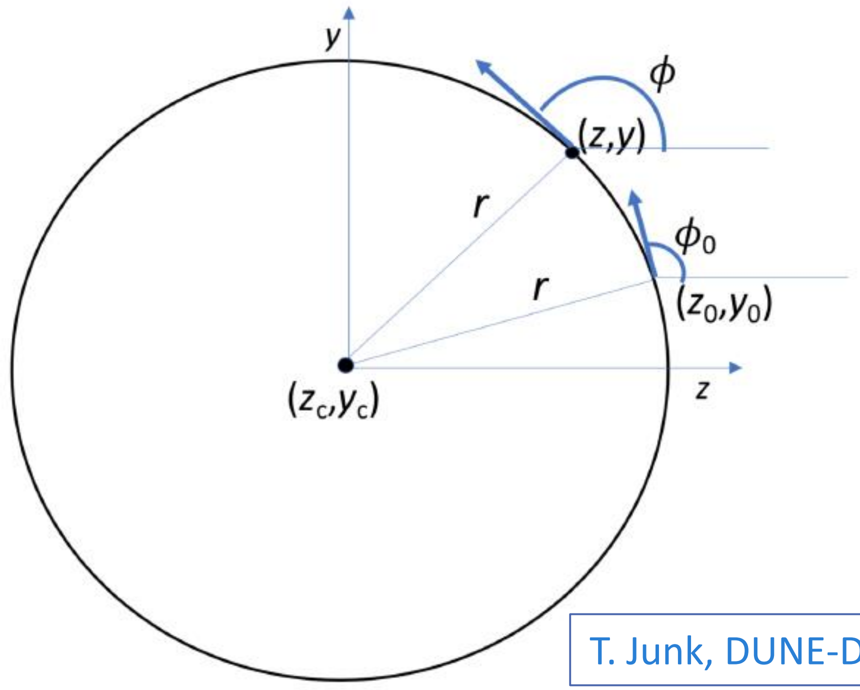
KALMAN FILTER APPLICATION

- We apply a Kalman filter to the motion of a charged particle in the magnetic field

$$\begin{cases} x = x_0 + r\tan\lambda\,(\phi - \phi_0) \\ y = y_C - r\cos\phi \\ z = z_C + r\sin\phi \end{cases}$$

TRACK PARAMETERS

We apply the Kalman filter to track candidates, consisting of groups of TPC clusters, which are identified and put together during the reconstruction process. Each step of the Kalman filter algorithm is identified by one of these TPC clusters



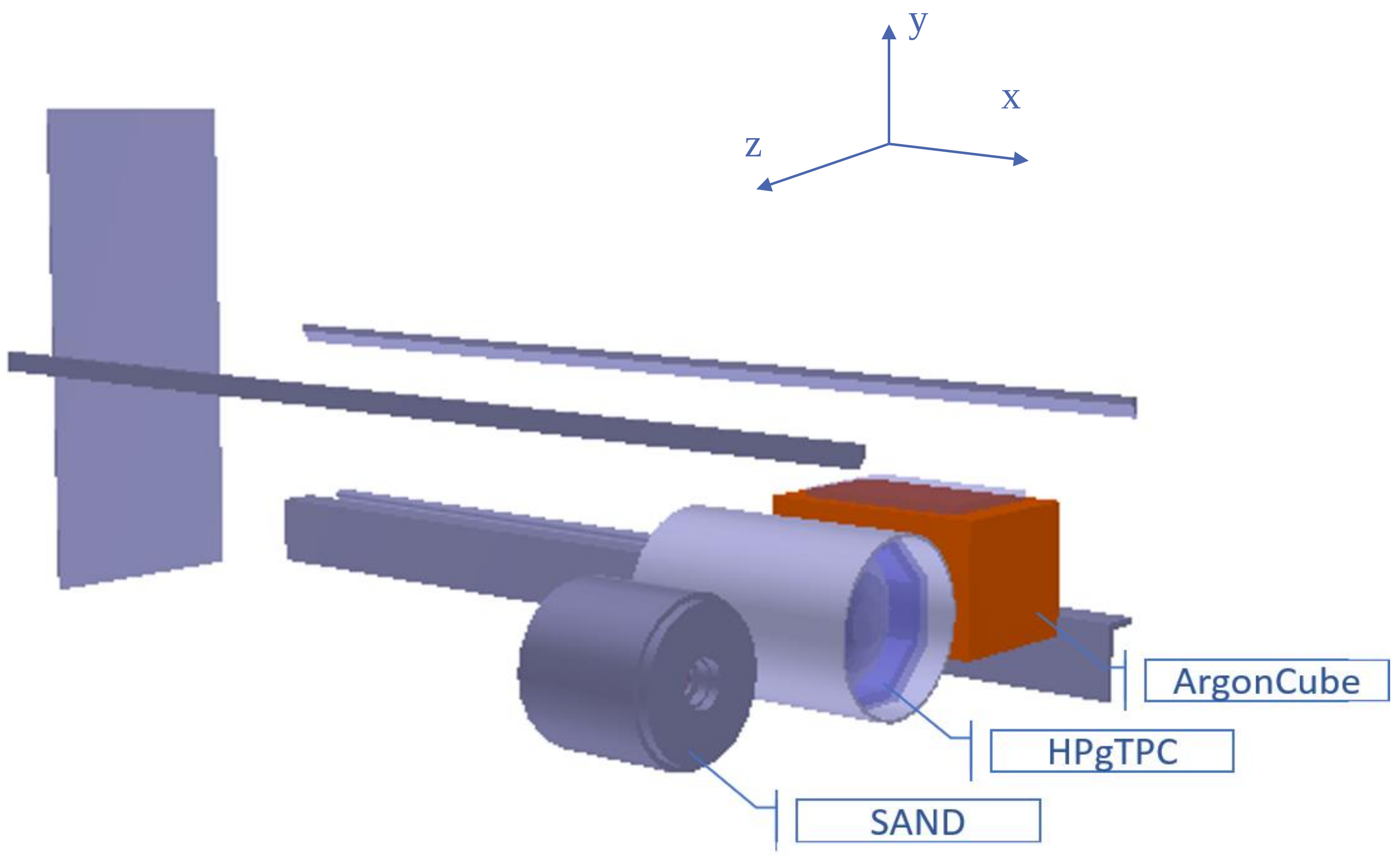
T. Junk, DUNE-Doc-13933

FIG. 1: Track parameter definitions in the (z, y) plane.

https://docs.dunescience.org/cgi-bin/private/ShowDocument?docid=13933

KALMAN FILTER: COORDINATE SYSTEM

- Note that in our coordinate sytem z is the flux direction, y is the vertical direction and x is the drift direction (i.e. the magnetic field direction)



KALMAN FILTER APPLICATION: INITIAL ESTIMATES

- Before the Kalman filter algorithm can be applied, we need an initial estimate for the state vector, which in our case includes $y, z, 1/r, \phi$ and λ and the covariance matrix

