Extended Kalman Filtering

Extended Kalman Filtering
Overview

- **Presentation of theoretical equations**
- **Numerical example involving drag and falling object**
- **Three attempts at designing an extended Kalman filter**
  - **Illustration of divergence problem**
  - **Process noise, accuracy of fundamental matrix and improved integration in state propagation**

Theoretical Equations - 1

Model of the real world is nonlinear

$$\dot{\mathbf{x}} = f(\mathbf{x}) + \mathbf{w}$$

Process noise matrix

$$\mathbf{Q} = E(\mathbf{w}\,\mathbf{w}^{\mathbf{T}})$$

Measurement equation is nonlinear

$$\mathbf{z} = h(\mathbf{x}) + \mathbf{v}$$

Measurement noise matrix

$$\mathbf{R} = E(\mathbf{v}\,\mathbf{v}^{\mathbf{T}})$$

Nonlinear measurement equation for discrete systems

$$\mathbf{z}_{\mathbf{k}} = h(\mathbf{x}_{\mathbf{k}}) + \mathbf{v}_{\mathbf{k}}$$

Systems dynamics and measurement matrices

$$\mathbf{F} = \left.\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}} \qquad \mathbf{H} = \left.\frac{\partial h(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}}$$

Theoretical Equations - 2

Fundamental matrix will only be used by Riccati equations

$$\Phi_k = I + FT_s + \frac{F^2T_s^2}{2!} + \frac{F^3T_s^3}{3!} + ...$$

Often we will use

$$\Phi_k \approx I + FT_s$$

We have already shown that Riccati equations are not too sensitive to approximation errors

Riccati equations for extended Kalman filter are unchanged

$$\mathbf{M_k = \Phi_k P_{k-1} \Phi_k^T + Q_k}$$

$$\mathbf{K_k = M_k H^T (H M_k H^T + R_k)^{-1}}$$

$$\mathbf{P_k = (I - K_k H) M_k}$$

Where discrete process noise matrix found from

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

Theoretical Equations - 3

Kalman filter equation

$$\hat{\mathbf{x}}_k = \bar{\mathbf{x}}_k + \mathbf{K}_k[\mathbf{z}_k - \mathbf{h}(\bar{\mathbf{x}}_k)]$$

Obtained from nonlinear measurement equation

Propagate states forward with numerical integration of nonlinear differential equations

If we use Euler integration for propagation

$$\bar{\mathbf{x}}_k = \hat{\mathbf{x}}_{k-1} + \hat{\dot{\mathbf{x}}}_{k-1}T_s$$

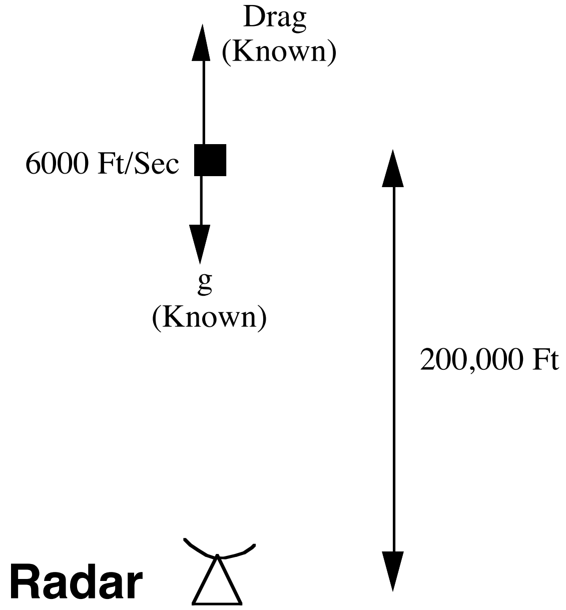
Integration step size

From nonlinear differential equation $\hat{\dot{\mathbf{x}}}_{k-1} = f(\hat{\mathbf{x}}_{k-1})$

