

The Kalman-Bucy Filter

Alexander Wittmond

University of Missouri

October 20, 2020

Table of Contents

1 The Filtering Problem

2 The Kalman-Bucy Filter

3 An Example

Classical Models

Classically, time dependent systems are modeled by:

Classical Models

Classically, time dependent systems are modeled by:
Differential equations

$$f : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}^n \quad (1)$$

$$\frac{dx}{dt} = f(x, t) \quad (2)$$

Classical Models

Classically, time dependent systems are modeled by:
Differential equations

$$f : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}^n \quad (1)$$

$$\frac{dx}{dt} = f(x, t) \quad (2)$$

Solved by:

$$x : \mathbb{R} \rightarrow \mathbb{R}^n \quad (3)$$

The problem of noise

- reality is more complex

The problem of noise

- reality is more complex
- measurements of a system have a maximum resolution

The problem of noise

- reality is more complex
- measurements of a system have a maximum resolution
- beyond that resolution we see random noise caused by unknown variables

The problem of noise

- reality is more complex
- measurements of a system have a maximum resolution
- beyond that resolution we see random noise caused by unknown variables
- so our observations are better modeled by a random variable X on some probability space (Ω, P) than a point in \mathbb{R}^n



Figure: An example of a noisy measurement

The

Kalman-Bucy
Filter

Alexander
Wittmond

The Filtering
Problem

The
Kalman-Bucy
Filter

An Example

Stochastic Models

Stochastic Models

Stochastic differential equations

$$dx_t = f(x_t, t)dt + g(x_t, t)d\beta_t \quad (4)$$

meaning

$$x_t - x_0 = \int_0^t f(x_t, t)dt + \int_0^t g(x_t, t)d\beta_t \quad (5)$$

Stochastic Models

Stochastic differential equations

$$dx_t = f(x_t, t)dt + g(x_t, t)d\beta_t \quad (4)$$

meaning

$$x_t - x_0 = \int_0^t f(x_t, t)dt + \int_0^t g(x_t, t)d\beta_t \quad (5)$$

Solved by a stochastic process

$$X : (\Omega, P) \times \mathbb{R} \rightarrow \mathbb{R}^n \quad (6)$$

$$\{X_t\}_{t \in \mathbb{R}} \quad (7)$$

Stochastic Model of Measurement

The Filtering
Problem

The
Kalman-Bucy
Filter

An Example

In general we cannot measure a system directly and our measurement may be subject to noise.

Stochastic Model of Measurement

In general we cannot measure a system directly and our measurement may be subject to noise.

$$dx_t = f(x_t, t)dt + g(x_t, t)d\beta_t \quad (8)$$

$$y_n = h(x_{t_n}) + \nu_n \quad (9)$$

Stochastic Model of Measurement

In general we cannot measure a system directly and our measurement may be subject to noise.

$$dx_t = f(x_t, t)dt + g(x_t, t)d\beta_t \quad (8)$$

$$y_n = h(x_{t_n}) + \nu_n \quad (9)$$

We will assume our measurement is subject to white noise

$$\nu_n \sim N(0, R_k) \text{ i.i.d} \quad (10)$$

The Filtering Problem and its Solution

The
Kalman-Bucy
Filter

Alexander
Wittmond

The Filtering
Problem

The
Kalman-Bucy
Filter

An Example

Given a set of observations $\{Y_{t_1}, \dots, Y_{t_n}\} = Y_n$ we want to estimate the distribution of X_t .

The Filtering Problem and its Solution

The
Kalman-Bucy
Filter

Alexander
Wittmond

The Filtering
Problem

The
Kalman-Bucy
Filter

An Example

Given a set of observations $\{Y_{t_1}, \dots, Y_{t_n}\} = Y_n$ we want to estimate the distribution of X_t .

The Filtering Problem and its Solution

Given a set of observations $\{Y_{t_1}, \dots, Y_{t_n}\} = Y_n$ we want to estimate the distribution of X_t .