STATE VECTOR

$$x_0^T = (y_0 \quad z_0 \quad 1/r_0 \quad \phi_0 \quad \lambda_0) = (0 \quad 0 \quad 0.1 \quad 0 \quad 0)$$

COVARIANCE MATRIX

$$P_0 = \begin{pmatrix} 1^2 & 0 & 0 & 0 & 0 \\ 0 & 1^2 & 0 & 0 & 0 \\ 0 & 0 & 0.5^2 & 0 & 0 \\ 0 & 0 & 0 & 0.5^2 & 0 \\ 0 & 0 & 0 & 0 & 0.5^2 \end{pmatrix}$$

- The estimated quantities from the state vector can be used to estimate the particle's momentum

$$\begin{cases} p_x = p_T \tan\lambda \\ p_y = p_T \sin\phi \\ p_z = p_T \cos\phi \end{cases}$$

$$p_T(\mathrm{GeV}/c) = 0.3 \times B(T) \times r(m)$$

KALMAN FILTER APPLICATION: PREDICTION AND MEASUREMENT

- From the equation of motion we obtain the prediction function for our state vector. Note that the drift direction x is used as our free parameter

$$\hat{x}_k^- = f\begin{pmatrix} \hat{y}_{k-1}^+ \\ \hat{z}_{k-1}^+ \\ 1/\hat{r}_{k-1}^+ \\ \hat{\phi}_{k-1}^+ \\ \hat{\lambda}_{k-1}^+ \end{pmatrix} = \begin{pmatrix} \hat{y}_{k-1}^+ + \mathrm{dx_k} \times \cot\hat{\lambda}_{k-1} \times \sin\hat{\phi}_{k-1}^+ \\ \hat{z}_{k-1}^+ + \mathrm{dx_k} \times \cot\hat{\lambda}_{k-1} \times \cos\hat{\phi}_{k-1}^+ \\ 1/\hat{r}_{k-1}^+ \\ \hat{\phi}_{k-1}^+ + \mathrm{dx_k} \times \cot\hat{\lambda}_{k-1} \times 1/\hat{r}_{k-1}^+ \\ \hat{\lambda}_{k-1}^+ \end{pmatrix}$$

Note: the prediction model does not account for dE/dx energy loss

- The only measured quantities in our case are y and z, and are set at the center of the TPC cluster correspondent to the present step.

$$s_k^h = \begin{pmatrix} y_k^h \\ z_k^h \end{pmatrix}$$

KALMAN FILTER APPLICATION: COVARIANCE MATRIX PREDICTION

- First step to make the prediction for the covariance matrix is to calculate the Jacobian

$$F_{k-1} = \frac{\partial f(\hat{x}_{k-1}^{+})}{\partial \hat{x}_{k-1}^{+}} = \begin{bmatrix} 1 & 0 & 0 & dx_k \cot\hat{\lambda}_{k-1}^{+} \cos\hat{\phi}_{k-1}^{+} & dx_k \sin\hat{\phi}_{k-1}^{+}(-1-\cot^2\hat{\lambda}_{k-1}^{+}) \\ 0 & 1 & 0 & -dx_k \cot\hat{\lambda}_{k-1}^{+} \sin\hat{\phi}_{k-1}^{+} & dx_k \cos\hat{\phi}_{k-1}^{+}(-1-\cot^2\hat{\lambda}_{k-1}^{+}) \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & dx_k \cot\hat{\lambda}_{k-1}^{+} & 1 & dx_k/\hat{r}_{k-1}^{+}\,(-1-\cot^2\hat{\lambda}_{k-1}^{+}) \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

- The step uncertainty matrix is also needed

$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{\Delta 1/r} & 0 & 0 \\ 0 & 0 & 0 & \sigma_{\Delta\phi} & 0 \\ 0 & 0 & 0 & 0 & \sigma_{\Delta\lambda} \end{bmatrix}$$

- The prediction is then:

$$P_k^- = F_{k-1} P_{k-1} F_{k-1}^T + Q$$

KALMAN FILTER APPLICATION: EVALUATE THE RESIDUAL

- We now evaluate the residual and Kalman Gain

RESIDUAL

$$\tilde{y}_k = s_k^h - H(\hat{x}_k^-) = \begin{pmatrix} y_k^h - \hat{y}_k^- \\ z_k^h - \hat{z}_k^- \end{pmatrix}$$

KALMAN GAIN

$$K_k = P_k^- H(R + HP_k^- H^T)^{-1}$$

With:

$$R = \begin{pmatrix} \Sigma_{yz}^2 & 0 \\ 0 & \Sigma_{yz}^2 \end{pmatrix}$$

MEASUREMENT NOISE COVARIANCE

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

CONVERSION MATRIX

Note: The uncertainties in R are fixed, before the Kalman filter is applied, as external parameters: R is not updated.

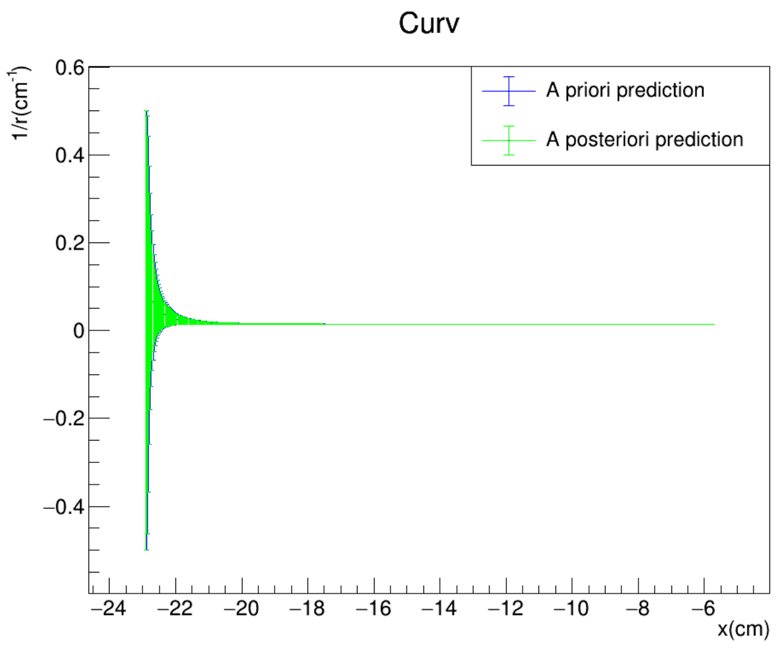
KALMAN FILTER APPLICATION: PREDICTION UPDATE