Last state estimate

Drag Acting on Falling Object

Radar Tracking Falling Object in Presence of Drag



- Altitude measurements every second
- 1000 ft measurement accuracy

Want to estimate altitude and velocity of falling object

Model of real world

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

Drag and gravity act on object

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

Dynamic pressure

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

Air density

Therefore

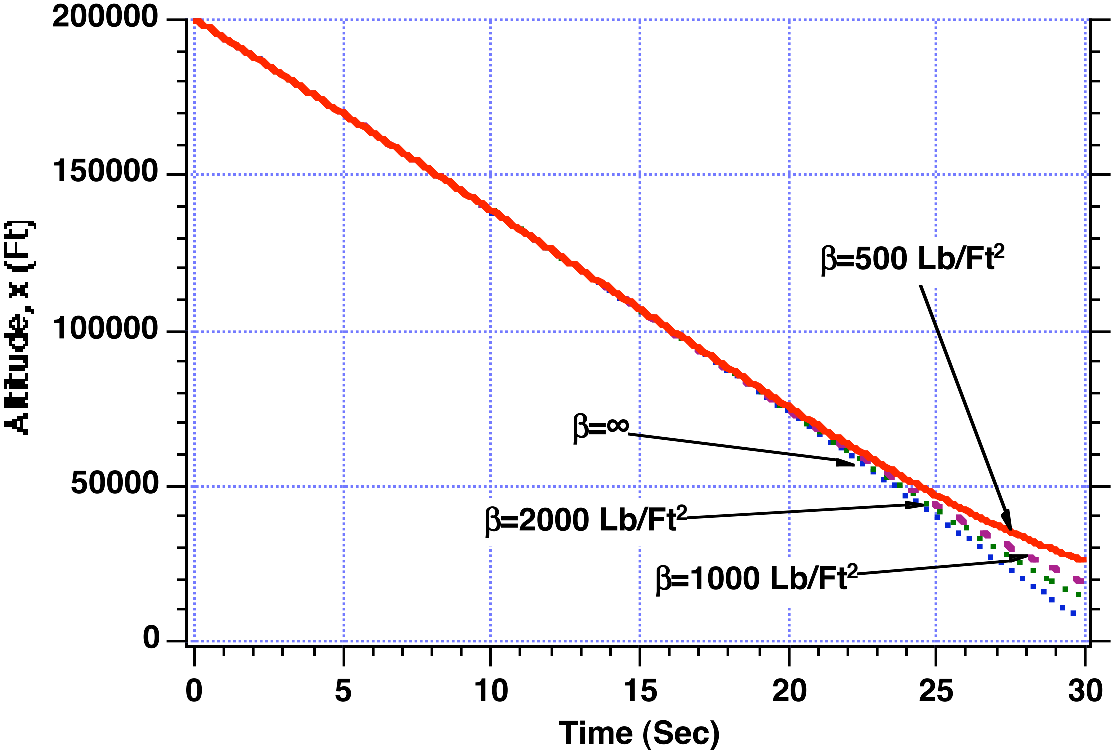
$$\ddot{x} = \frac{Q_p g}{\beta} - g = \frac{.5g\rho\dot{x}^2}{\beta} - g = \frac{.0034ge^{-x/22000}\dot{x}^2}{2\beta} - g \longleftarrow$$

