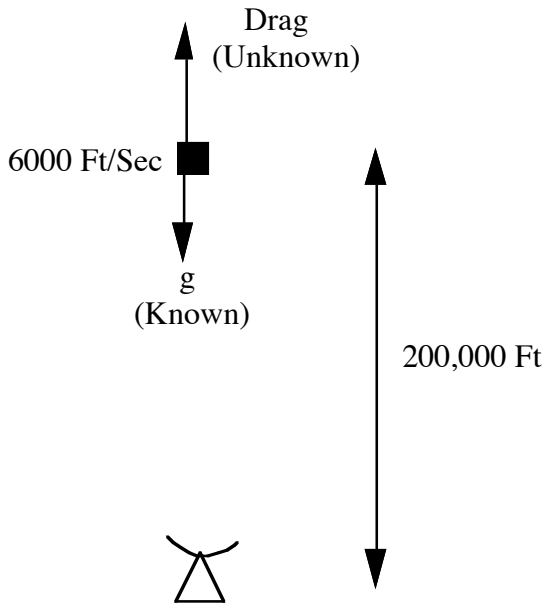
Drag and Falling Object

Drag and Falling Object
Overview

- Designing extended Kalman filter with ballistic coefficient as state
- Designing extended Kalman filter with inverse ballistic coefficient as state
- Importance of process noise
- Linear polynomial Kalman filter

Radar Tracking Falling Object in Presence of Unknown Drag



Recall

$$\ddot{x} = \text{Drag} - g = \frac{Q_p g}{\beta} - g$$

$$Q_p = .5\rho\dot{x}^2$$

$$\rho = .0034e^{-x/22000}$$

Nonlinear model of real world

$$\ddot{x} = \frac{Q_p g}{\beta} - g = \frac{.5g\rho\dot{x}^2}{\beta} - g = \frac{.0034ge^{-x/22000}\dot{x}^2}{2\beta} - g$$

Designing an Extended Kalman Filter With Ballistic Coefficient as State (Estimate Altitude, Velocity and β of Object)

Setting Up Linearized State Space Equation

Choice of states

$$\mathbf{x} = \begin{bmatrix} x \\ \dot{x} \\ \beta \end{bmatrix}$$

Differential equations involved

$$\ddot{x} = \frac{.0034 g e^{-x/22000} \dot{x}^2}{2\beta} - g$$

$$\dot{x} = \dot{x}$$

$$\dot{\beta} = u_s \longleftarrow \text{Process noise for safety}$$

We can linearize our model of the real world

$$\begin{bmatrix} \Delta\dot{x} \\ \Delta\ddot{x} \\ \Delta\dot{\beta} \end{bmatrix} = \begin{bmatrix} \frac{\partial\dot{x}}{\partial x} & \frac{\partial\dot{x}}{\partial\dot{x}} & \frac{\partial\dot{x}}{\partial\beta} \\ \frac{\partial\ddot{x}}{\partial x} & \frac{\partial\ddot{x}}{\partial\dot{x}} & \frac{\partial\ddot{x}}{\partial\beta} \\ \frac{\partial\dot{\beta}}{\partial x} & \frac{\partial\dot{\beta}}{\partial\dot{x}} & \frac{\partial\dot{\beta}}{\partial\beta} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta\dot{x} \\ \Delta\beta \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ u_s \end{bmatrix}$$

Defining and Evaluating Systems Dynamics Matrix

Systems dynamics matrix

$$\mathbf{F} = \left[\begin{array}{ccc} \frac{\partial \dot{x}}{\partial x} & \frac{\partial \dot{x}}{\partial \dot{x}} & \frac{\partial \dot{x}}{\partial \beta} \\ \frac{\partial \ddot{x}}{\partial x} & \frac{\partial \ddot{x}}{\partial \dot{x}} & \frac{\partial \ddot{x}}{\partial \beta} \\ \frac{\partial \dot{\beta}}{\partial x} & \frac{\partial \dot{\beta}}{\partial \dot{x}} & \frac{\partial \dot{\beta}}{\partial \beta} \end{array} \right]_{\mathbf{x}=\hat{\mathbf{x}}}$$

Process noise matrix

$$\mathbf{Q} = E(\mathbf{w}\mathbf{w}^{\mathbf{T}}) \longrightarrow \mathbf{Q}(t) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \Phi_s \end{bmatrix}$$

Evaluate partial derivatives for first equation

$$\dot{x} = \dot{x}$$

$$\frac{\partial \dot{x}}{\partial x} = 0$$

$$\frac{\partial \dot{x}}{\partial \dot{x}} = 1$$

$$\frac{\partial \dot{x}}{\partial \beta} = 0$$

Evaluating More Partial Derivatives

Evaluate partial derivatives for second equation

$$\ddot{x} = \frac{.0034 g e^{-x/22000} \dot{x}^2}{2\beta} - g$$

$$\frac{\partial \ddot{x}}{\partial x} = \frac{-.0034 e^{-x/22000} \dot{x}^2 g}{2\beta(22000)} = \frac{-\rho g \dot{x}^2}{44000\beta}$$

$$\frac{\partial \ddot{x}}{\partial \dot{x}} = \frac{2*.0034 e^{-x/22000} \dot{x} g}{2\beta} = \frac{\rho g \dot{x}}{\beta}$$

$$\frac{\partial \ddot{x}}{\partial \beta} = \frac{-.0034 e^{-x/22000} \dot{x}^2 g}{2\beta^2} = \frac{-\rho g \dot{x}^2}{2\beta^2}$$

Evaluate partial derivatives for third equation

$$\dot{\beta} = u_s$$

$$\frac{\partial \dot{\beta}}{\partial x} = 0$$

$$\frac{\partial \dot{\beta}}{\partial \dot{x}} = 0$$

$$\frac{\partial \dot{\beta}}{\partial \beta} = 0$$

Shorthand Notation For Systems Dynamics Matrix

Systems dynamics matrix turns out to be

$$\mathbf{F} = \begin{bmatrix} 0 & 1 & 0 \\ \dfrac{-\hat{\rho} g \dot{\hat{x}}^2}{44000\hat{\beta}} & \dfrac{\hat{\rho} g \dot{\hat{x}}}{\hat{\beta}} & \dfrac{-\hat{\rho} g \dot{\hat{x}}^2}{2\hat{\beta}^2} \\ 0 & 0 & 0 \end{bmatrix}$$

Where $\hat{\rho} = .0034e^{-\hat{x}/22000}$

If we define

$$f_{21} = \frac{-\hat{\rho} g \dot{\hat{x}}^2}{44000\hat{\beta}}$$

$$f_{22} = \frac{\hat{\rho} \dot{\hat{x}} g}{\hat{\beta}}$$

$$f_{23} = \frac{-\hat{\rho} g \dot{\hat{x}}^2}{2\hat{\beta}^2}$$

$$\longrightarrow \quad \mathbf{F}(t) = \begin{bmatrix} 0 & 1 & 0 \\ f_{21} & f_{22} & f_{23} \\ 0 & 0 & 0 \end{bmatrix}$$

Finding Fundamental Matrix

We can use Taylor series to find fundamental matrix

$$\Phi(t) = \mathbf{I} + \mathbf{F}t + \frac{\mathbf{F}^2t^2}{2!} + \frac{\mathbf{F}^3t^3}{3!} + \ldots \quad \textbf{where} \quad \mathbf{F}(t) = \begin{bmatrix} 0 & 1 & 0 \\ f_{21} & f_{22} & f_{23} \\ 0 & 0 & 0 \end{bmatrix}$$

Using first two terms

$$\Phi(t) \approx \mathbf{I} + \mathbf{F}t = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ f_{21} & f_{22} & f_{23} \\ 0 & 0 & 0 \end{bmatrix} t = \begin{bmatrix} 1 & t & 0 \\ f_{21}t & 1+f_{22}t & f_{23}t \\ 0 & 0 & 1 \end{bmatrix}$$

$$\Phi(t) \approx \begin{bmatrix} 1 & t & 0 \\ f_{21}t & 1+f_{22}t & f_{23}t \\ 0 & 0 & 1 \end{bmatrix}$$

Finding discrete fundamental matrix

$$\Phi_k \approx \begin{bmatrix} 1 & T_s & 0 \\ f_{21}T_s & 1+f_{22}T_s & f_{23}T_s \\ 0 & 0 & 1 \end{bmatrix}$$

Important Matrices Related to Measurement

Measurement equation is linear

$$x^* = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \beta \end{bmatrix} + v$$

Therefore measurement matrix is given by

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

Discrete measurement noise matrix is scalar

$$\mathbf{R_k} = E(\mathbf{v_k v_k^T})$$

$$\mathbf{R_k} = \sigma_v^2$$

Finding Discrete Process Noise Matrix

Recall continuous process noise matrix given by

$$\mathbf{Q}(t) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \Phi_s \end{bmatrix}$$

Discrete process noise matrix can be found from

$$\mathbf{Q_k} = \int_0^{T_s} \mathbf{\Phi}(\tau)\mathbf{Q}\mathbf{\Phi}^{\mathbf{T}}(\tau)\mathbf{dt}$$

Substitution yields

$$\mathbf{Q_k} = \Phi_s \int_0^{T_s} \begin{bmatrix} 1 & \tau & 0 \\ f_{21}\tau & 1+f_{22}\tau & f_{23}\tau \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & f_{21}\tau & 0 \\ \tau & 1+f_{22}\tau & 0 \\ 0 & f_{23}\tau & 0 \end{bmatrix} d\tau$$

After some algebra

$$\mathbf{Q_k} = \Phi_s \int_0^{T_s} \begin{bmatrix} 0 & 0 & 0 \\ 0 & f_{23}^2\tau^2 & f_{23}\tau \\ 0 & f_{23}\tau & 1 \end{bmatrix} d\tau \longrightarrow \mathbf{Q_k} = \Phi_s \begin{bmatrix} 0 & 0 & 0 \\ 0 & f_{23}^2\frac{T_s^3}{3} & f_{23}\frac{T_s^2}{2} \\ 0 & f_{23}\frac{T_s^2}{2} & T_s \end{bmatrix}$$

Final answer

Extended Kalman Filter Structure

Integrate to propagate states

$$\bar{\ddot{x}}_{k-1} = \frac{.0034 g e^{-\hat{x}_{k-1}/22000}\, \hat{\dot{x}}^2_{k-1}}{2\hat{\beta}_{k-1}} - g$$

Filtering equations

$$\hat{x}_k = \bar{x}_k + K_{1_k}(x^*_k - \bar{x}_k)$$

$$\hat{\dot{x}}_k = \bar{\dot{x}}_k + K_{2_k}(x^*_k - \bar{x}_k)$$

$$\hat{\beta}_k = \hat{\beta}_{k-1} + K_{3_k}(x^*_k - \bar{x}_k)$$

Barred quantities obtained by integrating

True BASIC Simulation of Extended Kalman Filter-1

```
OPTION NOLET
REM UNSAVE "DATFIL"
REM UNSAVE "COVFIL"
OPEN #1:NAME "DATFIL",ACCESS OUTPUT,CREATE NEW, ORGANIZATION TEXT
OPEN #2:NAME "COVFIL",ACCESS OUTPUT,CREATE NEW, ORGANIZATION TEXT
SET #1: MARGIN 1000
SET #2: MARGIN 1000
DIM PHI(3,3),P(3,3),M(3,3),PHIP(3,3),PHIPPHIT(3,3),GAIN(3,1)
DIM Q(3,3),HMAT(1,3),HM(1,3),MHT(3,1)
DIM PHIT(3,3)
DIM HMHT(1,1),HT(3,1),KH(3,3),IDNP(3,3),IKH(3,3)
ITERM=1
SIGNOISE=25.
X=200000.
XD=-6000.
BETA=500.
XH=200025.
XDH=-6150.
BETAH=800.
ORDER=3
TS=.1
TF=30.
PHIS=0.
T=0.
S=0.
H=.001
HP=.001
MAT PHI=ZER(ORDER,ORDER)
MAT P=ZER(ORDER,ORDER)
MAT IDNP=IDN(ORDER,ORDER)
MAT Q=ZER(ORDER,ORDER)
P(1,1)=SIGNOISE*SIGNOISE
P(2,2)=20000.
P(3,3)=300.^2
MAT HMAT=ZER(1,ORDER)
MAT HT=ZER(ORDER,1)
HMAT(1,1)=1.
HT(1,1)=1.
```

Initial actual and estimated states

Initial covariance matrix

Measurement matrix and transpose

True BASIC Simulation of Extended Kalman Filter-2

```
DO WHILE T<=TF
        XOLD=X
        XDOLD=XD
        XDD=.0034*32.2*XD*XD*EXP(-X/22000.)/(2.*BETA)-32.2
        X=X+H*XD
        XD=XD+H*XDD
        T=T+H
        XDD=.0034*32.2*XD*XD*EXP(-X/22000.)/(2.*BETA)-32.2
        X=.5*(XOLD+X+H*XD)
        XD=.5*(XDOLD+XD+H*XDD)
        S=S+H
        IF S>=(TS-.00001) THEN
                S=0.
                RHOH=.0034*EXP(-XH/22000.)
                F21=-32.2*RHOH*XDH*XDH/(44000.*BETAH)
                F22=RHOH*32.2*XDH/BETAH
                F23=-RHOH*32.2*XDH*XDH/(2.*BETAH*BETAH)
                IF ITERM=1 THEN
                        PHI(1,1)=1.
                        PHI(1,2)=TS
                        PHI(2,1)=F21*TS
                        PHI(2,2)=1.+F22*TS
                        PHI(2,3)=F23*TS
                        PHI(3,3)=1.
                ELSE
                        PHI(1,1)=1.+.5*TS*TS*F21
                        PHI(1,2)=TS+.5*TS*TS*F22
                        PHI(1,3)=.5*TS*TS*F23
                        PHI(2,1)=F21*TS+.5*TS*TS*F22*F21
                        PHI(2,2)=1.+F22*TS+.5*TS*TS*(F21+F22*F22)
                        PHI(2,3)=F23*TS+.5*TS*TS*F22*F23
                        PHI(3,3)=1.
                END IF
                Q(2,2)=F23*F23*PHIS*TS*TS*TS/3.
                Q(2,3)=F23*PHIS*TS*TS/2.
                Q(3,2)=Q(2,3)
                Q(3,3)=PHIS*TS
                MAT PHIT=TRN(PHI)
```