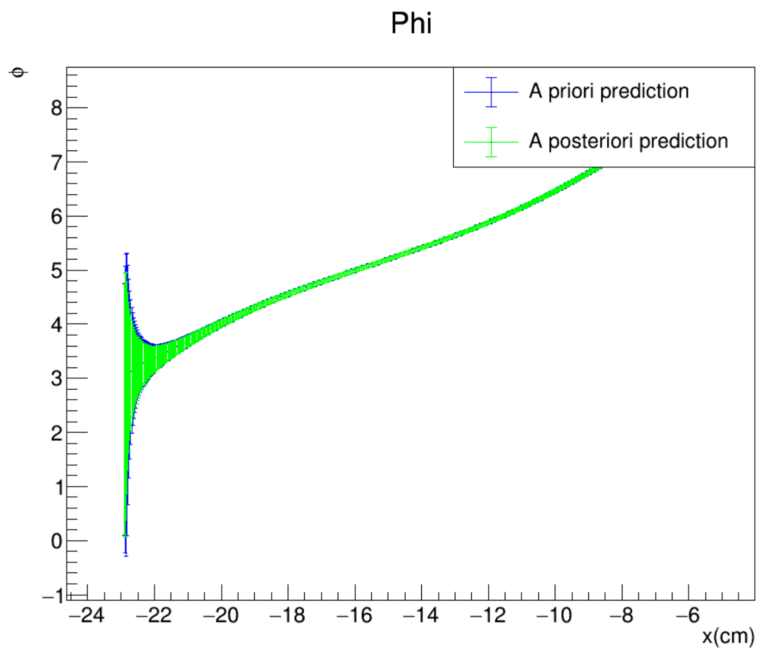
- We are now finally able to update our estimates using both the a priori prediction and the measurement

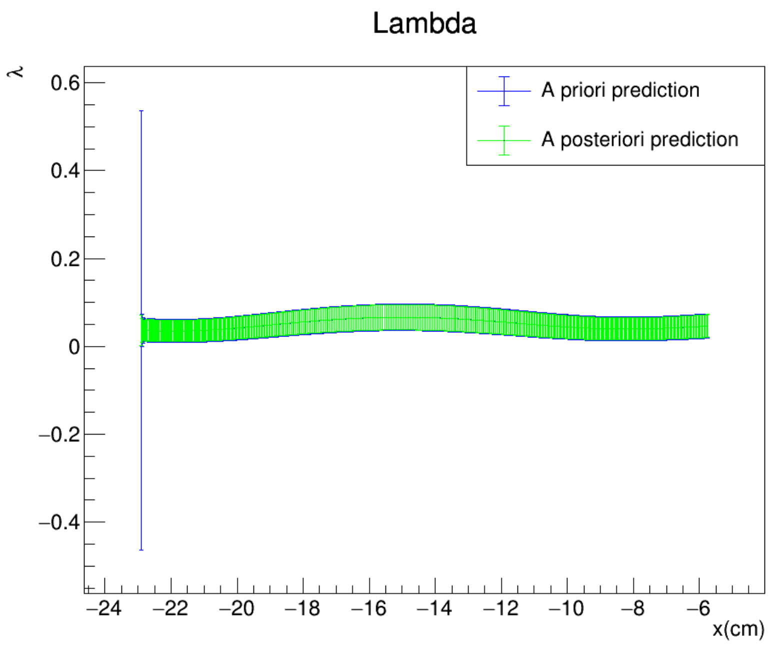
STATE VECTOR	$\hat{x}_k^+ = \hat{x}_k^- + K_k \tilde{y}_k$
COVARIANCE MATRIX	$P_k^+ = (1 - H_k K_k) P_k^-$

TOY MONTE CARLO

- Plots describing the evolution of the estimate of the non measured quantities $(\frac{1}{r}, \phi, \lambda)$ as a function of the free parameter x for the perfect helix