Nonlinear differential equation describing real world

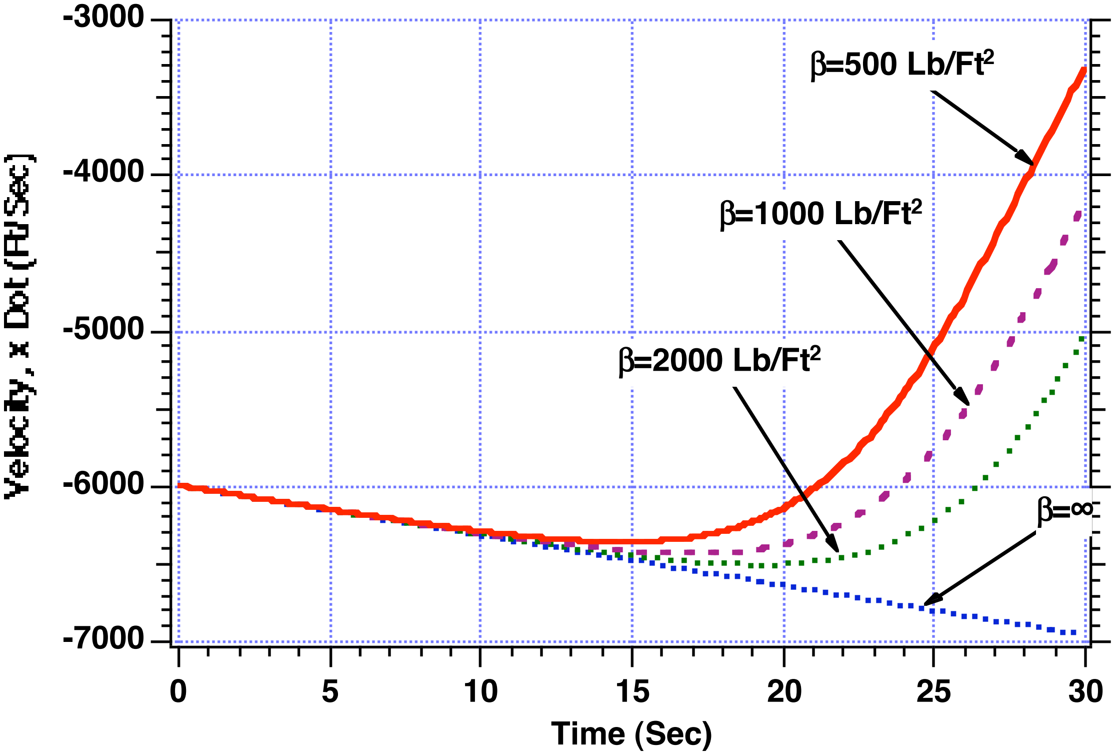
FORTRAN Simulation of Falling Object Under Influence of Drag and Gravity

```
IMPLICIT REAL*8 (A-H)
IMPLICIT REAL*8 (O-Z)
G=32.2
X=200000.
XD=-6000.
BETA=500.
OPEN(1,STATUS='UNKNOWN',FILE='DATFIL')
TS=.1
TF=30.
T=0.
S=0.
H=.001
WHILE(T<=TF)
        XOLD=X
        XDOLD=XD
        XDD=.0034*G*XD*XD*EXP(-X/22000.)/(2.*BETA)-G
        X=X+H*XD
        XD=XD+H*XDD
        T=T+H
        XDD=.0034*G*XD*XD*EXP(-X/22000.)/(2.*BETA)-G
        X=.5*(XOLD+X+H*XD)
        XD=.5*(XDOLD+XD+H*XDD)
        S=S+H
        IF(S>=(TS-.00001))THEN
                S=0.
                WRITE(9,*)T,X,XD,XDD
                WRITE(1,*)T,X,XD,XDD
        ENDIF
END DO
PAUSE
CLOSE(1)
END
```

