Kalman Filtering

The following is a suggested implementation of the Kalman Filter (KF). This is not intended as an explanation of how the KF works, but rather as an outline of how to implement it. I will provide a brief description, however, for those who haven't looked into Kalman filtering yet. Many of the references posted on the E614 web site provide excellent information on this topic.

The process of Kalman filtering consists of mainly three steps:

1. **Prediction:** in this step the state vector x_k (which basically contains the track parameters) is propagated from detector *k-1* to detector *k* using some track model. The track model is implemented in the propagator (sometimes referred to as the transport matrix), *F*. In the following $x_{k,p}$ refers to the predicted state vector at detector *k*, while F_{k-1} refers to the transport matrix that propagates the track from detector *k-1* to detector *k*.
2. **Filtering:** once a prediction for the state vector at detector *k* is made, the KF looks for a measurement in detector *k*. If a measurement exists the state vector is "recalculated" to include the information from the new measurement. This is called the *filtering* step. In the formalism below $x_{k,F}$ stands for the filtered state vector at detector *k*. The process then continues by extrapolating the filtered state vector to the next detector layer (i.e. predicting the state vector at that layer as in step 1), *filtering* the state vector at this layer, and so on until the last detector is reached.
3. **Smoothing:** Once the last detector layer is reached the process is then reversed, and steps 1 and 2 are repeated this time by extrapolating backwards (from the last detector to the first detector). This process is referred to as *smoothing*. The reason for this step is to include all the measurement information in the calculation of the state vector at each detector layer. In the following the smoothed state vector at detector *k* is denoted by $x_{k,s}$.

I divided the steps required for this calculation in terms of "subroutines". A module, kalman_mod.f90 will contain the subroutines:

KalInit: intializes the KF.
KalPredict: predicts the state vector at detector k.
KalFilter: filters the state vector.
KalChi2: calculates the incremental and total χ^2.
KalSmooth: calculates the smoothed state vector.

Since there are several matrix operations in the KF, another convenient module to have is matrix_mod.f90, which would contain subroutines for the required matrix operations:

MatrixInvert: inverts a matrix.
MatrixTranspose: calculates a matrix transpose.

Two opertaors will be defined for this purpose **.invert.** and **.transpose.** so that **.invert.A** is the inverse of matrix A and **.transpose.A** is the transpose of martix A, to access these features the statement "USE matrix_mod" should appear where needed. Fortran 90 already has generalizations of the (+) and (-) operators so that when two matrices are added/subtracted using the (+/-) operator their corresponding elements are added/subtracted. Adding/subtracting a constant to/from a matrix using the (+/-) operator adds the constant to each element of the matrix, and multiplying a matrix by a constant using the (*) operator multiplies each element of the matrix by that constant. However, note that multiplying two matrices using the (*) multiplies their corresponding elements, which is not the mathematical definition of matrix multiplication. Instead the FORTRAN 90 transfomational function **MATMUL(A,B)** should be used to multiply two matrices A and B.

STEP 1. Initialize the covariance matrix: (kalInit)

1. Initialize the state Vector, $x_{1,P}$ (from previous code).
2. Initialize the covariance matrix, $C_{1,P.}$

STEP 2. Extrapolate the state vector and covariance matrix: (KalPredict)

1. Calculate the transport matrix (propagator), $F_{k-1.}$
2. Calculate the measurement error matrix, V_k and it's inverse, G_K

$$G_k = V_k^{-1}$$

3. Calculate the measurement matrix, $H_{k.}$
4. Extrapolate the state vector to detector layer k

$$x_{k,p} = F_{k-1} x_{k-1,F}$$

5. Calculate the process noise matrix (multiple scattering), Q_{k-1}.
6. Calculate the covariance matrix

$$C_{k,p} = F_{k-1} C_{k-1,F} F_{k-1}^T + Q_{k-1}$$

7. Calculate the prediction residuals

$$r_{k,p} = m_k - H_k x_{k,p}$$

8. Calculate the covariance matrix of predicted residuals

$$R_{k,p} = V_k + H_k C_{k,p} H_k^T$$

STEP 3. Update the state vector with the new measurement: (KalFilter)

1. Calculate the Kalman gain matrix

$$K_k = C_{k,p} H_k^T \left(V_k + H_k C_{k,p} H_k^T \right)^{-1}$$

2. Update the state vector

$$x_{k,F} = x_{k,p} + K_k (m_k - H_k x_{k,p})$$

3. Update the covariance matrix

$$C_{k,F} = (I - K_k H_k) C_{k,p}$$

4. Calculate the filtered residuals

$$r_{k,F} = m_k - H_k x_{k,F}$$

5. Calculate the covariance matrix of filtered residuals

$$R_{k,F} = V_k - H_k C_k H_k^T$$

STEP 4. Calculate χ^2: (KalChi2)

1. Calculate the χ^2 increment

$$\chi_k^{\ 2} = r_{k,F}^T R_{k,F}^{-1} r_k$$

2. Update the total χ^2

$$\chi_{k,tot}^2 = \chi_{k-1,tot}^2 + \chi_k^2$$

STEP 5. Smooth the state vector: (KalSmooth)

1. Calculate the smoother gain matrix

$$A_k = C_{k,F} F_k^T (C_{k+1,p})^{-1}$$

2. Calculate the smoothed state vector

$$x_{k,s} = x_{k,F} + A_k (x_{k+1,s} - x_{k+1,F})$$

3. Calculate the smoothed residuals

$$r_{k,s} = m_k - H_k x_{k,s}$$