Second-order Runge-Kutta integration on nonlinear differential equation

Non zero elements of systems dynamics matrix

Two or three term approximation to fundamental matrix

Non zero elements of process noise matrix

True BASIC Simulation of Extended Kalman Filter-3

```
                MAT PHIP=PHI*P
                MAT PHIPPHIT=PHIP*PHIT
                MAT M=PHIPPHIT+Q
                MAT HM=HMAT*M
                MAT HMHT=HM*HT
                HMHTR=HMHT(1,1)+SIGNOISE*SIGNOISE
                HMHTRINV=1./HMHTR
                MAT MHT=M*HT
                MAT GAIN=HMHTRINV*MHT
                MAT KH=GAIN*HMAT
                MAT IKH=IDNP-KH
                MAT P=IKH*M
                CALL GAUSS(XNOISE,SIGNOISE)
                CALL PROJECT(T,TS,XH,XDH,BETAH,XB,XDB,XDDB,HP)
                RES=X+XNOISE-XB
                XH=XB+GAIN(1,1)*RES
                XDH=XDB+GAIN(2,1)*RES
                BETAH=BETAH+GAIN(3,1)*RES
                ERRX=X-XH
                SP11=SQR(P(1,1))
                ERRXD=XD-XDH
                SP22=SQR(P(2,2))
                ERRBETA=BETA-BETAH
                SP33=SQR(P(3,3))
                PRINT T,X,XH,XD,XDH,BETA,BETAH
                PRINT #1:T,X,XH,XD,XDH,BETA,BETAH
                PRINT #2:T,ERRX,SP11,-SP11,ERRXD,SP22,-SP22,ERRBETA,SP33,-SP33
        END IF
LOOP
CLOSE #1
CLOSE #2
END
```

Riccati equations

Project states forward

Extended Kalman filter

True BASIC Simulation of Extended Kalman Filter-4

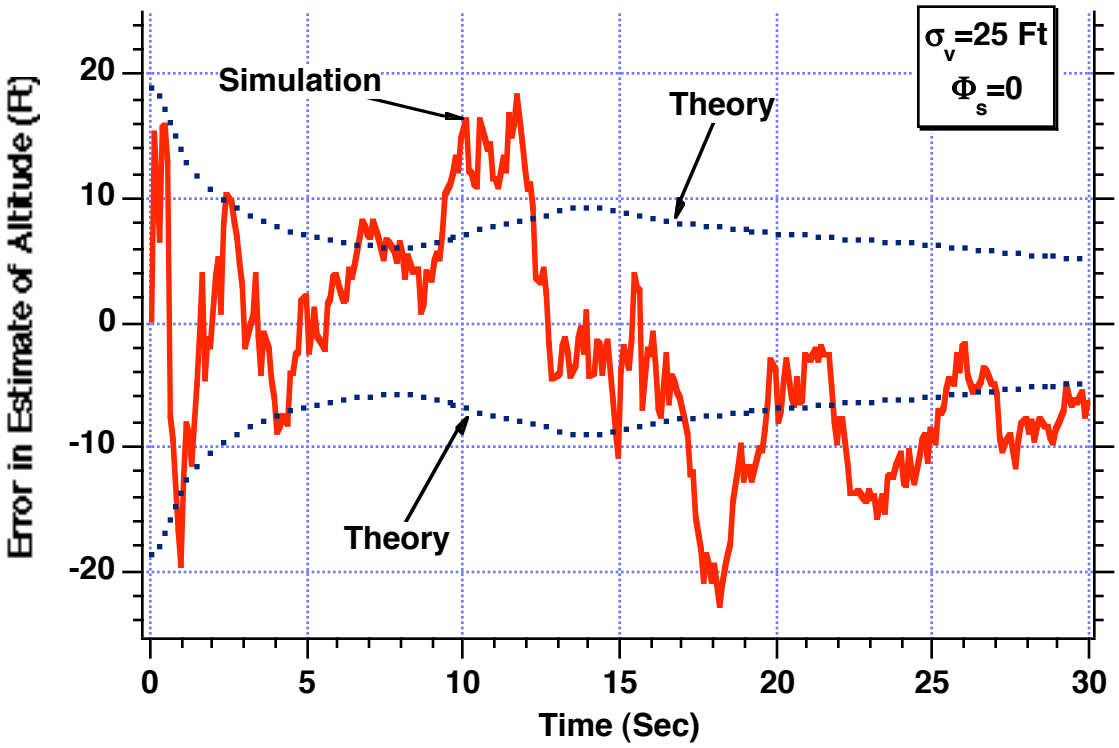
```
SUB PROJECT(TP,TS,XP,XDP,BETAP,XH,XDH,XDDH,HP)
T=0.
X=XP
XD=XDP
BETA=BETAP
H=HP
DO WHILE T<=(TS-.0001)
        XDD=.0034*32.2*XD*XD*EXP(-X/22000.)/(2.*BETA)-32.2
        XD=XD+H*XDD
        X=X+H*XD
        T=T+H
LOOP
XH=X
XDH=XD
XDDH=XDD
END SUB
```

Integrating nonlinear differential equation forward one sampling Interval with Euler integration

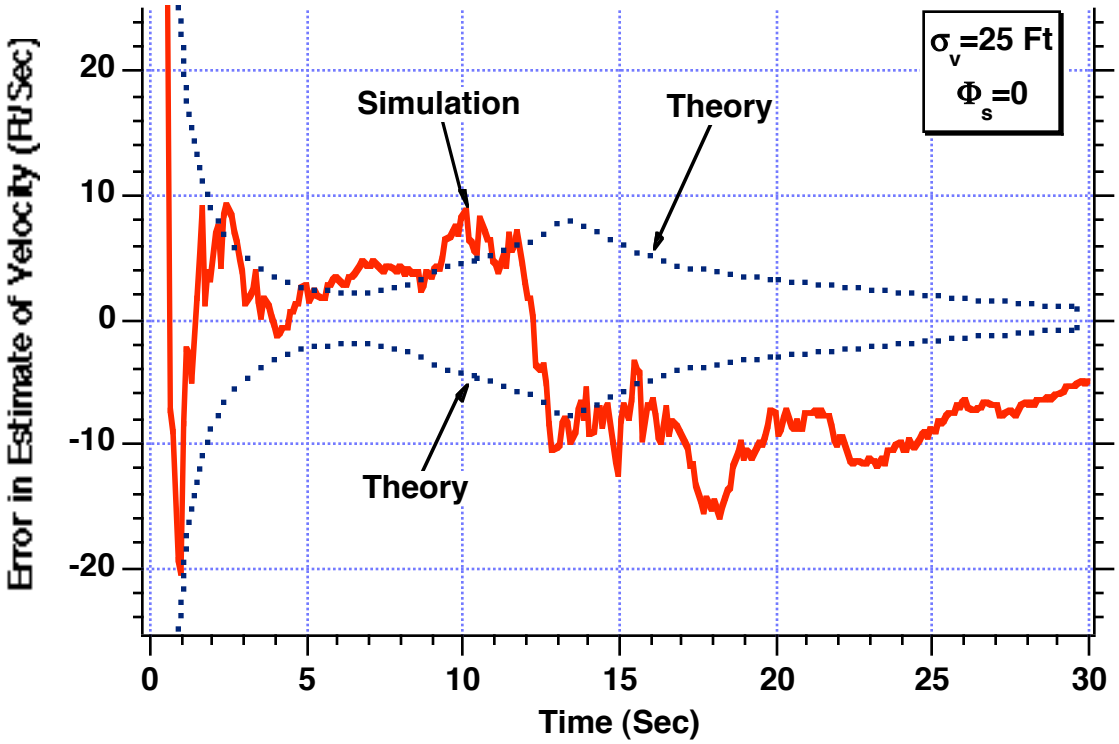
```
SUB GAUSS(X,SIG)
LET X=RND+RND+RND+RND+RND+RND-3
LET X=1.414*X*SIG
END SUB
```

Generate zero-mean Gaussian distributed random numbers

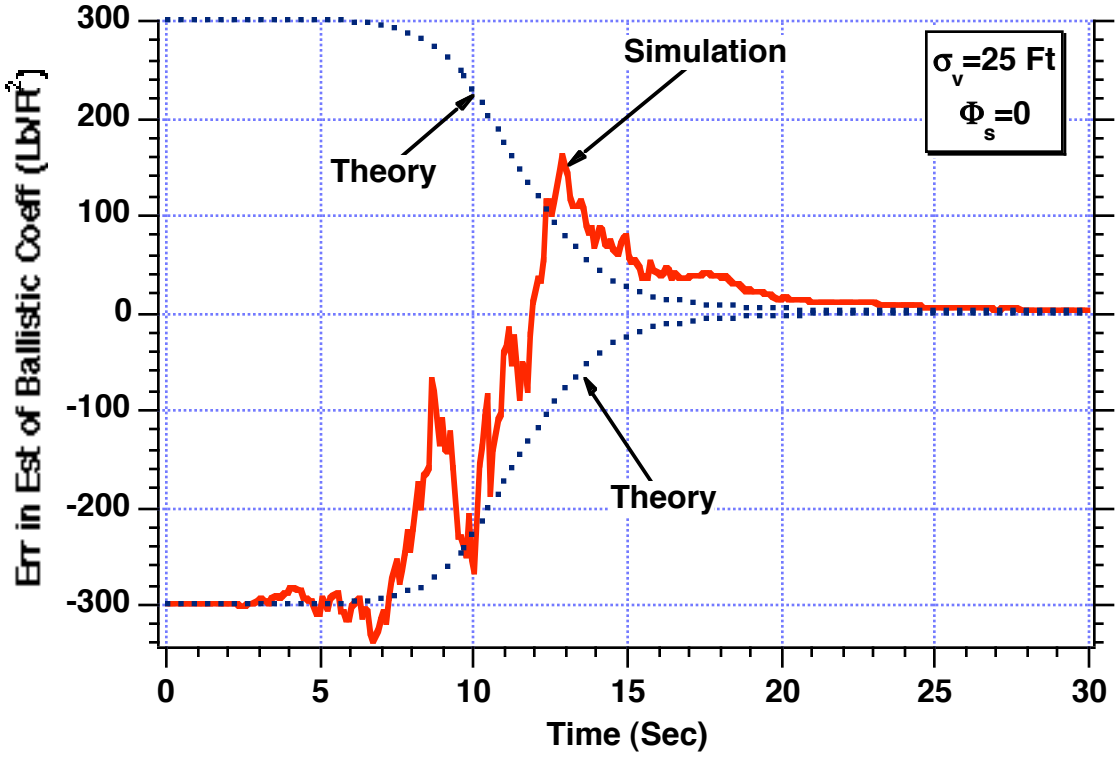
Error in Estimate of Altitude is Within Theoretical Bounds



Error in Estimate of Velocity Appears to Have Hangoff Error

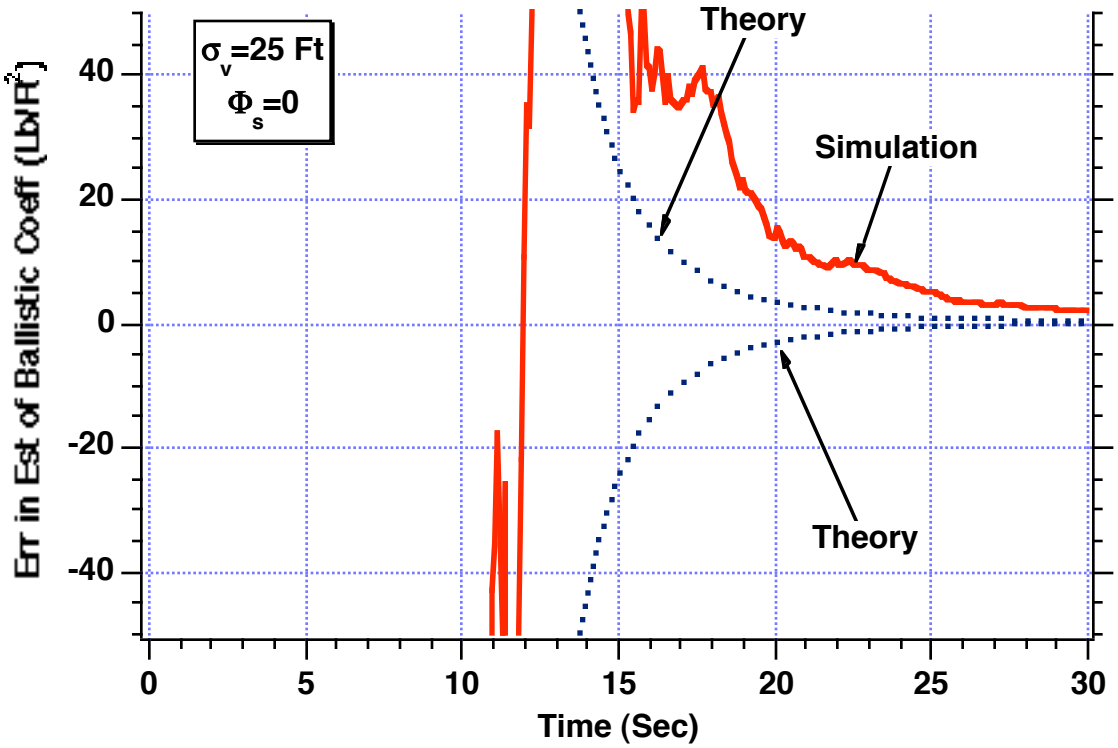


It Takes More Than Ten Seconds to Reduce Errors in Estimating Ballistic Coefficient