- **Smoothing** $t < t_n$

The Filtering Problem and its Solution

Given a set of observations $\{Y_{t_1}, \dots, Y_{t_n}\} = Y_n$ we want to estimate the distribution of X_t .

- **Smoothing** $t < t_n$
- **Filtering** $t = t_n$

The Filtering Problem and its Solution

Given a set of observations $\{Y_{t_1}, \dots, Y_{t_n}\} = Y_n$ we want to estimate the distribution of X_t .

- **Smoothing** $t < t_n$
- **Filtering** $t = t_n$
- **Prediction** $t > t_n$

The Filtering Problem and its Solution

Given a set of observations $\{Y_{t_1}, \dots, Y_{t_n}\} = Y_n$ we want to estimate the distribution of X_t .

- **Smoothing** $t < t_n$
- **Filtering** $t = t_n$
- **Prediction** $t > t_n$

We want to compute

$$p(x, t \mid Y_n) \tag{11}$$

The Filtering Problem and its Solution

Given a set of observations $\{Y_{t_1}, \dots, Y_{t_n}\} = Y_n$ we want to estimate the distribution of X_t .

- **Smoothing** $t < t_n$
- **Filtering** $t = t_n$
- **Prediction** $t > t_n$

We want to compute

$$p(x, t \mid Y_n) \quad (11)$$

The minimum variance estimate of X_t is

$$\mathbb{E}[x \mid Y_n] \quad (12)$$

the mean of $p(x, t \mid Y_n)$

Table of Contents

- 1 The Filtering Problem
- 2 The Kalman-Bucy Filter
- 3 An Example

The Linear Problem

Linear systems are given by:

The Linear Problem

Linear systems are given by:

$$dx_t = F(t)x_t dt + G(t)d\beta_t \quad (13)$$

$$y_k = M(t_k)x_{t_k} + \nu_k \quad (14)$$

$$\mathbb{E}[d\beta_t d\beta_t^T] = Q(t)dt, \quad \nu_k \sim N(0, R_k) \quad (15)$$

The Linear Problem

Linear systems are given by:

$$dx_t = F(t)x_t dt + G(t)d\beta_t \quad (13)$$

$$y_k = M(t_k)x_{t_k} + \nu_k \quad (14)$$

$$\mathbb{E}[d\beta_t d\beta_t^T] = Q(t)dt, \quad \nu_k \sim N(0, R_k) \quad (15)$$

Characterization of Linear Systems

Characterization of Linear Systems

- Linear systems are Gauss Markov processes with means described by deterministic linear systems.

$$x_t \sim N(\hat{x}_t, P_t) \quad (16)$$

Characterization of Linear Systems

- Linear systems are Gauss Markov processes with means described by deterministic linear systems.

$$x_t \sim N(\hat{x}_t, P_t) \quad (16)$$

- We only need to know the mean and variance of the system at a given time to characterize $P(x, t)$

Characterization of Linear Systems

- Linear systems are Gauss Markov processes with means described by deterministic linear systems.

$$x_t \sim N(\hat{x}_t, P_t) \quad (16)$$

- We only need to know the mean and variance of the system at a given time to characterize $P(x, t)$

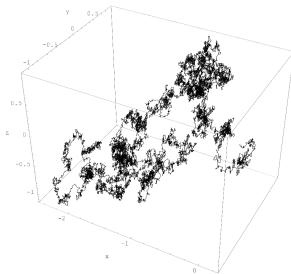


Figure: Three dimensional Brownian motion

The Solution To The Linear Discrete-Continuous Filtering Problem

The Kalman-Bucy filter consists of:

Evolution for

$$\text{Conditional Mean } \mathbb{E}[x_t \mid Y_{t_k}] = \hat{x}_t^{t_k} \quad (17)$$

$$\text{Conditional Variance } \mathbb{E}[(x_t - \hat{x}_t)(x_t - \hat{x}_t)^T \mid Y_{t_k}] = P_t^{t_k} \quad (18)$$

given some prior x_0 and P_0^0 .

