# HighRR TFR Hands-On: Kalman Filter

**Alessio Piucci**

**31 May 2017**
**HighRR TFR Hands-On**


UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

# Kalman Filter

**A few slides from my old presentation at the HighRR meeting:**

**Wiki / Fitting: Kalman filter / kalman_Alessio.pdf**

# Definitions

- for a given layer, you define a measurement m with R covariance:
  m = (x, y)
  R = (2x2)

- for a track you define states x with P covariances:
  s = (x, y, $t_x$, $t_y$, qop)
  P = (5x5)

- fit nodes: fit sub-elements made of a cluster + predicted states

- state → measure projection through the projection matrix H(2x5)

- the Kalman gain K minimises the covariance of the prediction

# Kalman Filter - Flowchart

**1) Create and update the seed state, with approximated values of $t_x$, $t_y$**

**2) Forward extrapolation**

**3) Create and update the backward seed state**

**4) Backward extrapolation**

**5) Smooth the states**

**6) Compute the $x^2$ of the track**

# Kalman Filter - Extrapolation

**For each fit node:**

**1) PredictState(TFRFitnode *node, TFRFitnode *node_next)**

* propagate the state to the next layer: ***how?***

**2) UpdateState(TFRFitnode *node)**

**Kalman gain:** $K = P * H^t * (H * P * H^t + R)^{-1}$

**updated state:** $s' = s + K * (m - H * s)$

**updated covariance:** $P' = P - (K * H) * P$

**3) compute the $x^2$**

$\delta' = m - H * s'$

$P_\delta' = R - H * P' * H^t$

$x^2 = \delta' * (P_\delta'^{-1} * \delta')$

# Kalman Filter - Smoothing

**Smoothing of each fit node:**

- $s_f$, $P_f$ = forward-predicted state and covariance
- $s_b$, $P_b$ = backward-predicted state and covariance
- $K = P_b * (P_f + P_b)^{-1}$

**the smoothed state and covariance are:**

$$s_{sm} = s_b + K * (s_f - s_b)$$
$$P_{sm} = K * P_f$$

**and remember to compute its $x^2$!**

# Kalman Filter

**Task: implement a fit based on Kalman filtering in the TFRKalmanFilter class**

**- not take in account the multiple scattering in the covariance**