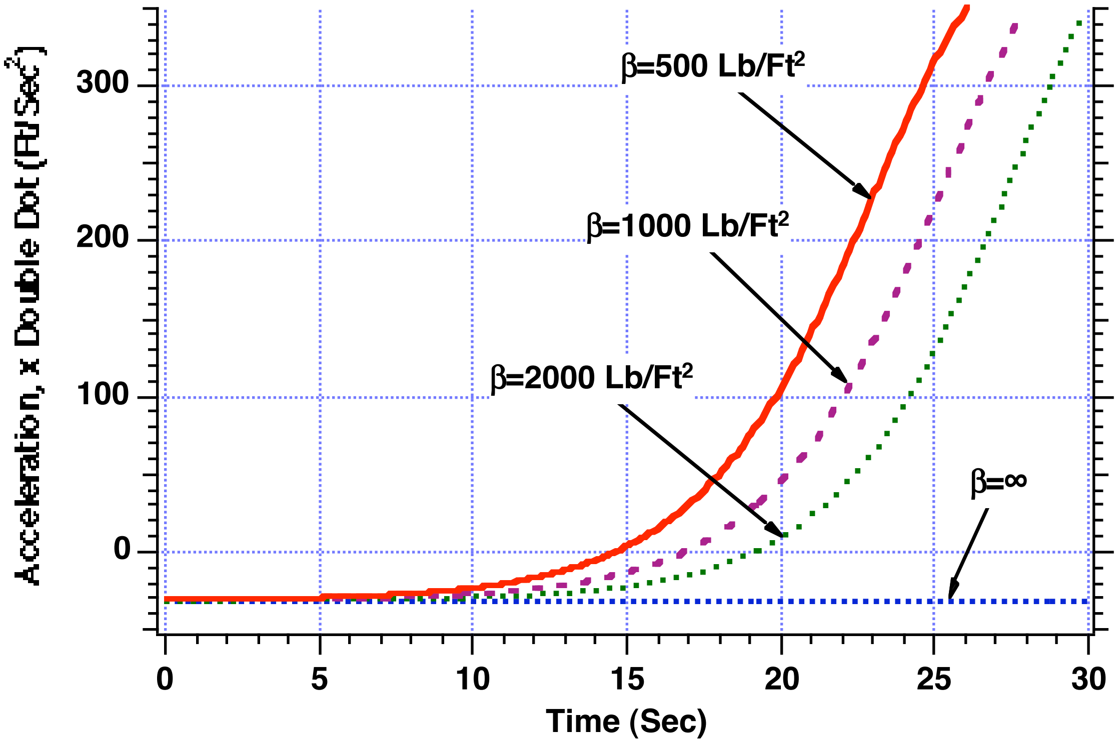
Reducing Drag (Increasing β) Decreases Lowest Altitude That Can be Reached in Thirty Seconds



Increasing Drag (Decreasing β) Reduces Velocity at Lower Altitudes



Drag Causes High Decelerations



First Attempt at Extended Kalman Filter
(Estimate Altitude and Velocity of Object)

Setting Up Extended Kalman Filter

Choose states

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

Model of real world

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

Linearize

Added for protection

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

Systems dynamics matrix

$$\mathbf{F} = \begin{bmatrix} \frac{\partial \dot{x}}{\partial x} & \frac{\partial \dot{x}}{\partial \dot{x}} \\ \frac{\partial \dot{x}}{\partial x} & \frac{\partial \dot{x}}{\partial \dot{x}} \end{bmatrix}_{\mathbf{x}=\hat{\mathbf{x}}}$$

Process noise matrix

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

Find Systems Dynamics Matrix

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

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

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

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

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

Systems dynamics matrix

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

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

Find Fundamental Matrix

Use Taylor series approximation

$$\Phi(t) = \mathbf{I} + \mathbf{F}t + \frac{\mathbf{F}^2t^2}{2!} + \frac{\mathbf{F}^3t^3}{3!} + ...$$

If we define

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

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

Systems dynamics matrix simplifies to

$$\mathbf{F}(t) = \begin{bmatrix} 0 & 1 \\ \frac{-\hat{\rho}g\hat{\dot{x}}^2}{44000\beta} & \frac{\hat{\rho}\hat{\dot{x}}g}{\beta} \end{bmatrix} \longrightarrow \mathbf{F}(t) = \begin{bmatrix} 0 & 1 \\ f_{21} & f_{22} \end{bmatrix}$$

Two term Taylor series approximation

$$\Phi(t) = \mathbf{I} + \mathbf{F}t = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ f_{21} & f_{22} \end{bmatrix} t = \begin{bmatrix} 1 & t \\ f_{21}t & 1+f_{22}t \end{bmatrix} \longrightarrow \Phi_{\mathbf{k}} = \begin{bmatrix} 1 & T_s \\ f_{21}T_s & 1+f_{22}T_s \end{bmatrix}$$