There is very little drag at higher altitudes (first 10 sec)

Expanding Scales Shows That There is also Hangoff Error in Estimating Ballistic Coefficient



Adding Another Term to Fundamental Matrix

Recall

$$\mathbf{F}(t) = \begin{bmatrix} 0 & 1 & 0 \\ f_{21} & f_{22} & f_{23} \\ 0 & 0 & 0 \end{bmatrix}$$

Three term Taylor series expansion

$$\mathbf{\Phi}(t) = \mathbf{I} + \mathbf{F}t + \frac{\mathbf{F}^2 t^2}{2!}$$

$$\mathbf{\Phi}_3 = \begin{bmatrix} 1 & t & 0 \\ f_{21}t & 1+f_{22}t & f_{23}t \\ 0 & 0 & 1 \end{bmatrix} + \frac{\mathbf{F}^2 t^2}{2}$$

Squaring systems dynamics matrix

$$\mathbf{F}^2 = \begin{bmatrix} 0 & 1 & 0 \\ f_{21} & f_{22} & f_{23} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ f_{21} & f_{22} & f_{23} \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} f_{21} & f_{22} & f_{23} \\ f_{22}f_{21} & f_{21}+f_{22}^2 & f_{22}f_{23} \\ 0 & 0 & 0 \end{bmatrix}$$

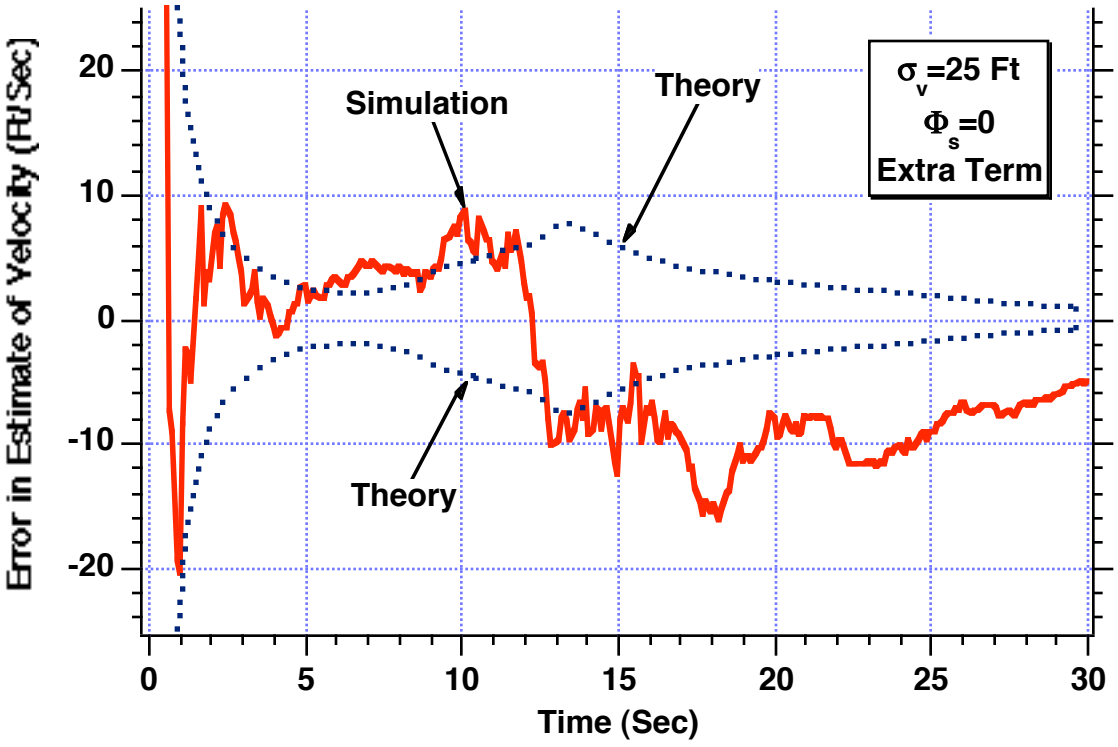
We finally obtain

$$\mathbf{\Phi}_3 = \begin{bmatrix} 1+f_{21}\frac{t^2}{2} & t+f_{22}\frac{t^2}{2} & f_{23}\frac{t^2}{2} \\ f_{21}t+f_{22}f_{21}\frac{t^2}{2} & 1+f_{22}t+(f_{21}+f_{22}^2)\frac{t^2}{2} & f_{23}t+f_{22}f_{23}\frac{t^2}{2} \\ 0 & 0 & 1 \end{bmatrix}$$

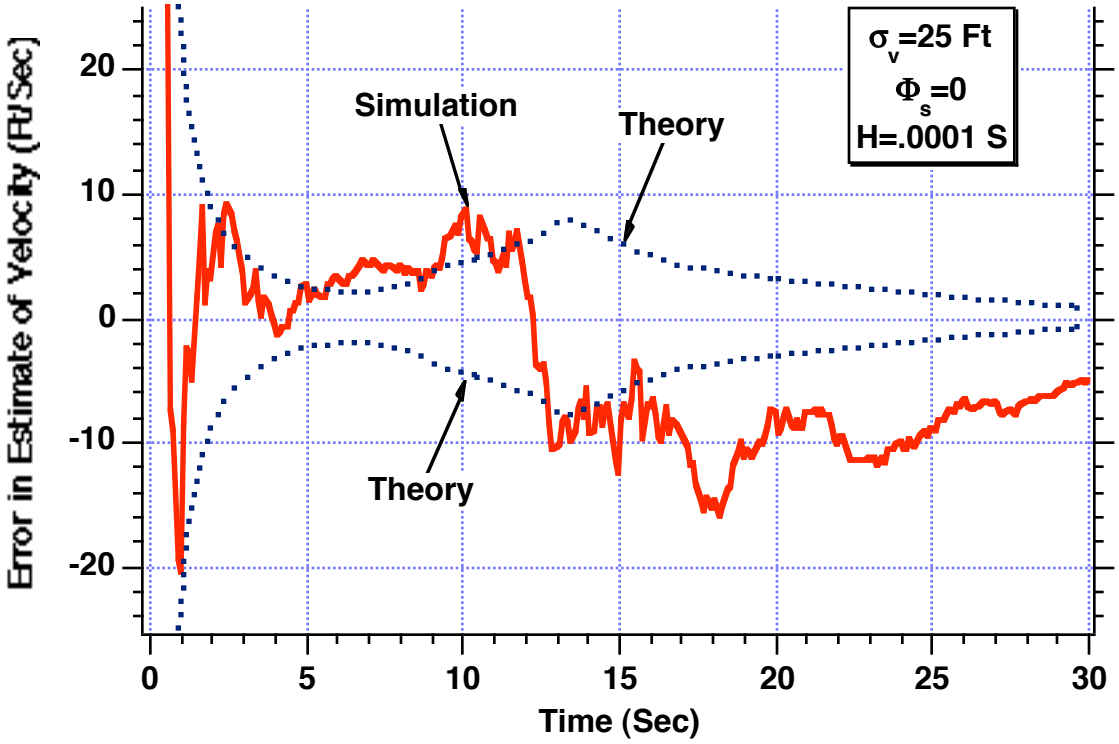
Discrete Three Term Fundamental Matrix

$$\Phi_{3_k} = \begin{bmatrix} 1+f_{21}\frac{T_s^2}{2} & T_s+f_{22}\frac{T_s^2}{2} & f_{23}\frac{T_s^2}{2} \\ f_{21}T_s+f_{22}f_{21}\frac{T_s^2}{2} & 1+f_{22}T_s+(f_{21}+f_{22}^2)\frac{T_s^2}{2} & f_{23}T_s+f_{22}f_{23}\frac{T_s^2}{2} \\ 0 & 0 & 1 \end{bmatrix}$$

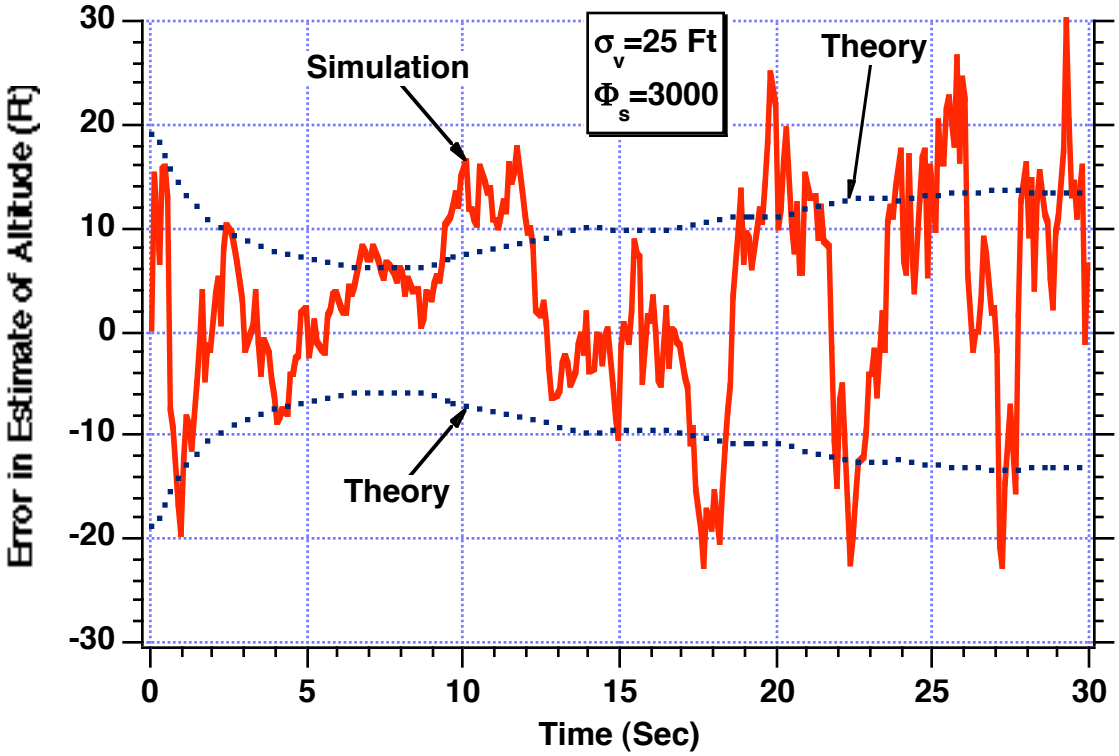
Adding Extra Term to Fundamental Matrix Does Not Remove Hangoff Error



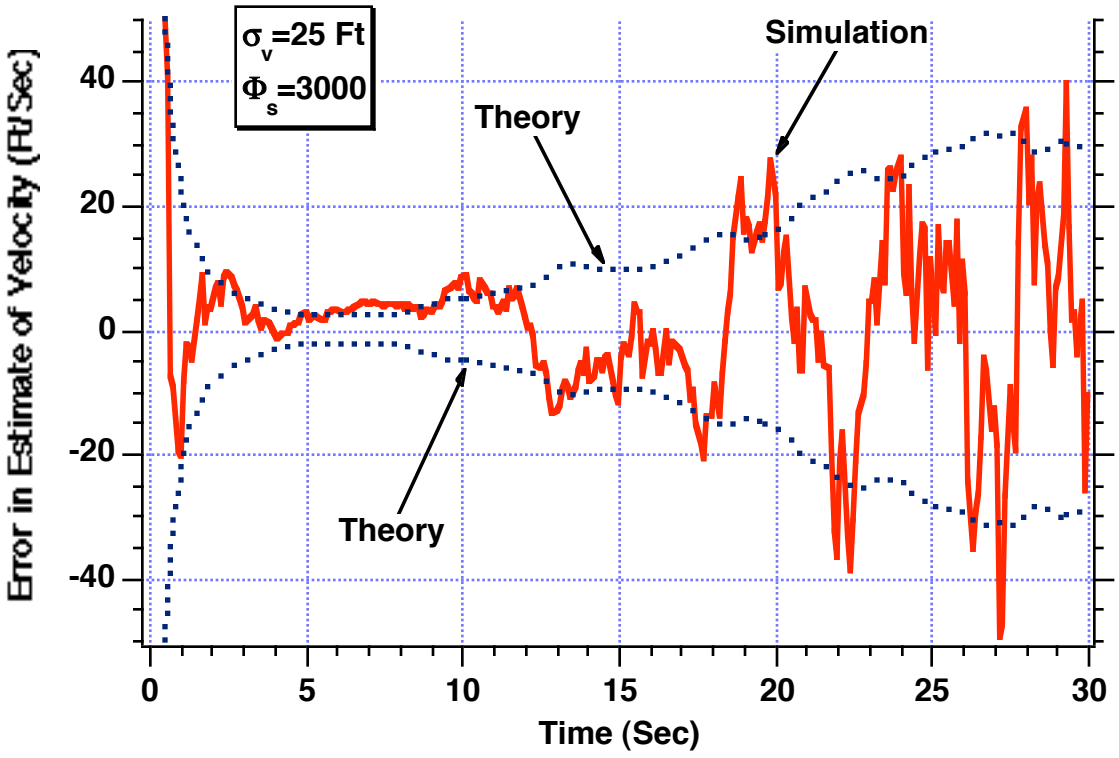
Making Integration Interval Ten Times Smaller in PROJECT Subroutine Does Not Remove Hangoff Error in Velocity Estimate