The Solution To The Linear Discrete-Continuous Filtering Problem

The Kalman-Bucy filter consists of:

Prediction between observations

$$\frac{d}{dt}\hat{x}_t^{t_k} = F(t)\hat{x}_t^{t_k} \quad (17)$$

$$\frac{d}{dt}P_t^{t_k} = F(t)P_t^{t_k} + P_t^{t_k}F^T(t) + G(t)Q(t)G^T(t) \quad (18)$$

$$t_k \leq t < t_{k+1} \quad (19)$$

The Solution To The Linear Discrete-Continuous Filtering Problem

The Kalman-Bucy filter consists of:

Update at observations

$$\hat{x}_{t_k}^{t_k} = \hat{x}_{t_k}^{t_k-1} + K(t_k)(y_k - M(t_k)\hat{x}_{t_k}^{t_k-1}) \quad (17)$$

$$P_{t_k}^{t_k-1} = P_{t_k}^{t_k-1} - K(t_k)M(t_k)P_{t_k}^{t_k-1} \quad (18)$$

where the **Kalman Gain** $K(t)$ is given by

$$P_{t_k}^{t_k-1} M^T(t_k) [M(t_k) P_{t_k}^{t_k-1} M^T(t_k) + R_k]^{-1} \quad (19)$$

In Code

```

23 arma::dvec KalmanFilter::update(double t, arma::vec observation){
24
25     const arma::dmat M = measurement->measurement_matrix(t),
26         R = measurement->covariance(t);
27
28     //predict
29     const arma::dvec old_state = model->extrapolate( t );
30     const arma::dmat P = matrix_model->extrapolate(t);
31
32     // the Kalman gain
33     const arma::dmat K =
34         P * arma::trans(M) * arma::inv(M * P * arma::trans(M) + R);
35
36     // adjust
37     const arma::dmat KM = K * M;
38     const arma::dmat I_KM = arma::eye(KM.n_rows,KM.n_cols) - KM;
39
40     //correct with residual multiplied by Kalman Gain
41     const arma::dvec new_state = old_state + K*(observation - M*old_state);
42
43     //adjust covariance with the change in uncertainty caused by observation
44     const arma::dmat new_covariance = I_KM*P*arma::trans(I_KM) + K*R*arma::trans(K);
45
46     //now integrate from new initial estimates
47     model->set_initial_conditions(t , new_state );
48     matrix_model->set_initial_conditions(t , new_covariance);
49
50     return new_state;
51 }

```

Figure: The Kalman Filter update in C++

The

Kalman-Bucy
Filter

Alexander
Wittmond

The Filtering
Problem

The
Kalman-Bucy
Filter

An Example

Considerations

Considerations

- To integrate the prediction equations we need to fix the boundary conditions \hat{x}_{t_0}, P_{t_0}

Considerations

- To integrate the prediction equations we need to fix the boundary conditions \hat{x}_{t_0}, P_{t_0}
- Calculating the Kalman gain is $O(n^3)$ in the dimension of the measurement due to the matrix inversion

Considerations

- To integrate the prediction equations we need to fix the boundary conditions \hat{x}_{t_0}, P_{t_0}
- Calculating the Kalman gain is $O(n^3)$ in the dimension of the measurement due to the matrix inversion
- The Kalman gain can be precomputed

Considerations

- To integrate the prediction equations we need to fix the boundary conditions \hat{x}_{t_0}, P_{t_0}
- Calculating the Kalman gain is $O(n^3)$ in the dimension of the measurement due to the matrix inversion
- The Kalman gain can be precomputed
- The update is recursive so we only need to store our previous estimates at the time of the update