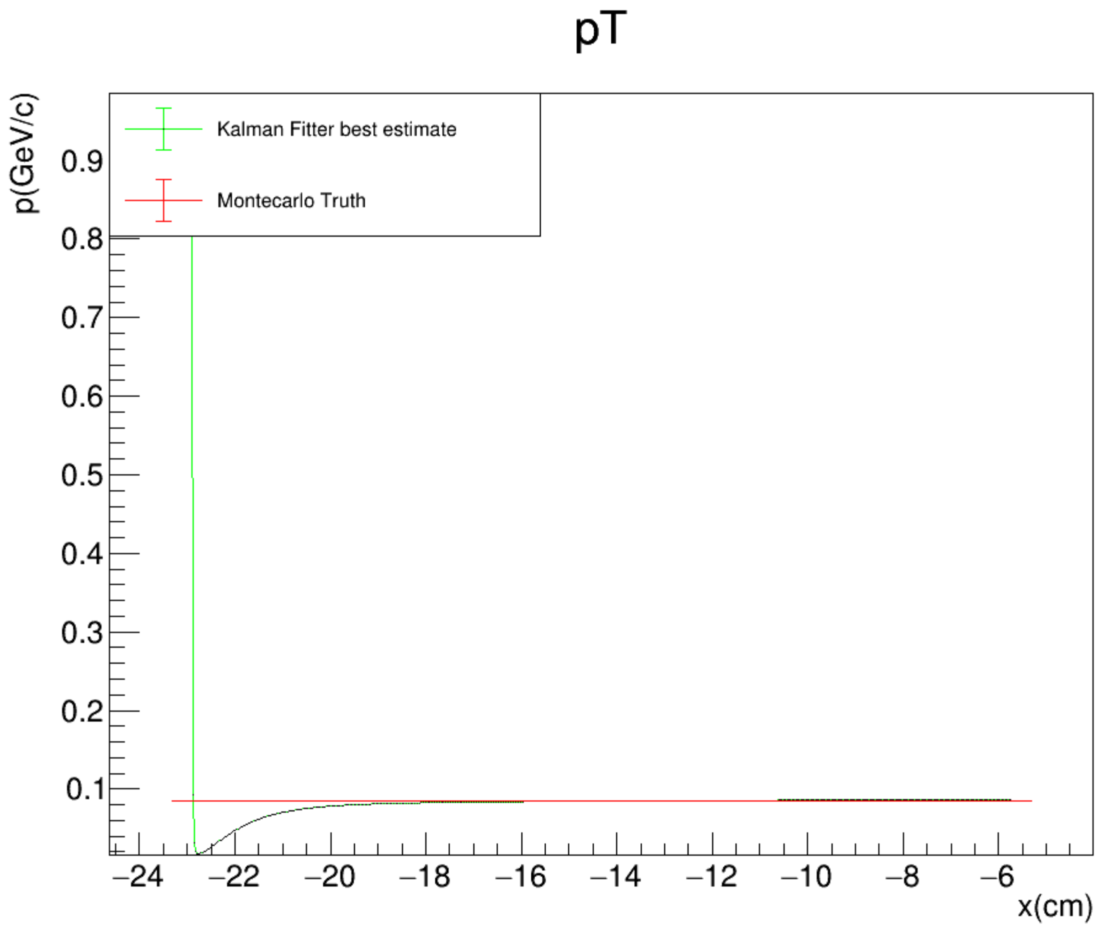


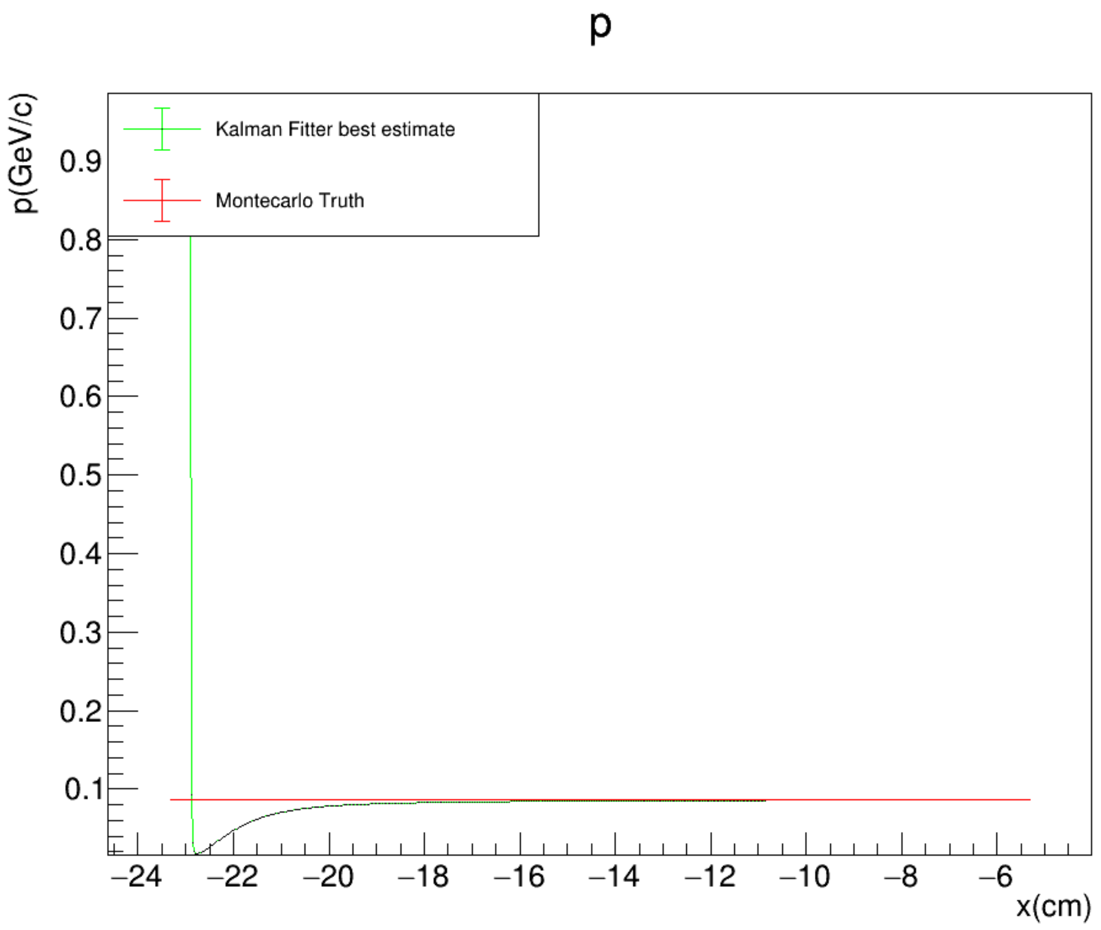
$$(\sigma_x, \sigma_{yz}) = (0.5cm, 1cm)$$

FILTER PROPAGATION DIRECTION

TOY MONTE CARLO

- Plots comparing the reconstructed tansverse momentum p_T and the reconstructed total momentum p_T from the Kalman fitter algorithm to the MC truth as a function of the free parameter x



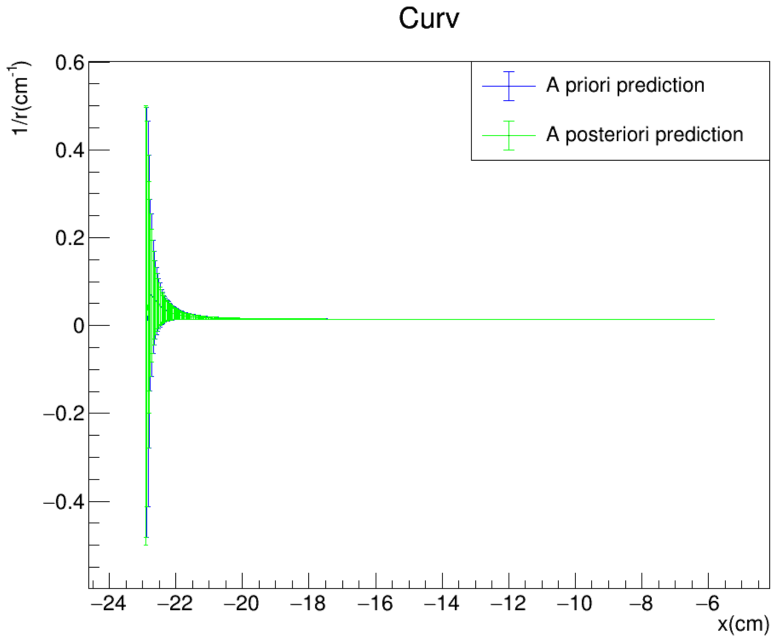


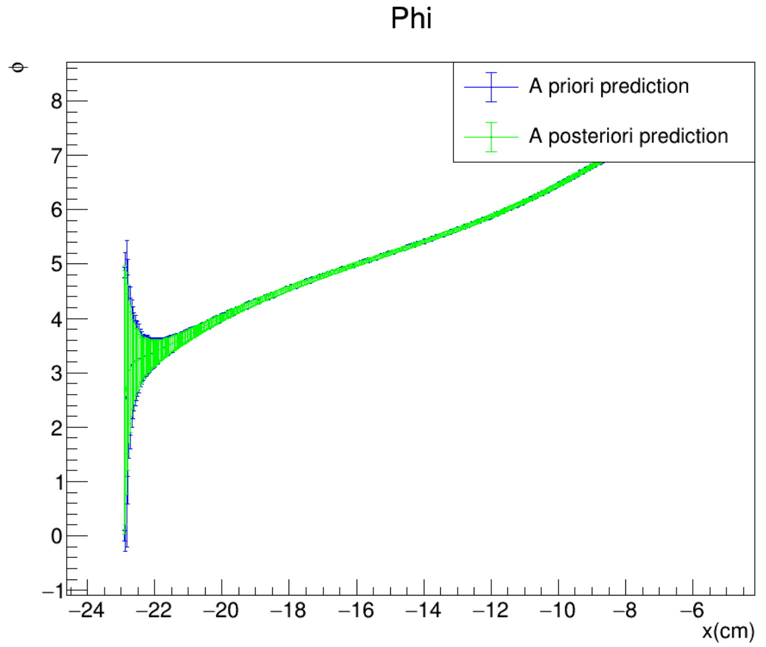
$(\sigma_x, \sigma_{yz}) = (0.5cm, 1cm)$