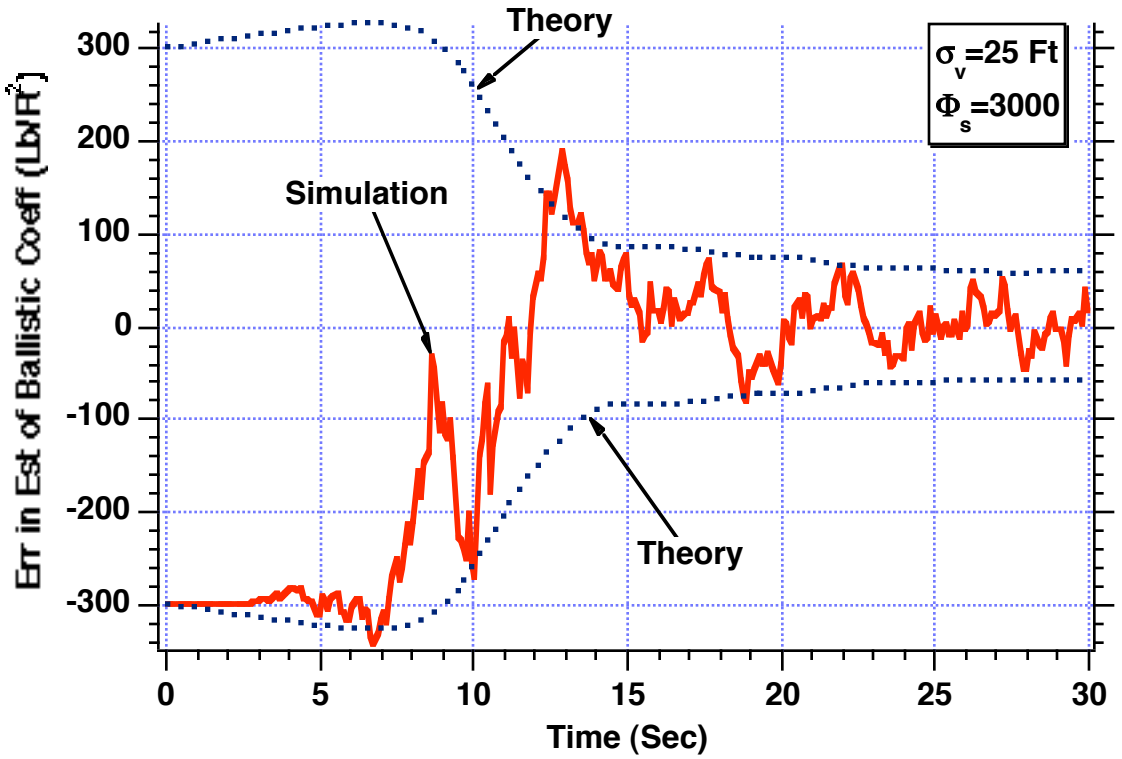
Adding Process Noise Increases Errors in Estimate of Altitude



Adding Process Noise Removes Hangoff Error at Expense of Increasing Errors in Estimate of Velocity



Adding Process Noise Removes Hangoff Error at Expense of Increasing Errors in Estimate of Ballistic Coefficient



Designing Extended Kalman Filter With Inverse Ballistic Coefficient as State

Choosing New States

Recall our model of real world

$$\ddot{x} = \frac{.0034 g e^{-x/22000} \dot{x}^2}{2\beta} - g = \left(\frac{1}{\beta}\right)\frac{.0034 g e^{-x/22000} \dot{x}^2}{2} - g$$

Alternate states

$$\mathbf{x} = \begin{bmatrix} x \\ \dot{x} \\ 1/\beta \end{bmatrix}$$

Ballistic coefficient is still constant but for safety

$$\dot{(1/\beta)} = u_s$$

Linearized equations in state space form

$$\begin{bmatrix} \Delta\dot{x} \\ \Delta\ddot{x} \\ \dot{\Delta(1/\beta)} \end{bmatrix} = \begin{bmatrix} \frac{\partial\dot{x}}{\partial x} & \frac{\partial\dot{x}}{\partial\dot{x}} & \frac{\partial\dot{x}}{\partial(1/\beta)} \\ \frac{\partial\ddot{x}}{\partial x} & \frac{\partial\ddot{x}}{\partial\dot{x}} & \frac{\partial\ddot{x}}{\partial(1/\beta)} \\ \frac{\partial\dot{(1/\beta)}}{\partial x} & \frac{\partial\dot{(1/\beta)}}{\partial\dot{x}} & \frac{\partial\dot{(1/\beta)}}{\partial(1/\beta)} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta\dot{x} \\ \Delta(1/\beta) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ u_s \end{bmatrix}$$

Obtaining Systems Dynamics Matrix-1

From previous slide

$$\mathbf{F} = \left[\begin{array}{ccc} \frac{\partial \dot{x}}{\partial x} & \frac{\partial \dot{x}}{\partial \dot{x}} & \frac{\partial \dot{x}}{\partial (1/\beta)} \\ \frac{\partial \ddot{x}}{\partial x} & \frac{\partial \ddot{x}}{\partial \dot{x}} & \frac{\partial \ddot{x}}{\partial (1/\beta)} \\ \frac{\partial \dot{(1/\beta)}}{\partial x} & \frac{\partial \dot{(1/\beta)}}{\partial \dot{x}} & \frac{\partial \dot{(1/\beta)}}{\partial (1/\beta)} \end{array} \right]_{\mathbf{x}=\hat{\mathbf{x}}}$$

First row

$$\dot{x} = \dot{x} \longrightarrow \frac{\partial \dot{x}}{\partial x} = 0 \qquad \frac{\partial \dot{x}}{\partial \dot{x}} = 1 \qquad \frac{\partial \dot{x}}{\partial (1/\beta)} = 0$$

Second row

$$\ddot{x} = \left(\frac{1}{\beta}\right) \frac{.0034 g e^{-x/22000} \dot{x}^2}{2} - g$$

$$\frac{\partial \ddot{x}}{\partial x} = \frac{-.0034 e^{-x/22000} \dot{x}^2 g}{2\beta(22000)} = \frac{-\rho g \dot{x}^2}{44000\beta} = \left(\frac{1}{\beta}\right)\left(\frac{-\rho g \dot{x}^2}{44000}\right)$$

$$\frac{\partial \ddot{x}}{\partial \dot{x}} = \frac{2*.0034 e^{-x/22000} \dot{x} g}{2\beta} = \frac{\rho g \dot{x}}{\beta} = \left(\frac{1}{\beta}\right) \rho g \dot{x}$$

$$\frac{\partial \ddot{x}}{\partial (1/\beta)} = \frac{.0034 e^{-x/22000} \dot{x}^2 g}{2} = \frac{\rho g \dot{x}^2}{2}$$

Obtaining Systems Dynamics Matrix-2

Third row

$$\dot{(1/\beta)} = u_s \longrightarrow \frac{\partial \dot{(1/\beta)}}{\partial x} = 0 \qquad \frac{\partial \dot{(1/\beta)}}{\partial \dot{x}} = 0 \qquad \frac{\partial \dot{(1/\beta)}}{\partial (1/\beta)} = 0$$

Substitution yields

$$\mathbf{F}(t) = \begin{bmatrix} 0 & 1 & 0 \\ f_{21} & f_{22} & f_{23} \\ 0 & 0 & 0 \end{bmatrix}$$

Where

$$f_{21} = \frac{-\hat{\rho} g \hat{\dot{x}}^2}{44000} \widehat{(1/\beta)}$$

$$f_{22} = \hat{\rho} \hat{\dot{x}} g \widehat{(1/\beta)}$$

$$f_{23} = \frac{-\hat{\rho} g \hat{\dot{x}}^2}{2}$$

Fundamental Matrix

Since

$$\mathbf{F}(t) = \begin{bmatrix} 0 & 1 & 0 \\ f_{21} & f_{22} & f_{23} \\ 0 & 0 & 0 \end{bmatrix}$$

Two term Taylor series expansion yields

$$\mathbf{\Phi}(t) \approx \mathbf{I} + \mathbf{F}t = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ f_{21} & f_{22} & f_{23} \\ 0 & 0 & 0 \end{bmatrix} t = \begin{bmatrix} 1 & t & 0 \\ f_{21}t & 1+f_{22}t & f_{23}t \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{\Phi}(t) \approx \begin{bmatrix} 1 & t & 0 \\ f_{21}t & 1+f_{22}t & f_{23}t \\ 0 & 0 & 1 \end{bmatrix}$$

Or in discrete form

$$\mathbf{\Phi}_k \approx \begin{bmatrix} 1 & T_s & 0 \\ f_{21}T_s & 1+f_{22}T_s & f_{23}T_s \\ 0 & 0 & 1 \end{bmatrix}$$

True BASIC Simulation of Alternate Filter-1

```
OPTION NOLET
REM UNSAVE "DATFIL"
REM UNSAVE "COVFIL"
OPEN #1:NAME "DATFIL",ACCESS OUTPUT,CREATE NEW, ORGANIZATION TEXT
OPEN #2:NAME "COVFIL",ACCESS OUTPUT,CREATE NEW, ORGANIZATION TEXT
SET #1: MARGIN 1000
SET #2: MARGIN 1000
DIM PHI(3,3),P(3,3),M(3,3),PHIP(3,3),PHIPPHIT(3,3),GAIN(3,1)
DIM Q(3,3),HMAT(1,3),HM(1,3),MHT(3,1)
DIM PHIT(3,3)
DIM HMHT(1,1),HT(3,1),KH(3,3),IDNP(3,3),IKH(3,3)
ITERM=1
G=32.2
SIGNOISE=25.
X=200000.
XD=-6000.
BETA=500.
XH=200025.
XDH=-6150.
BETAH=800.
BETAINV=1./BETA
BETAINVH=1./BETAH
ORDER=3
TS=.1
TF=30.
PHIS=0.
T=0.
S=0.
H=.001
HP=.001
MAT PHI=ZER(ORDER,ORDER)
MAT P=ZER(ORDER,ORDER)
MAT IDNP=IDN(ORDER,ORDER)
MAT Q=ZER(ORDER,ORDER)
P(1,1)=SIGNOISE*SIGNOISE
P(2,2)=20000.
P(3,3)=(BETAINV-BETAINVH)^2
MAT HMAT=ZER(1,ORDER)
MAT HT=ZER(ORDER,1)
HMAT(1,1)=1.
HT(1,1)=1.
```

Initial actual and estimated states

Initial covariance matrix

Measurement matrix and transpose

True BASIC Simulation of Alternate Filter-2

```
DO WHILE T<=TF
        XOLD=X
        XDOLD=XD
        XDD=.0034*32.2*XD*XD*EXP(-X/22000.)/(2.*BETA)-32.2
        X=X+H*XD
        XD=XD+H*XDD
        T=T+H
        XDD=.0034*32.2*XD*XD*EXP(-X/22000.)/(2.*BETA)-32.2
        X=.5*(XOLD+X+H*XD)
        XD=.5*(XDOLD+XD+H*XDD)
        S=S+H
        IF S>=(TS-.00001) THEN
                S=0.
                RHOH=.0034*EXP(-XH/22000.)
                F21=-G*RHOH*XDH*XDH*BETAINVH/44000.
                F22=RHOH*G*XDH*BETAINVH
                F23=.5*RHOH*XDH*XDH*G
                PHI(1,1)=1.
                PHI(1,2)=TS
                PHI(2,1)=F21*TS
                PHI(2,2)=1.+F22*TS
                PHI(2,3)=F23*TS
                PHI(3,3)=1.
                Q(2,2)=F23*F23*PHIS*TS*TS*TS/3.
                Q(2,3)=F23*PHIS*TS*TS/2.
                Q(3,2)=Q(2,3)
                Q(3,3)=PHIS*TS
                MAT PHIT=TRN(PHI)
                MAT PHIP=PHI*P
                MAT PHIPPHIT=PHIP*PHIT
                MAT M=PHIPPHIT+Q
                MAT HM=HMAT*M
                MAT HMHT=HM*HT
                HMHTR=HMHT(1,1)+SIGNOISE*SIGNOISE
                HMHTRINV=1./HMHTR
                MAT MHT=M*HT
                MAT GAIN=HMHTRINV*MHT
                MAT KH=GAIN*HMAT
                MAT IKH=IDNP-KH
                MAT P=IKH*M
```