The Extended Kalman Filter

```
64 arma::dvec ExtendedKalmanFilter::update(double t, arma::vec observation) {
65     arma::dvec nominal_state = model->extrapolate(t );
66
67     arma::dvec measurement_error = observation - measurement->measure(t , nominal_state);
68
69     arma::dvec error = perturbationProcessFilter.update(t, measurement_error);
70
71     arma::dvec state_estimate = nominal_state + error;
72
73     model->set_initial_conditions(t , state_estimate);
74
75     perturbationProcessFilter.get_model()->set_initial_conditions
76         (t,arma::dvec(nominal_state.size(),arma::fill::zeros));
77
78
79     return state_estimate;
80 }
```

Figure: The Extended Kalman Filter update in C++

Pros

Pros and Cons

Pros and Cons

Pros

- Recursiveness causes low memory requirements

Pros and Cons

Pros

- Recursiveness causes low memory requirements
- Relatively low cost to computing an update

Pros and Cons

Pros

- Recursiveness causes low memory requirements
- Relatively low cost to computing an update
- Most computationally expensive items can be computed in advance

Pros and Cons

Pros

- Recursiveness causes low memory requirements
- Relatively low cost to computing an update
- Most computationally expensive items can be computed in advance

Cons

Pros and Cons

Pros

- Recursiveness causes low memory requirements
- Relatively low cost to computing an update
- Most computationally expensive items can be computed in advance

Cons

- Is not adaptive, relies on the accuracy of the underlying model

Pros and Cons

Pros

- Recursiveness causes low memory requirements
- Relatively low cost to computing an update
- Most computationally expensive items can be computed in advance

Cons

- Is not adaptive, relies on the accuracy of the underlying model
- Suffers from filter divergence

Pros and Cons

Pros

- Recursiveness causes low memory requirements
- Relatively low cost to computing an update
- Most computationally expensive items can be computed in advance

Cons

- Is not adaptive, relies on the accuracy of the underlying model
- Suffers from filter divergence
- Can not capture multi-modalities

Table of Contents

- 1 The Filtering Problem
- 2 The Kalman-Bucy Filter
- 3 An Example**

Orbit Determination

The simulation will be filtering noisy measurements from a deterministic system.

Orbit Determination

The simulation will be filtering noisy measurements from a deterministic system.

We have

$$dx_t = f(x_t, t)dt \quad (20)$$

$$y_k = h(x_{t_k}, t) + \nu_t \quad (21)$$

$$\nu_t \sim N(0, Q) \quad (22)$$

Orbit Determination

$$x_t = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} \quad (23)$$

$$f(x_t, t) = \begin{bmatrix} \dot{x} \\ \dot{y} \\ -\mu \frac{x}{x^2+y^2} \\ -\mu \frac{y}{x^2+y^2} \end{bmatrix} \quad (24)$$

If p gives the position of the sensor and

$$\bar{x} = \begin{bmatrix} x \\ y \end{bmatrix} \quad (25)$$

then

$$h(x_t, t) = \begin{bmatrix} \|\bar{x} - p\| \\ \frac{(x-p_1)\dot{x} + (y-p_2)\dot{y}}{\|\bar{x} - p\|} \\ \frac{x \cdot p}{\|x\|} \end{bmatrix} \quad (26)$$

$$Q_t = \begin{bmatrix} n1 & 0 & 0 \\ 0 & n2 & 0 \\ 0 & 0 & n3 \end{bmatrix} \quad (27)$$

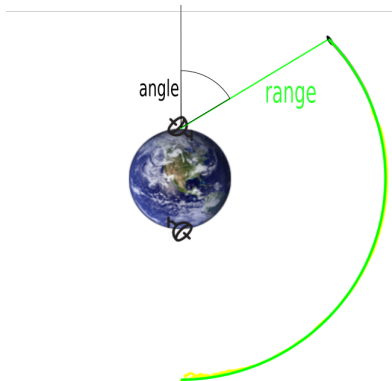


Figure: Measurements in Simulation