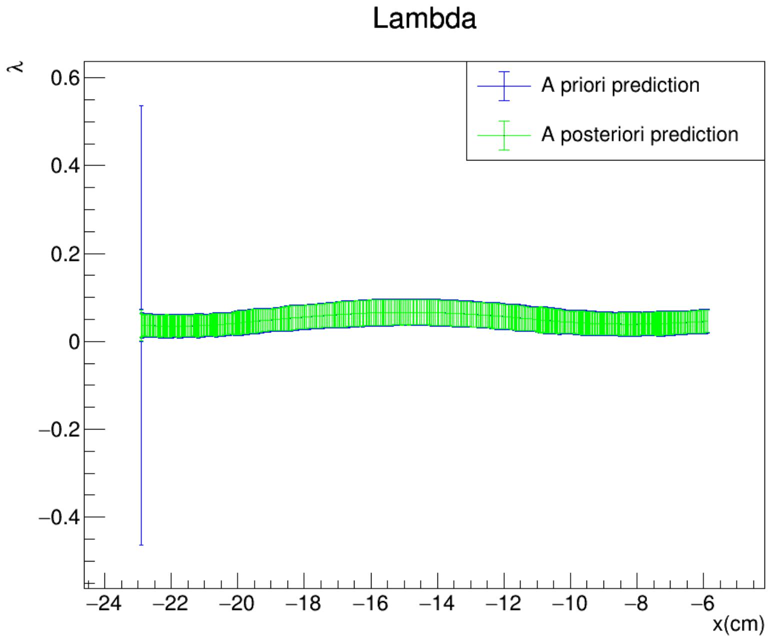
FORWARD FIT

RANDOMIZED X STEP

- Plots describing the evolution of the estimate of the non measured quantities $(\frac{1}{r}, \phi, \lambda)$ as a function of the free parameter x for the perfect helix







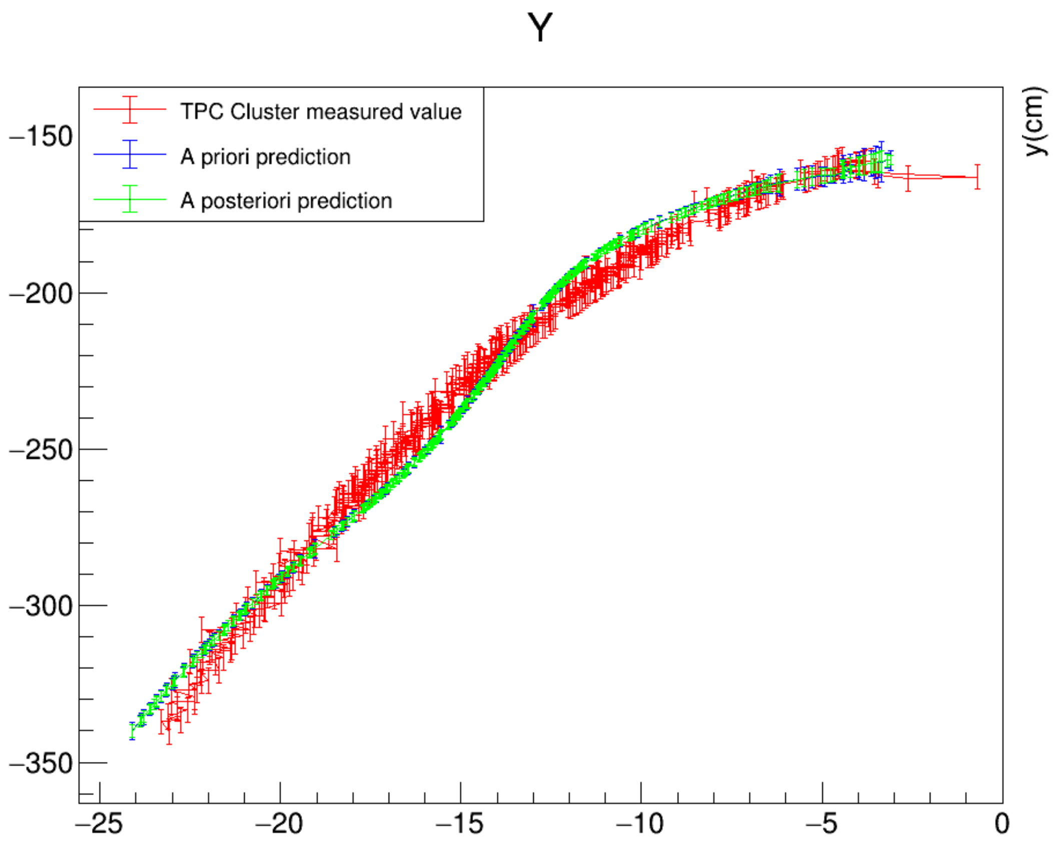
$(\sigma_x, \sigma_{yz}) = (0.5cm, 1cm)$