Second-order Runge-Kutta integration on nonlinear differential equation

Non zero elements of systems dynamics matrix

Two term approximation to fundamental matrix

Non zero elements of process noise matrix

Riccati equations

True BASIC Simulation of Alternate Filter-3

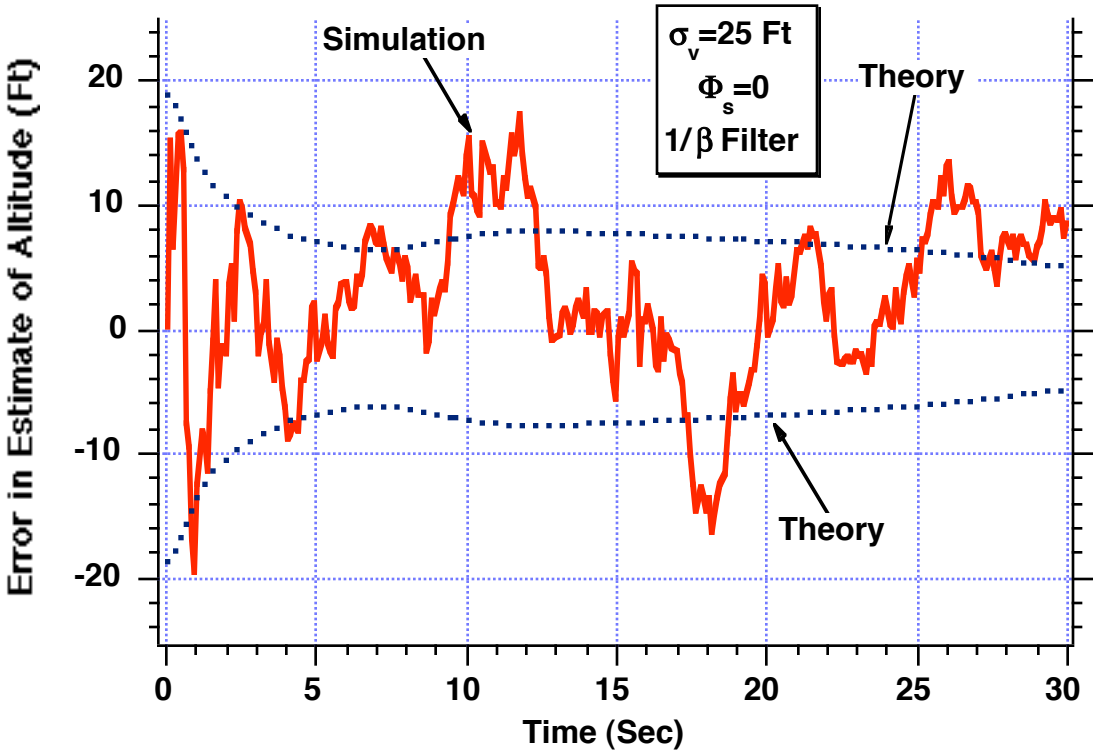
```
                    CALL GAUSS(XNOISE,SIGNOISE)
                    BETAH=1./BETAINVH
                    CALL PROJECT(T,TS,XH,XDH,BETAH,XB,XDB,XDDB,HP)
                    RES=X+XNOISE-XB
                    XH=XB+GAIN(1,1)*RES
                    XDH=XDB+GAIN(2,1)*RES
                    BETAINVH=BETAINVH+GAIN(3,1)*RES
                    ERRX=X-XH
                    SP11=SQR(P(1,1))
                    ERRXD=XD-XDH
                    SP22=SQR(P(2,2))
                    ERRBETAINV=1./BETA-BETAINVH
                    SP33=SQR(P(3,3))
                    BETAH=1./BETAINVH
                    PRINT T,X,XH,XD,XDH,BETA,BETAH
                    PRINT T,X,XH,XD,XDH,BETA,BETAH
                    PRINT #1:T,X,XH,XD,XDH,BETA,BETAH
                    PRINT #2:T,ERRX,SP11,-SP11,ERRXD,SP22,-SP22,ERRBETAINV,SP33,-SP33
          END IF
LOOP
CLOSE #1
CLOSE #2
END

SUB PROJECT(TP,TS,XP,XDP,BETAP,XH,XDH,XDDH,HP)
T=0.
X=XP
XD=XDP
BETA=BETAP
H=HP
DO WHILE T<=(TS-.0001)
          XDD=.0034*32.2*XD*XD*EXP(-X/22000.)/(2.*BETA)-32.2
          XD=XD+H*XDD
          X=X+H*XD
          T=T+H
LOOP
XH=X
XDH=XD
XDDH=XDD
END SUB
```

Project states forward

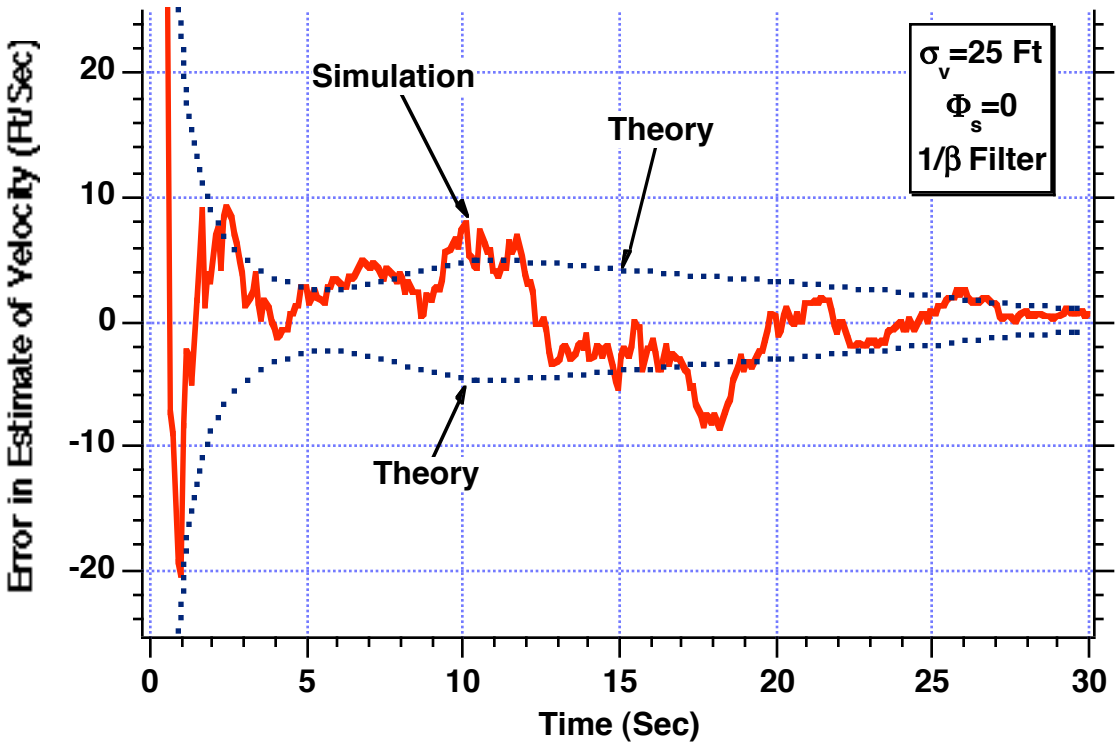
Extended Kalman filter

Integrating nonlinear differential equation forward one sampling Interval with Euler integration

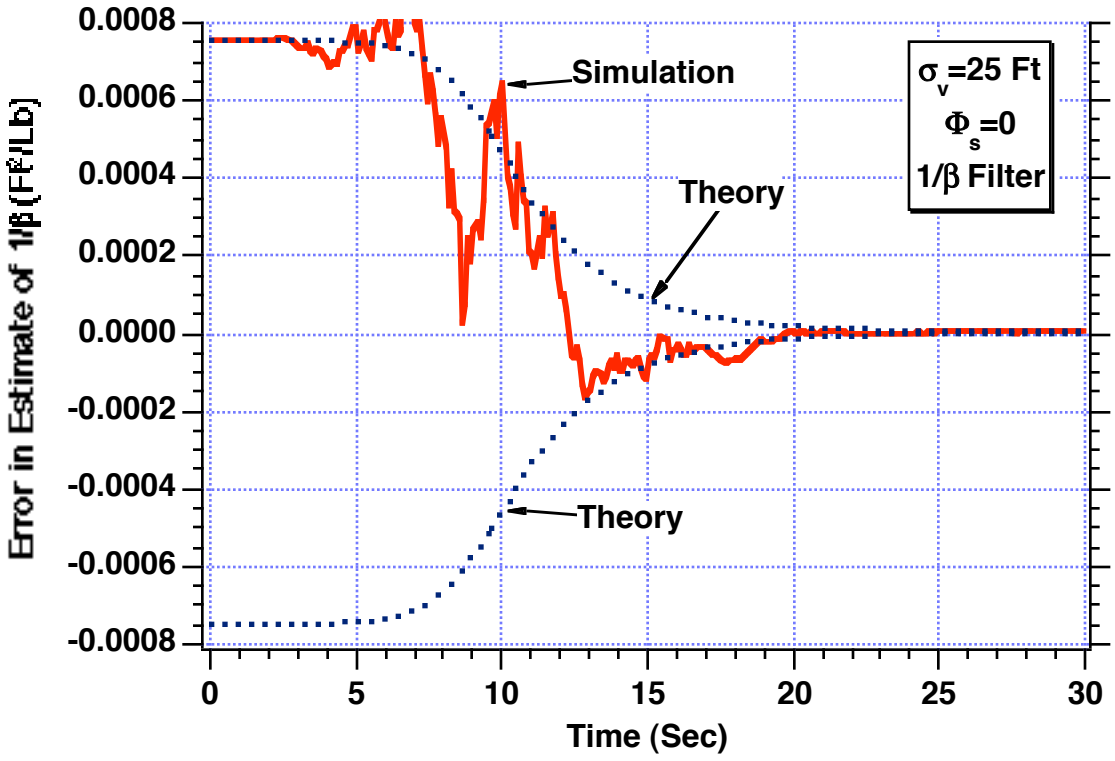
Error in the Estimate of Altitude is Approximately the Same for Both Extended Kalman Filters



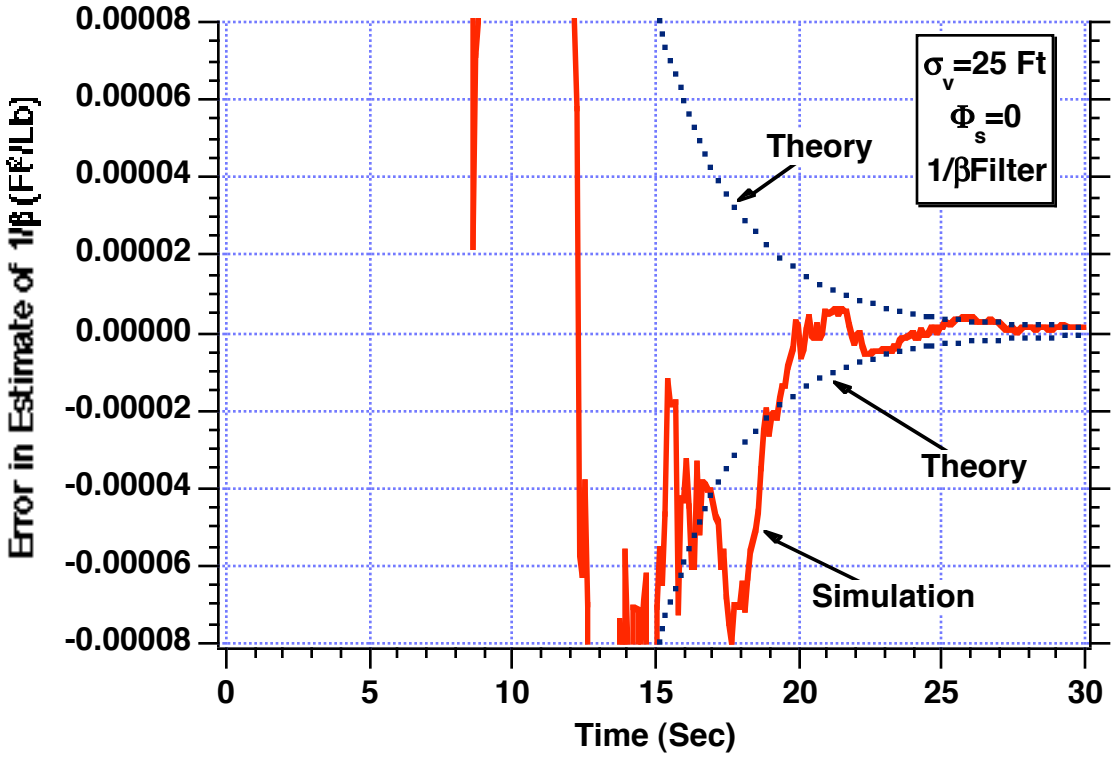
Hangoff Error has Been Eliminated With New Extended Kalman Filter



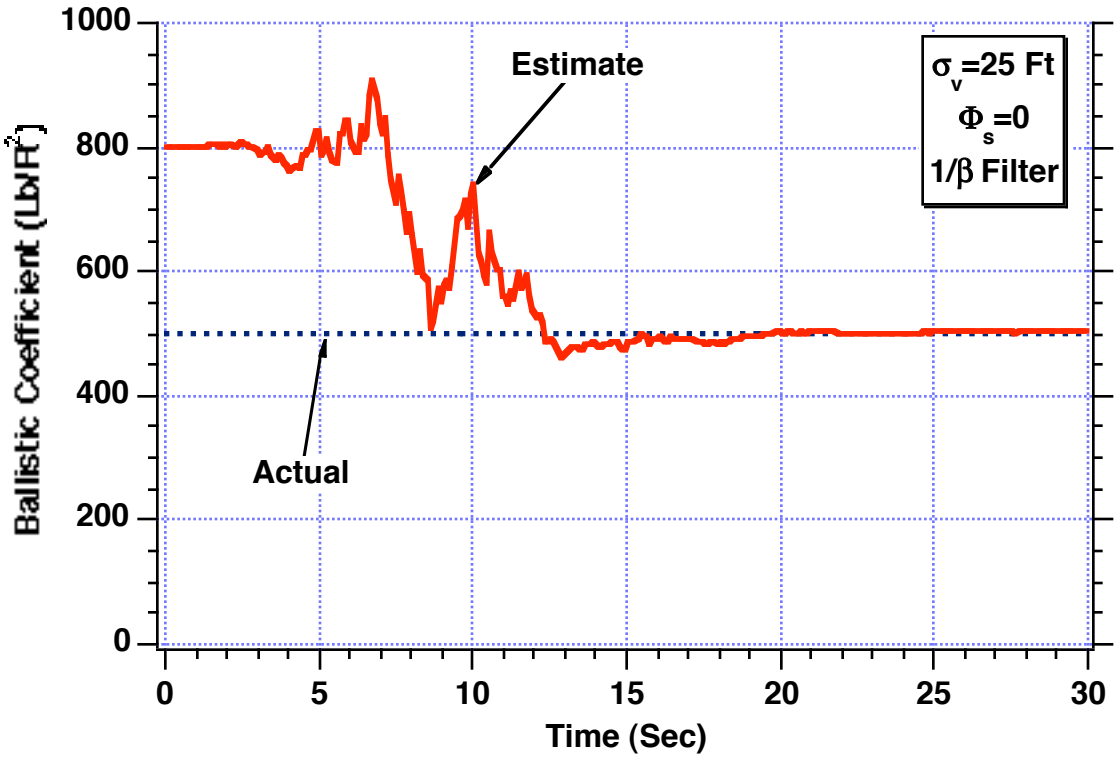
It Still Takes Approximately Ten Seconds to Reduce Error in Estimate of Inverse Ballistic Coefficient



Expanding Scales Shows That Hangoff Error Has Been Eliminated With New Extended Kalman Filter

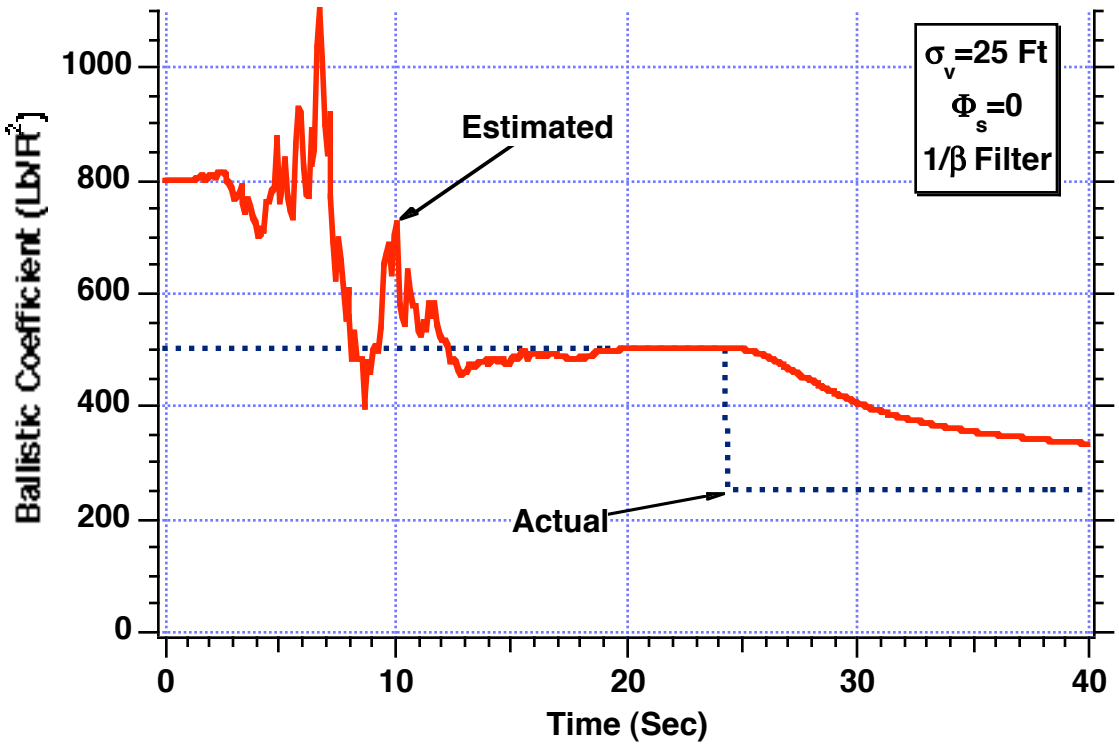


It Takes Ten Seconds to Get Accurate Estimates of Ballistic Coefficient

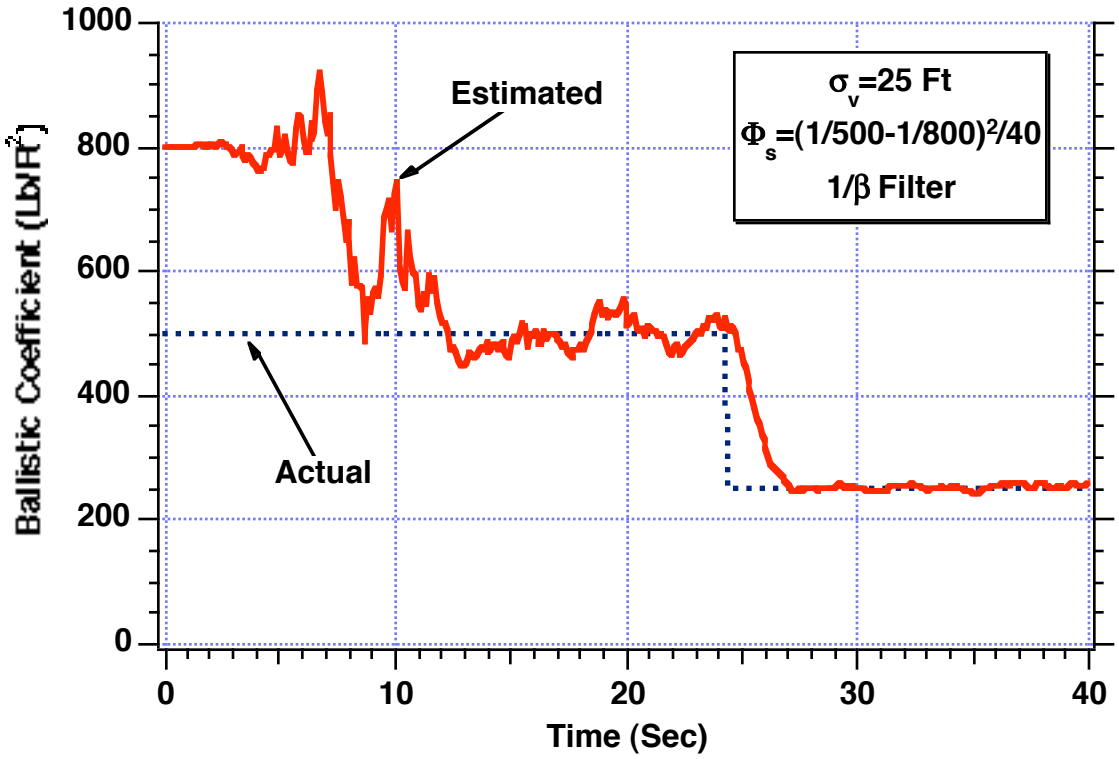


Why Process Noise is Required

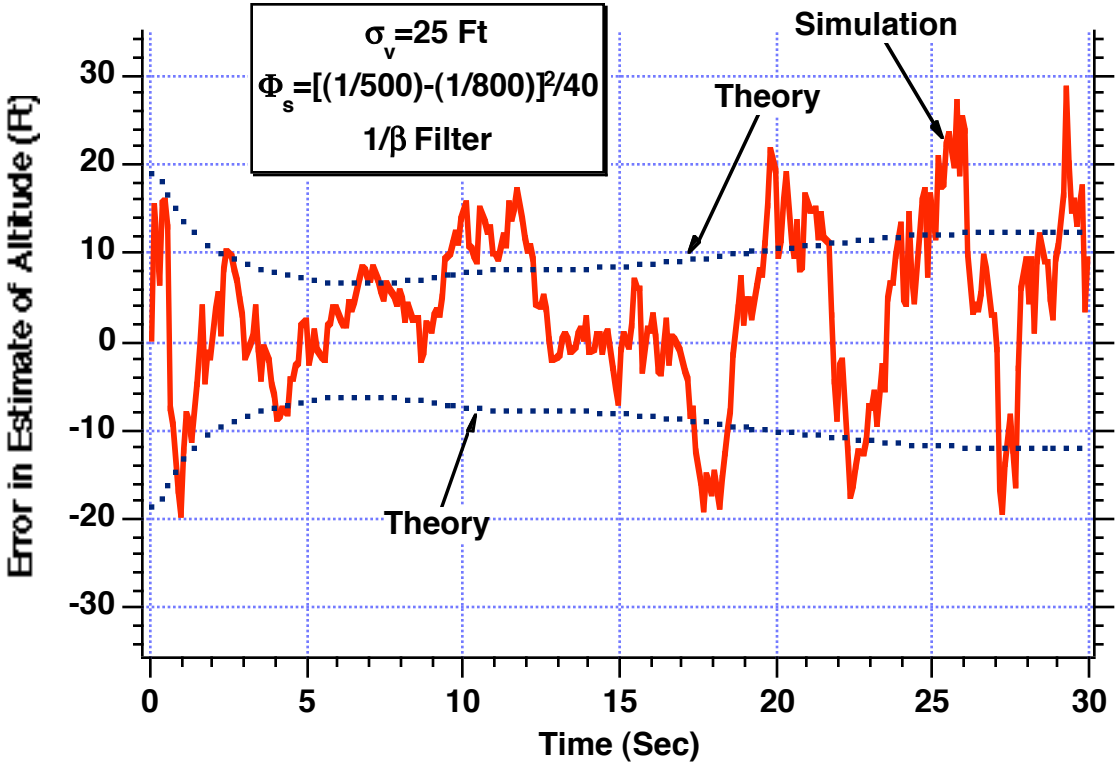
Filter Can Not Track Changes in Ballistic Coefficient if There is no Process Noise



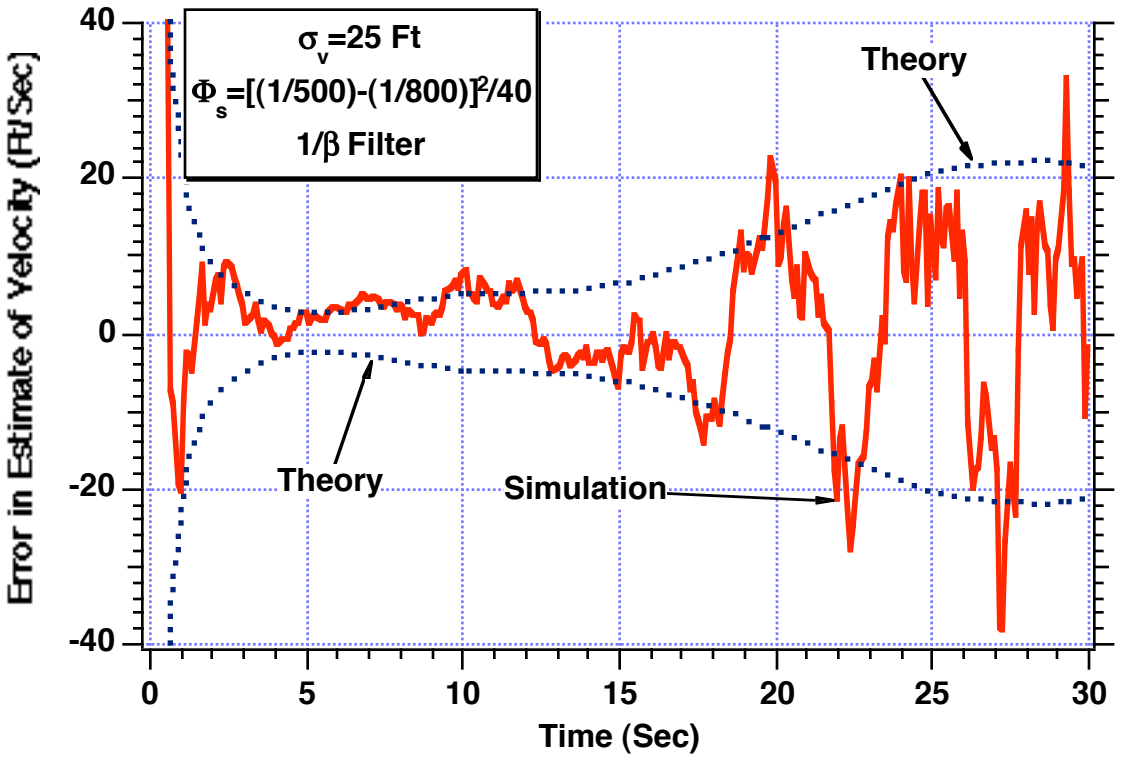
Adding Process Noise Enables Filter to Track Changes in the Ballistic Coefficient



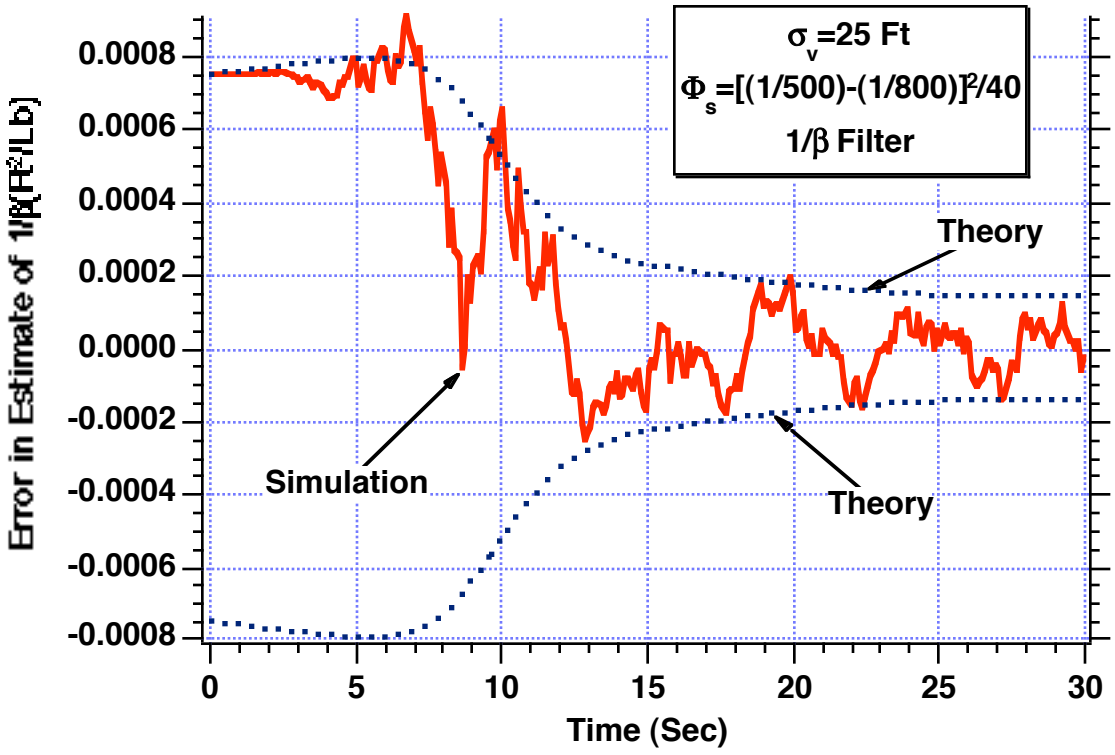
Nominal Errors in Estimate of Altitude Deteriorate Slightly When Process Noise is Added



Nominal Errors in Estimate of Velocity Deteriorate Slightly When Process Noise is Added



Nominal Errors in Estimate of Ballistic Coefficient Deteriorate Slightly When Process Noise is Added



Can We Use Linear Polynomial Kalman Filter Rather Than Extended Kalman Filter?