FILTER PROPAGATION DIRECTION

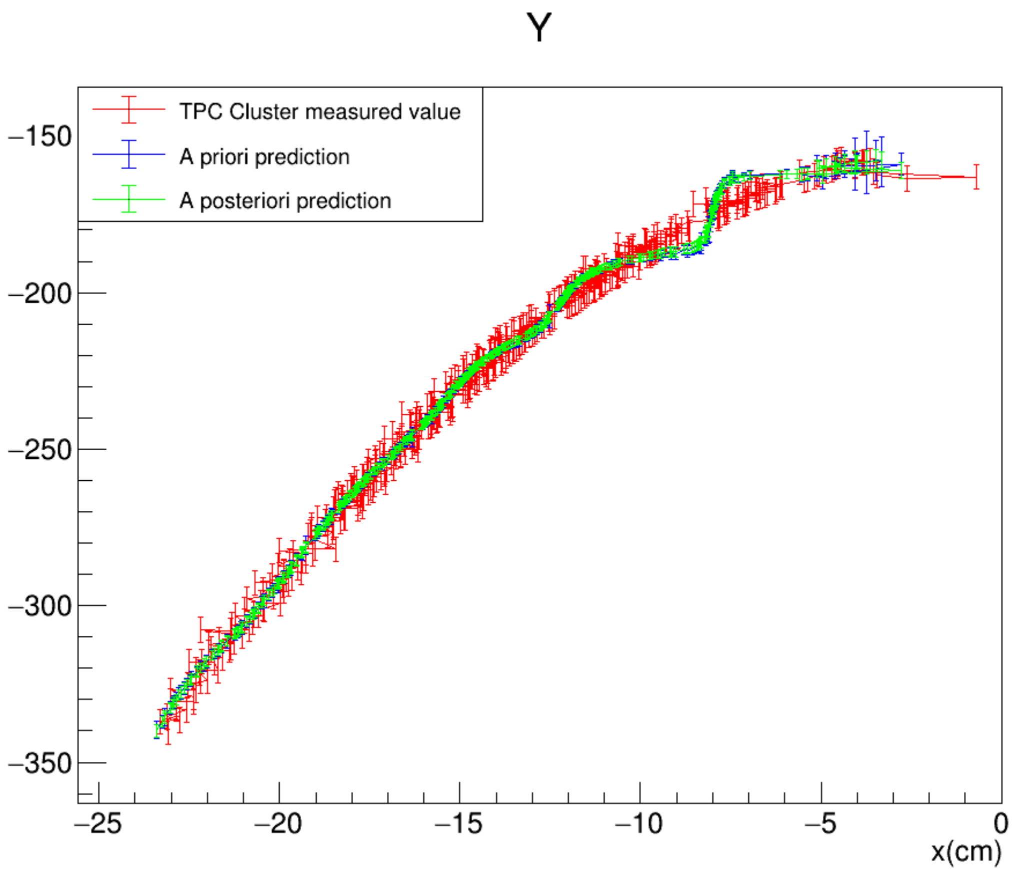
FEDERICO BATTISTI

UNDERSTANDING STEP DETERMINATION

- Reapplying the fit with the new $\sigma_{yz} = 4$cm we see that the 3D predictions are more in line with the Montecarlo truth, past the first few steps



$\sigma_{yz} = 2cm$

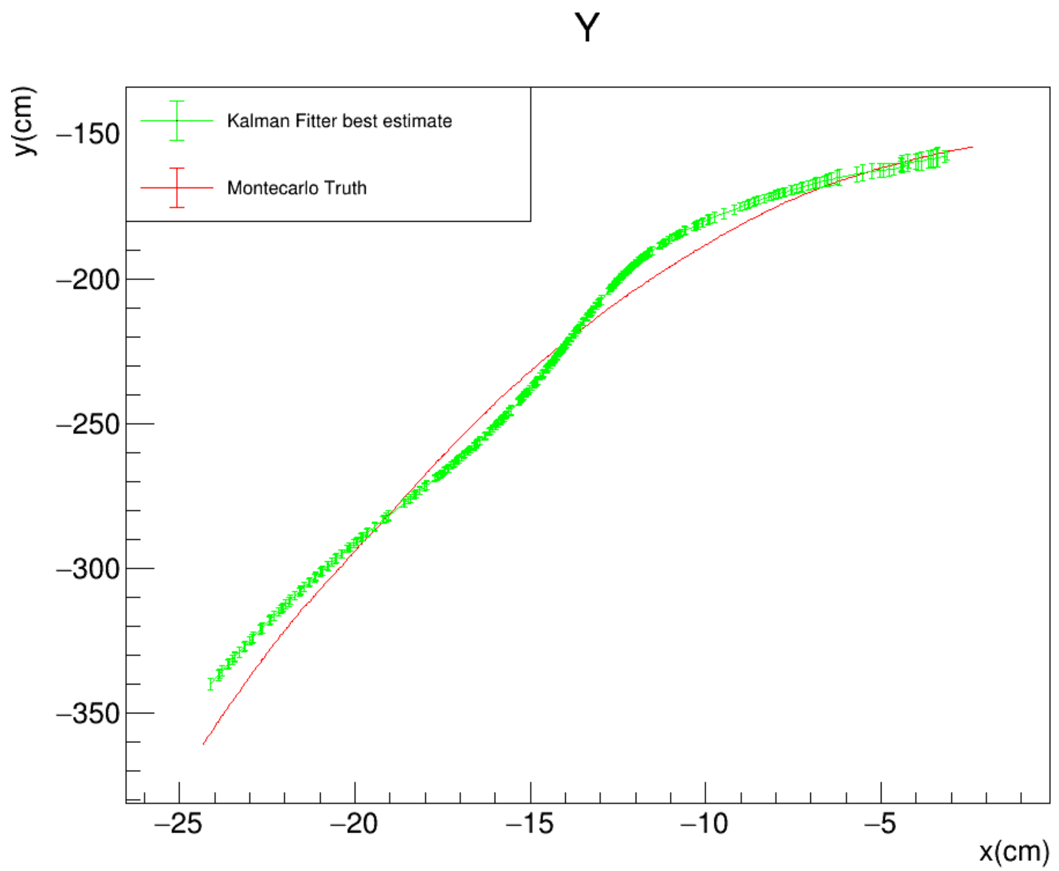


$\sigma_{yz} = 4cm$

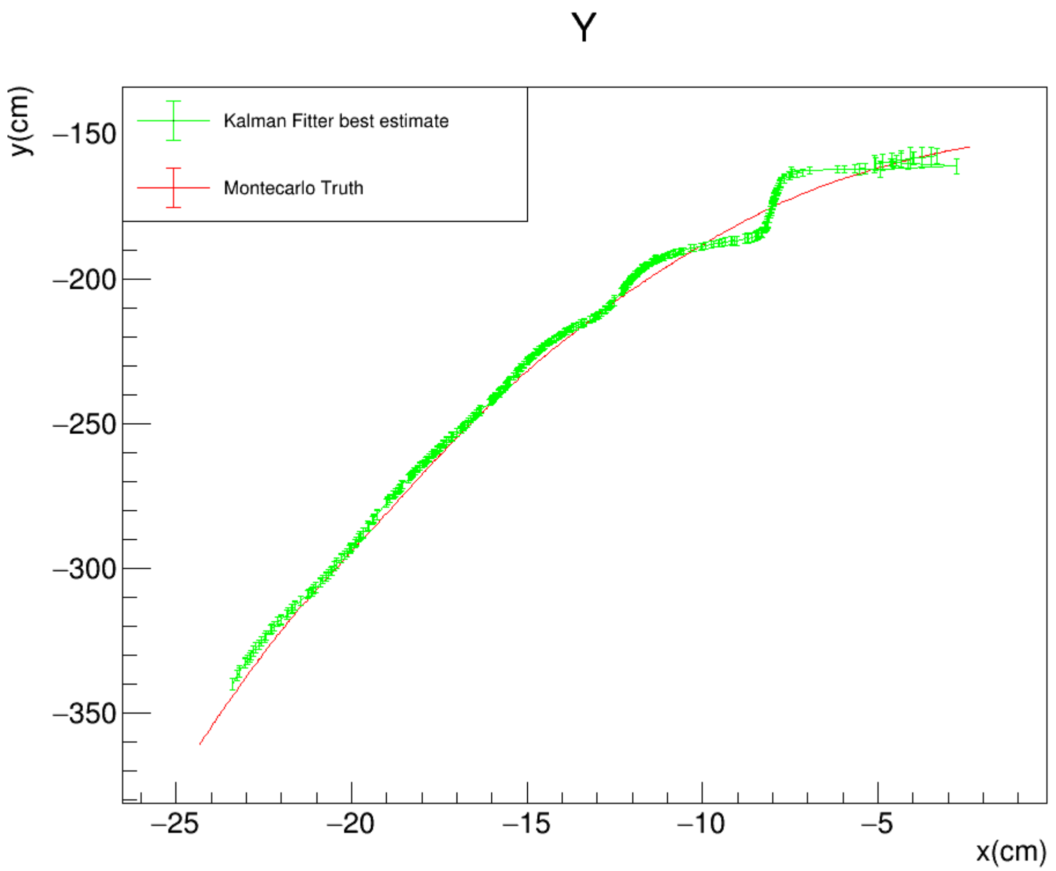
FITTER
OPAGATION
IRECTION

UNDERSTANDING STEP DETERMINATION

- Reapplying the fit with the new $\sigma_{yz} = 4$cm we see that the 3D predictions are more in line with the Montecarlo truth, past the first few steps