Reviewing Equations For Second-Order Polynomial Kalman Filter-1

Since acceleration of object varies model of real world is

$$\dddot{x} = u_s$$

Or in state space form

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ u_s \end{bmatrix}$$

Continuous process noise matrix

$$\mathbf{Q} = E\left[\begin{bmatrix} 0 & 0 & u_s \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ u_s \end{bmatrix} \right] = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \Phi_s \end{bmatrix}$$

Systems dynamics matrix

$$\mathbf{F} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Reviewing Equations For Second-Order Polynomial Kalman Filter -2

Measurement equation

$$x^* = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix} + v$$

Measurement matrix

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

Measurement noise matrix is scalar

$$\mathbf{R_k} = \sigma_n^2$$

We have seen previously that fundamental matrix is

$$\mathbf{\Phi_k} = \begin{bmatrix} 1 & T_s & .5T_s^2 \\ 0 & 1 & T_s \\ 0 & 0 & 1 \end{bmatrix}$$

We have already shown discrete process noise matrix to be

$$\mathbf{Q_k} = \Phi_s \begin{bmatrix} \frac{T_s^5}{20} & \frac{T_s^4}{8} & \frac{T_s^3}{6} \\ \frac{T_s^4}{8} & \frac{T_s^3}{3} & \frac{T_s^2}{2} \\ \frac{T_s^3}{6} & \frac{T_s^2}{2} & T_s \end{bmatrix}$$

Solving For Ballistic Coefficient

Recall acceleration of object given by

$$\ddot{x} = \frac{g\rho\dot{x}^2}{2\beta} - g$$

Solving for ballistic coefficient yields

$$\beta = \frac{g\rho\dot{x}^2}{2(\ddot{x} + g)}$$

Estimated ballistic coefficient obtained from filter outputs

$$\hat{\beta} = \frac{g\hat{\rho}\hat{\dot{x}}^2}{2(\hat{\ddot{x}} + g)}$$

Where

$$\hat{\rho} = .0034e^{-\hat{x}/22000}$$

MATLAB Three-State Linear Polynomial Kalman Filter For Tracking Falling Object-1

```
ITERM=1;
G=32.2;
SIGNOISE=25.;
RMAT=SIGNOISE^2;
X=200000.;
XD=-6000.;
BETA=500.;
XH=200025.;
XDH=-6150.;
XDDH=0.;
XNT=322.;
ORDER=3;
TS=.1;
TF=30.;
PHIS=XNT*XNT/TF;
T=0.;
S=0.;
H=.001;
HP=.001;
TS2=TS*TS;
TS3=TS2*TS;
TS4=TS3*TS;
TS5=TS4*TS;
PHI=[1 TS .5*TS*TS;0 1 TS; 0 0 1];
P=[SIGNOISE*SIGNOISE 0 0;0 20000 0;0 0 0];
IDNP=eye(ORDER);
Q=zeros(ORDER,ORDER);
PHIT=PHI';
Q(1,1)=TS5*PHIS/20.;
Q(1,2)=TS4*PHIS/8.;
Q(1,3)=PHIS*TS3/6.;
Q(2,1)=Q(1,2);
Q(2,2)=PHIS*TS3/3.;
Q(2,3)=.5*TS2*PHIS;
Q(3,1)=Q(1,3);
Q(3,2)=Q(2,3);
Q(3,3)=PHIS*TS;
HMAT=[1 0 0];
HT=HMAT';
count=0;
```

Initial actual and estimated states

Initial covariance matrix

Discrete process noise matrix

Measurement matrix and transpose

MATLAB Three-State Linear Polynomial Kalman Filter For Tracking Falling Object-2

```
while T<=TF
        XOLD=X;
        XDOLD=XD;
        XDD=.0034*32.2*XD*XD*exp(-X/22000.)/(2.*BETA)-32.2;
        X=X+H*XD;
        XD=XD+H*XDD;
        T=T+H;
        XDD=.0034*32.2*XD*XD*exp(-X/22000.)/(2.*BETA)-32.2;
        X=.5*(XOLD+X+H*XD);
        XD=.5*(XDOLD+XD+H*XDD);
        S=S+H;
        if S>=(TS-.00001)
                S=0.;
                PHIP=PHI*P;
                PHIPPHIT=PHIP*PHIT;
                M=PHIPPHIT+Q;
                HM=HMAT*M;
                HMHT=HM*HT;
                HMHTR=HMHT+RMAT;
                HMHTRINV=inv(HMHTR);
                MHT=M*HT;
                GAIN=HMHTRINV*MHT;
                KH=GAIN*HMAT;
                IKH=IDNP-KH;
                P=IKH*M;
                XNOISE=SIGNOISE*randn;
                RES=X+XNOISE-XH-TS*XDH-.5*TS*TS*XDDH;
                XH=XH+TS*XDH+.5*TS*TS*XDDH+GAIN(1,1)*RES;
                XDH=XDH+TS*XDDH+GAIN(2,1)*RES;
                XDDH=XDDH+GAIN(3,1)*RES;
                RHOH=.0034*exp(-XH/22000.);
                BETAH=16.1*RHOH*XDH*XDH/(XDDH+32.2);
                ERRX=X-XH;
                SP11=sqrt(P(1,1));
                ERRXD=XD-XDH;
                SP22=sqrt(P(2,2));
                ERRXDD=XDD-XDDH;
                SP33=sqrt(P(3,3));
                SP11P=-SP11;
                SP22P=-SP22;
                SP33P=-SP33;
```

Second-order Runge-Kutta integration on nonlinear differential equation

Riccati equations

Filter

Estimate ballistic coefficient

MATLAB Three-State Linear Polynomial Kalman Filter For Tracking Falling Object-3

```
count=count+1;
ArrayT(count)=T;
ArrayX(count)=X;
ArrayXH(count)=XH;
ArrayXD(count)=XD;
ArrayXDH(count)=XDH;
ArrayXDD(count)=XDD;
ArrayXDDH(count)=XDDH;
ArrayERRX(count)=ERRX;
ArraySP11(count)=SP11;
ArraySP11P(count)=SP11P;
ArrayERRXD(count)=ERRXD;
ArraySP22(count)=SP22;
ArraySP22P(count)=SP22P;
ArrayERRXDD(count)=ERRXDD;
ArraySP33(count)=SP33;
ArraySP33P(count)=SP33P;
end
```

Save as arrays for plotting and writing to files

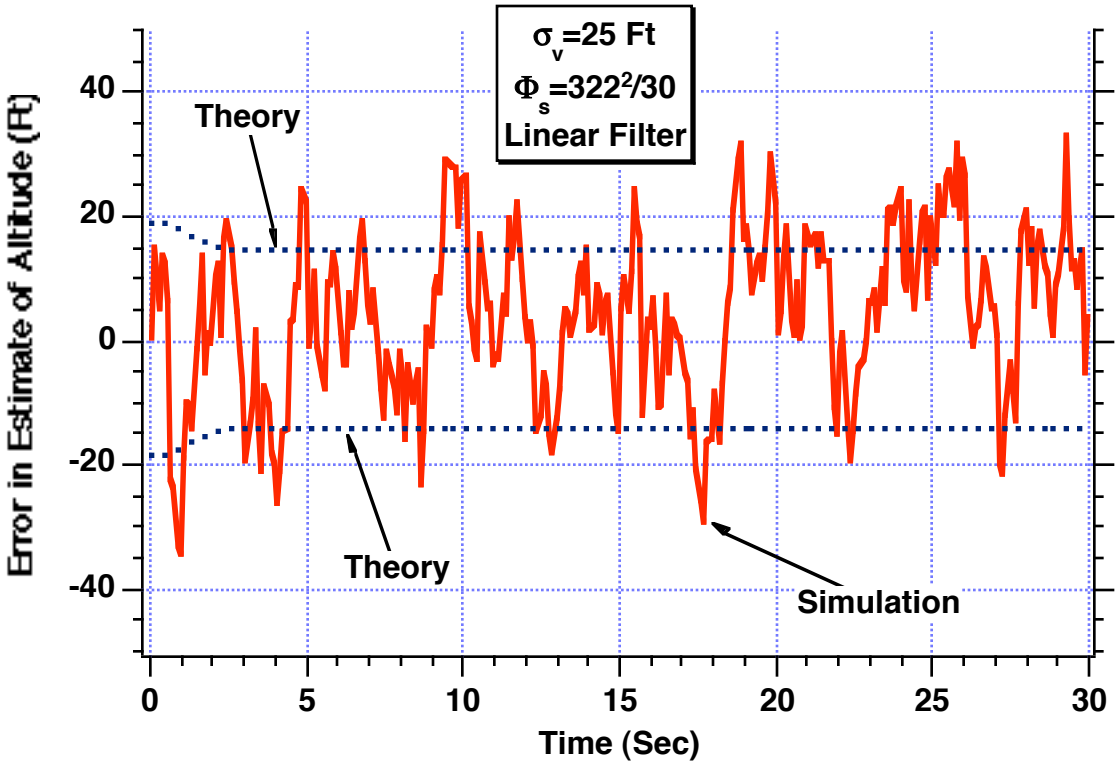
```
end
figure
plot(ArrayT,ArrayERRX,ArrayT,ArraySP11,ArrayT,ArraySP11P),grid
xlabel('Time (Sec)')
ylabel('Error in Estimate of Altitude (Ft)')
axis([0 30 -50 50])
figure
plot(ArrayT,ArrayERRXD,ArrayT,ArraySP22,ArrayT,ArraySP22P),grid
xlabel('Time (Sec)')
ylabel('Error in Estimate of Velocity (Ft/Sec)')
axis([0 30 -150 150])
figure
plot(ArrayT,ArrayERRXDD,ArrayT,ArraySP33,ArrayT,ArraySP33P),grid
xlabel('Time (Sec)')
ylabel('Error in Estimate of Acceleration (Ft/Sec^2)')
axis([0 30 -200 200])
```

Plot some data

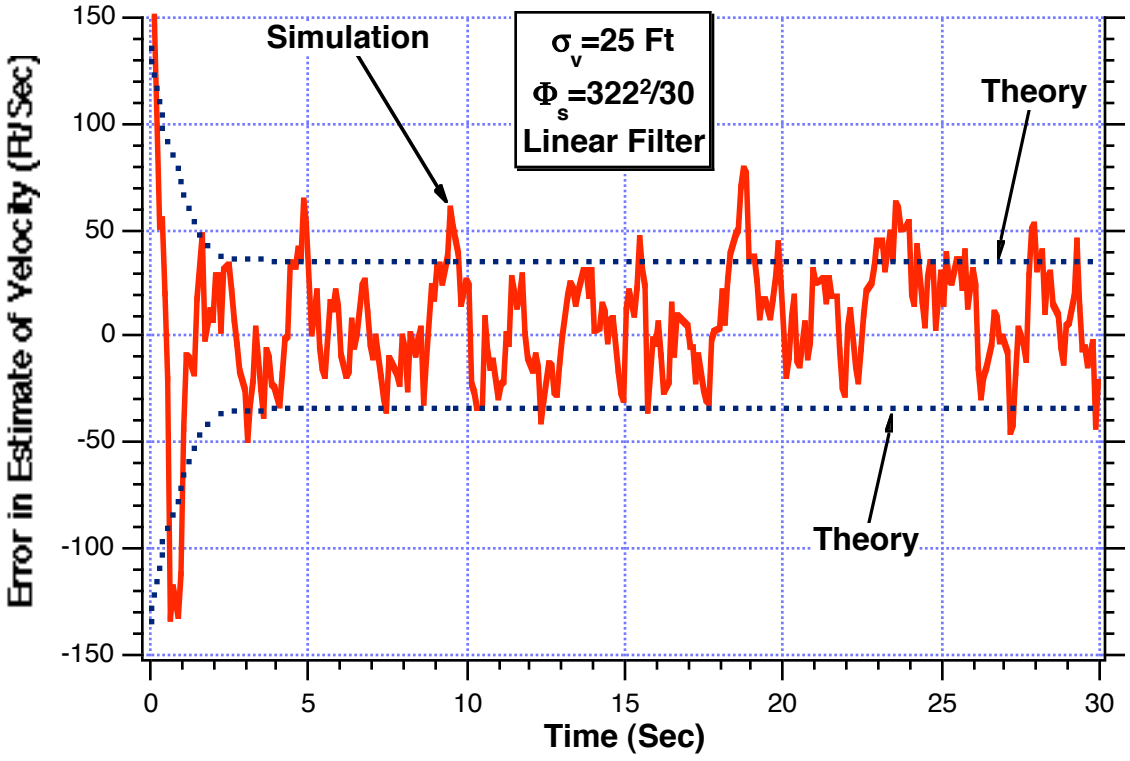
```
clc
output=[ArrayT',ArrayX',ArrayXH',ArrayXD',ArrayXDH',ArrayXDD',ArrayXDDH'];
save datfil output  -ascii
output=[ArrayT',ArrayERRX',ArraySP11',ArraySP11P',ArrayERRXD',ArraySP22',...
ArraySP22P',ArrayERRXDD',ArraySP33',ArraySP33P'];
save covfil output  -ascii
disp 'simulation finished'
```