$\sigma_{yz} = 2cm$

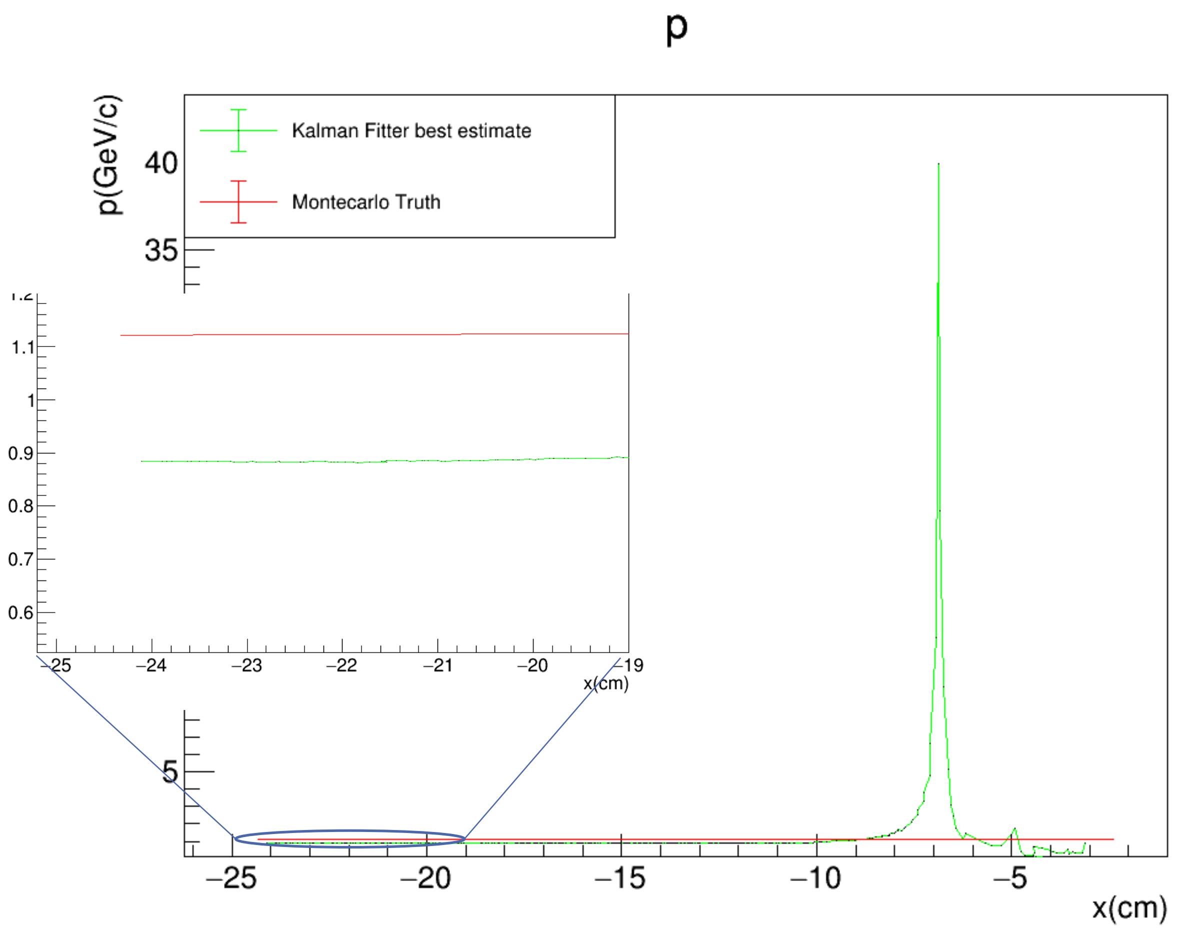


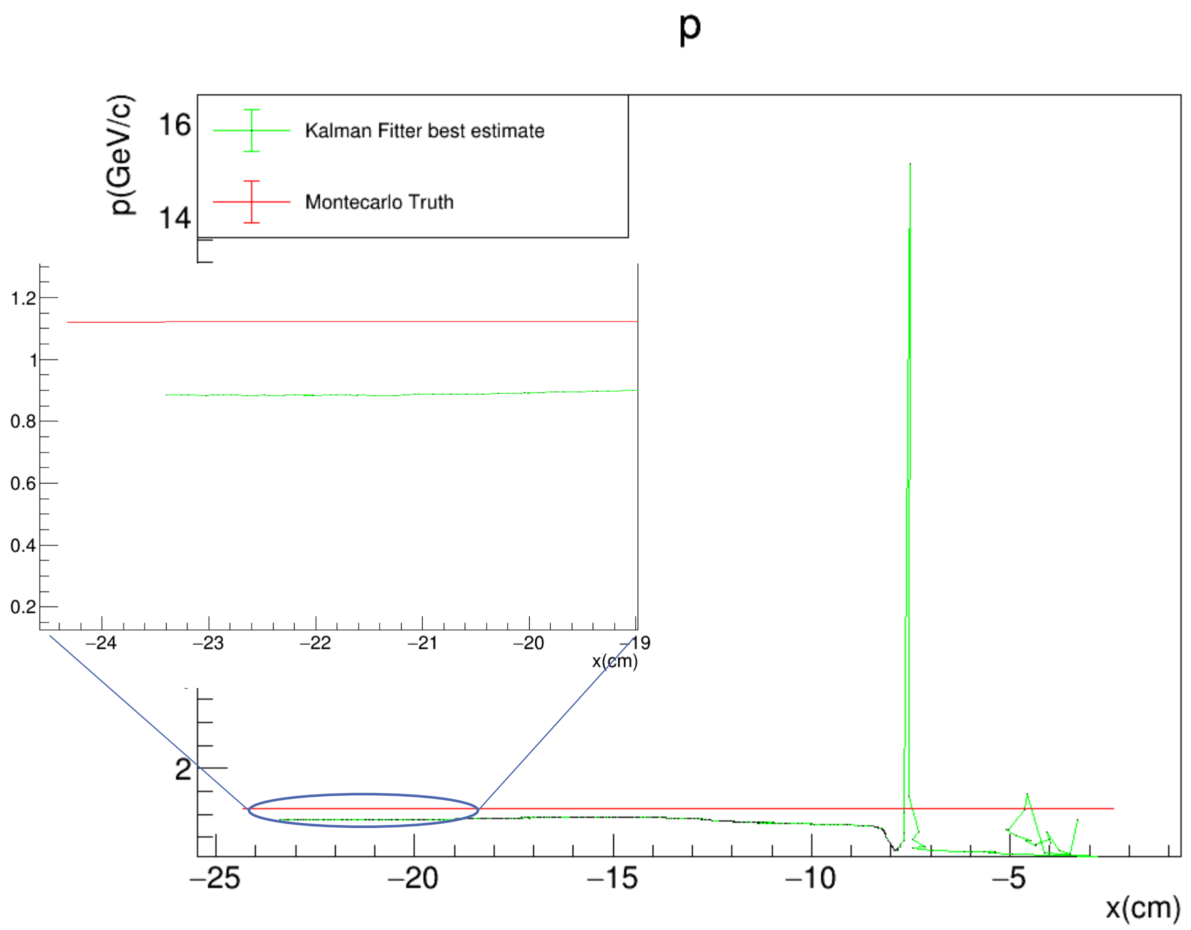
$\sigma_{yz} = 4cm$

FITTER PROPAGATION DIRECTION

UNDERSTANDING STEP DETERMINATION

- The momentum reconstruction performance remains roughly the same



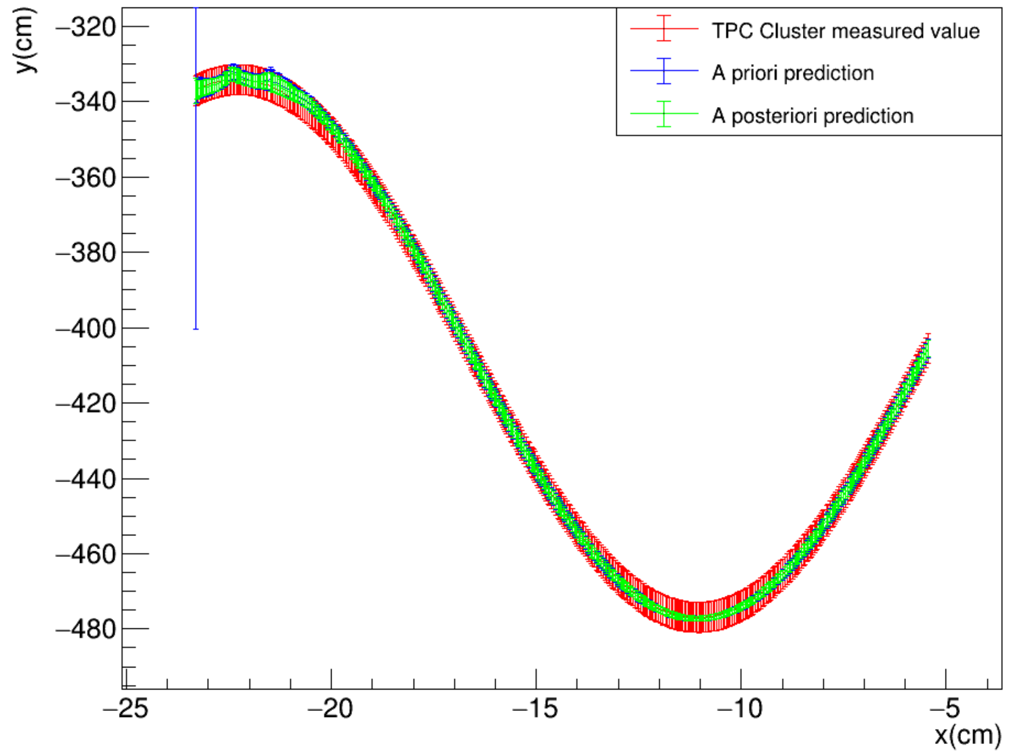


BACKWARD FIT

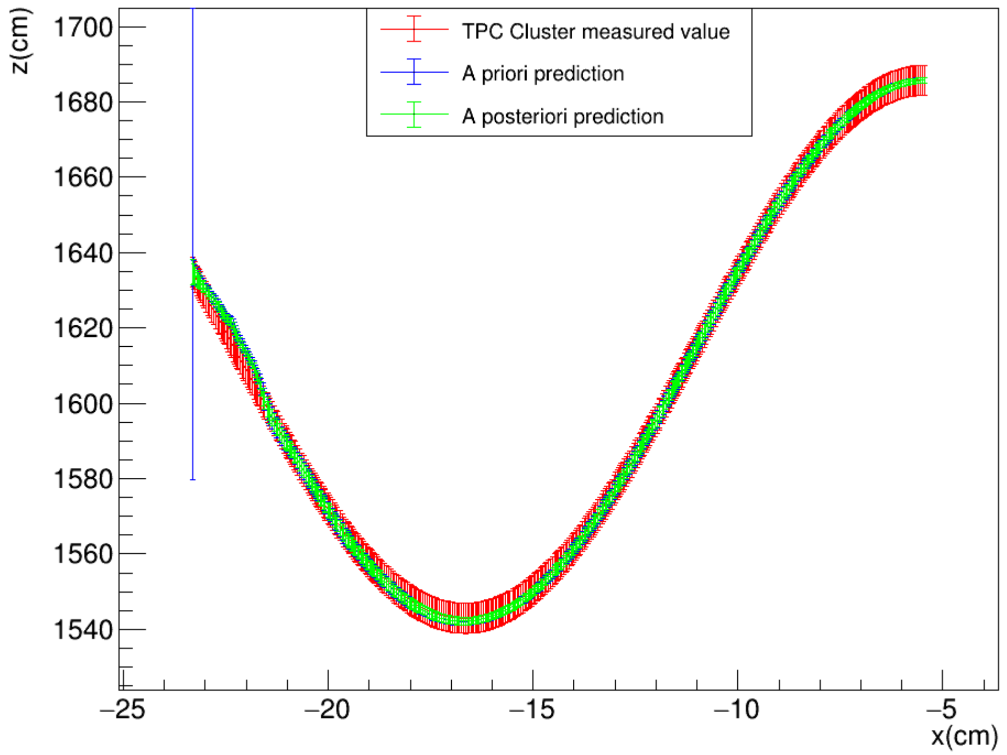
RESULTS FROM THE STANDARD KALMAN FILTER

- We try applying a standard Kalman Filter to our Toy Monte Carlo, such that it is identical to our Kalman Filter™ but it avoids the kinematic fitting algorithm completely
- This means that our free parameter coincides with the measured value of x for the TPC cluster

Y



Z



FITTER PROPAGATION DIRECTION

FEDERICO BATTISTI