Write data to files

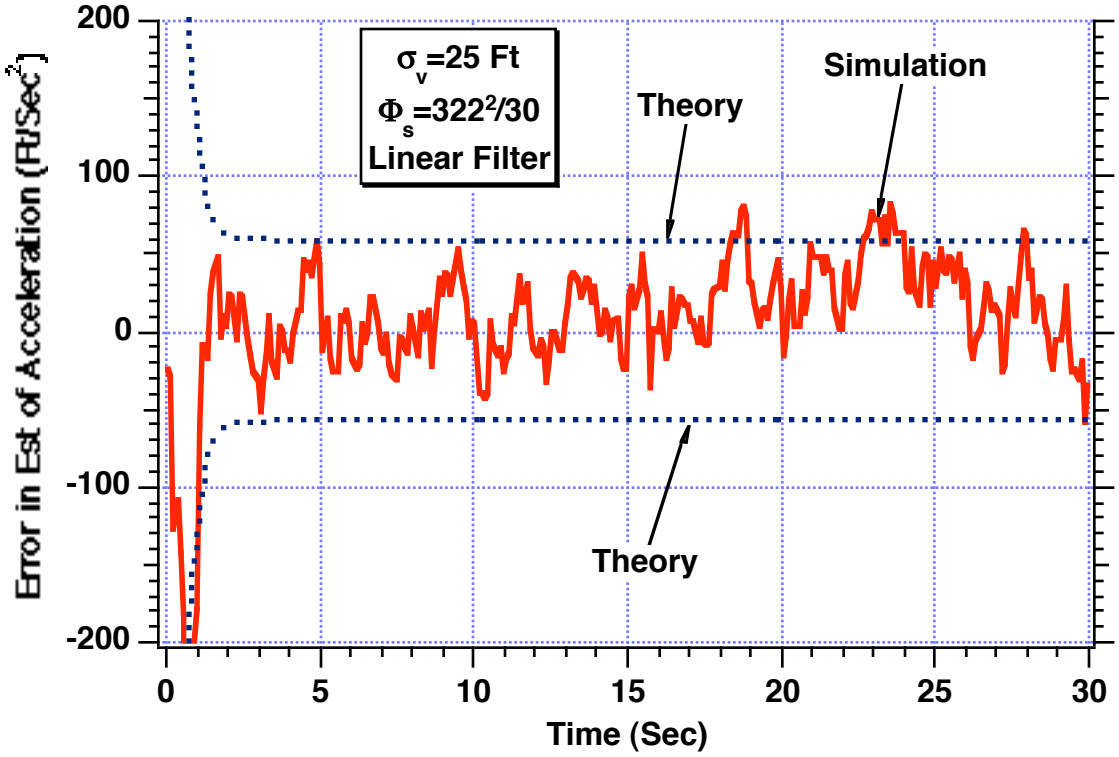
Altitude Estimates are Slightly Worse With Linear Filter



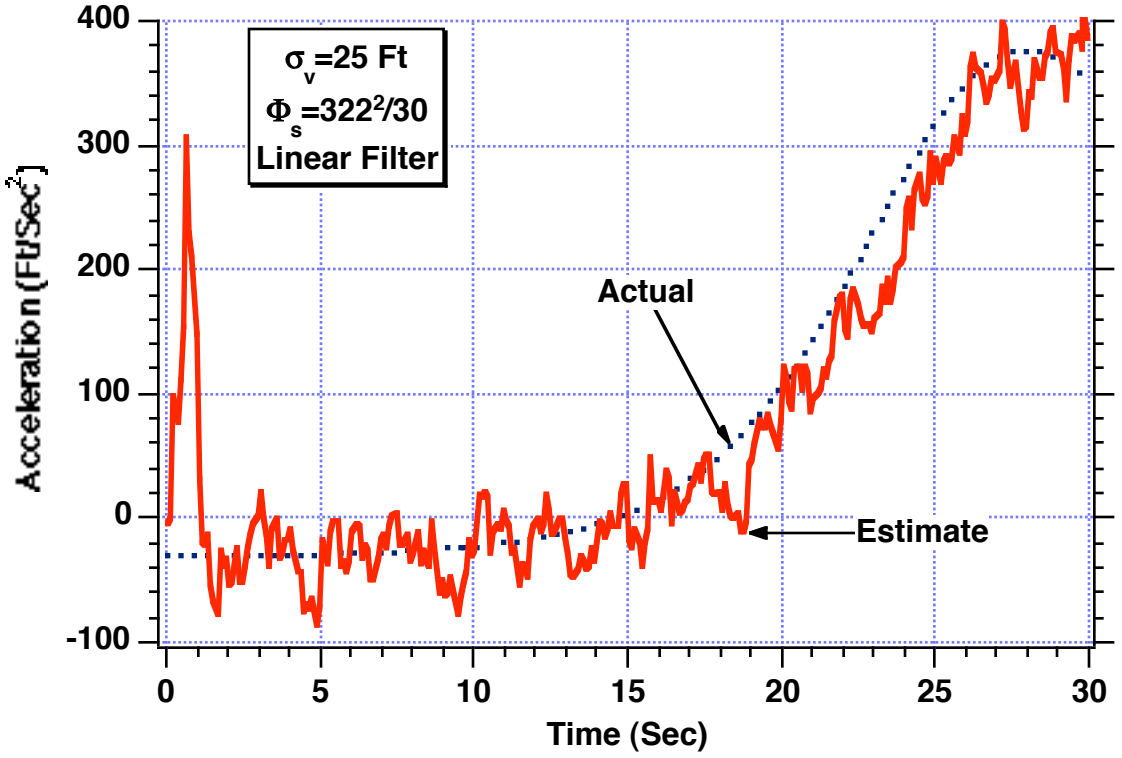
Velocity Estimates are Slightly Worse With Linear Filter



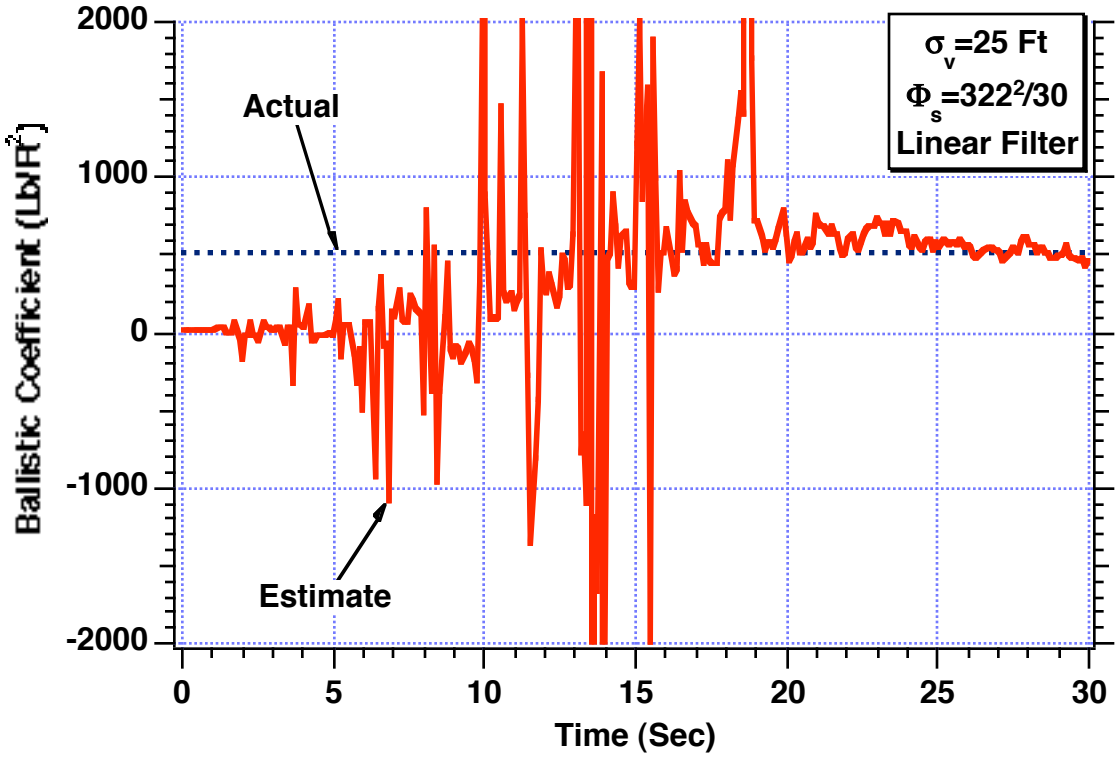
Second-Order Linear Polynomial Kalman Filter is Able to Track Object's Acceleration



Second-Order Linear Polynomial Kalman Filter is Able to Track Object's Acceleration Without Too Much Lag or Noise Propagation

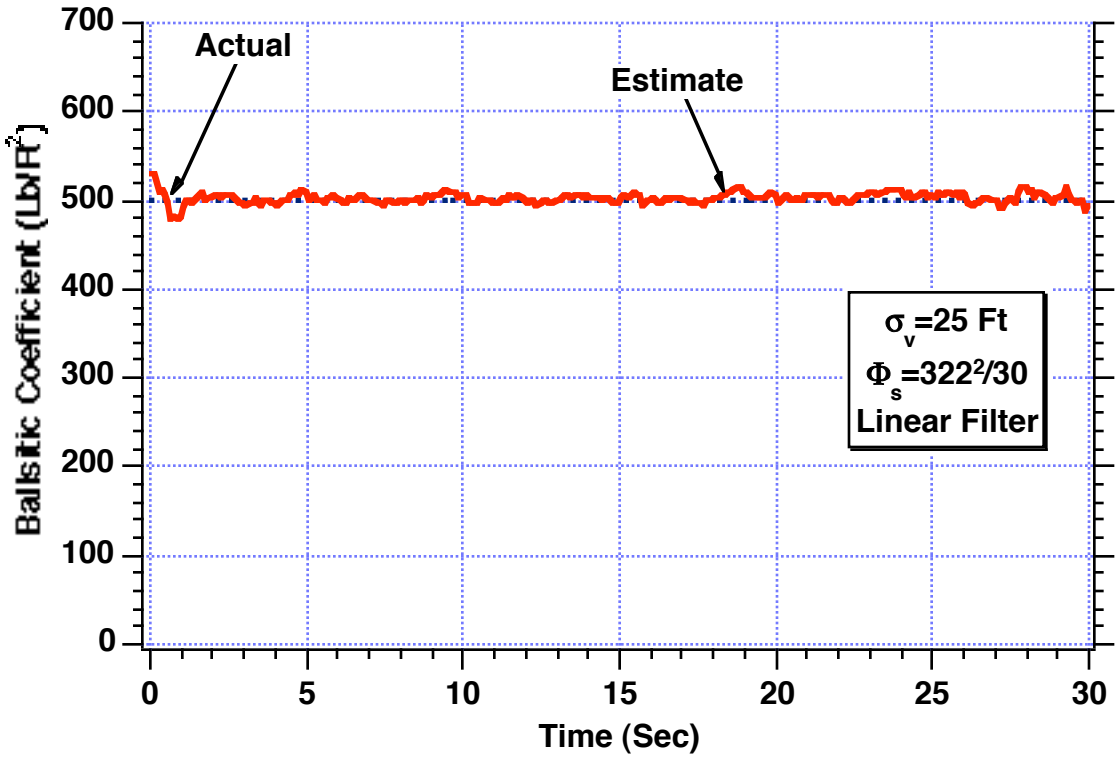


It Takes Approximately 20 Sec for the Second-Order Linear Polynomial Kalman Filter to Accurately Estimate the Ballistic Coefficient



$$\hat{\beta} = \frac{g\hat{\rho}\hat{\dot{x}}^2}{2(\hat{\ddot{x}} + g)} \qquad \hat{\rho} = .0034e^{-\hat{x}/22000}$$

If Acceleration Known Exactly Ballistic Coefficient Estimate is Near Perfect



$$\hat{\beta} = \frac{g\hat{\rho}\hat{\dot{x}}^2}{2(\ddot{x} + g)} \qquad \hat{\rho} = .0034e^{-\hat{x}/22000}$$

Drag and Falling Object Summary

- Two different extended Kalman filters investigated
  - Process noise can eliminate hangoff error
  - Process noise can make filter sensitive to changes in ballistic coefficient
  - Choice of states can be important
- Polynomial linear Kalman filter investigated
  - Can estimate accelerate of object
  - Ballistic coefficient estimate is poor