More Kalman Filtering Equations

Measurement equation is linear in this example

$$x_k^* = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ \dot{x}_k \end{bmatrix} + v_k \longrightarrow \mathbf{H} = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

Measurement noise matrix

$$\mathbf{R_k} = E(\mathbf{v_k v_k^T}) \longrightarrow \mathbf{R_k} = \sigma_v^2$$

Discrete process noise matrix

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

where

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

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

$$\mathbf{Q_k} = \Phi_s \int_0^{T_s} \begin{bmatrix} \tau^2 & \tau+f_{22}\tau^2 \\ \tau+f_{22}\tau^2 & 1+2f_{22}\tau+f_{22}^2\tau^2 \end{bmatrix} d\tau$$

We now have all matrices required for Riccati equations

$$\mathbf{Q_k} = \Phi_s \begin{bmatrix} \frac{T_s^3}{3} & \frac{T_s^2}{2}+f_{22}\frac{T_s^3}{3} \\ \frac{T_s^2}{2}+f_{22}\frac{T_s^3}{3} & T_s+f_{22}T_s^2+f_{22}^2\frac{T_s^3}{3} \end{bmatrix}$$

Extended Kalman Filter Structure

Nonlinear differential equation

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

Euler integration to propagate states

$$\bar{\dot{x}}_k = \hat{\dot{x}}_{k-1} + T_s \bar{\ddot{x}}_{k-1}$$

$$\bar{x}_k = \hat{x}_{k-1} + T_s \bar{\dot{x}}_{k-1}$$

Fundamental matrix does <u>not</u> propagate states

Kalman filter

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

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

Noisy measurement of altitude

Linearized matrices do <u>not</u> appear in extended Kalman filter

First Attempt at Extended Kalman Filter With MATLAB-1

```
SIGNOISE=1000.;
X=200000.;
XD=-6000.;
BETA=500.;
XH=200025.;
XDH=-6150.;
ORDER=2;
TS=.1;
TF=30.;
PHIS=0./TF;
T=0.;
S=0.;
H=.001;
PHI=zeros(ORDER,ORDER);
P=[SIGNOISE*SIGNOISE 0;0 20000.];
IDNP=eye(ORDER);
Q=zeros(ORDER,ORDER);
HMAT=[1 0];
HT=HMAT';
RMAT=SIGNOISE^2;
count=0;
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.;
                RHOH=.0034*exp(-XH/22000.);
                F21=-32.2*RHOH*XDH*XDH/(44000.*BETA);
                F22=RHOH*32.2*XDH/BETA;
                PHI(1,1)=1.;
                PHI(1,2)=TS;
                PHI(2,1)=F21*TS;
                PHI(2,2)=1.+F22*TS;
                Q(1,1)=PHIS*TS*TS*TS/3.;
                Q(1,2)=PHIS*(TS*TS/2.+F22*TS*TS*TS/3.);
```

Second-order Runge-Kutta integration of actual nonlinear equations

Fundamental matrix

First Attempt at Extended Kalman Filter With MATLAB-2

```
Q(2,1)=Q(1,2);
Q(2,2)=PHIS*(TS+F22*TS*TS+F22*F22*TS*TS*TS/3.);
```

Process noise matrix

```
PHIT=PHI';
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;
```

Riccati equations

```
XNOISE=SIGNOISE*randn;
XDDB=.0034*32.2*XDH*XDH*exp(-XH/22000.)/(2.*BETA)-32.2;
XDB=XDH+XDDB*TS;
```

State propagation by Euler integration

```
XB=XH+TS*XDB;
RES=X+XNOISE-XB;
XH=XB+GAIN(1,1)*RES;
XDH=XDB+GAIN(2,1)*RES;
```

Kalman filter

```
ERRX=X-XH;
SP11=sqrt(P(1,1));
ERRXD=XD-XDH;
SP22=sqrt(P(2,2));
SP11P=-SP11;
SP22P=-SP22;
```

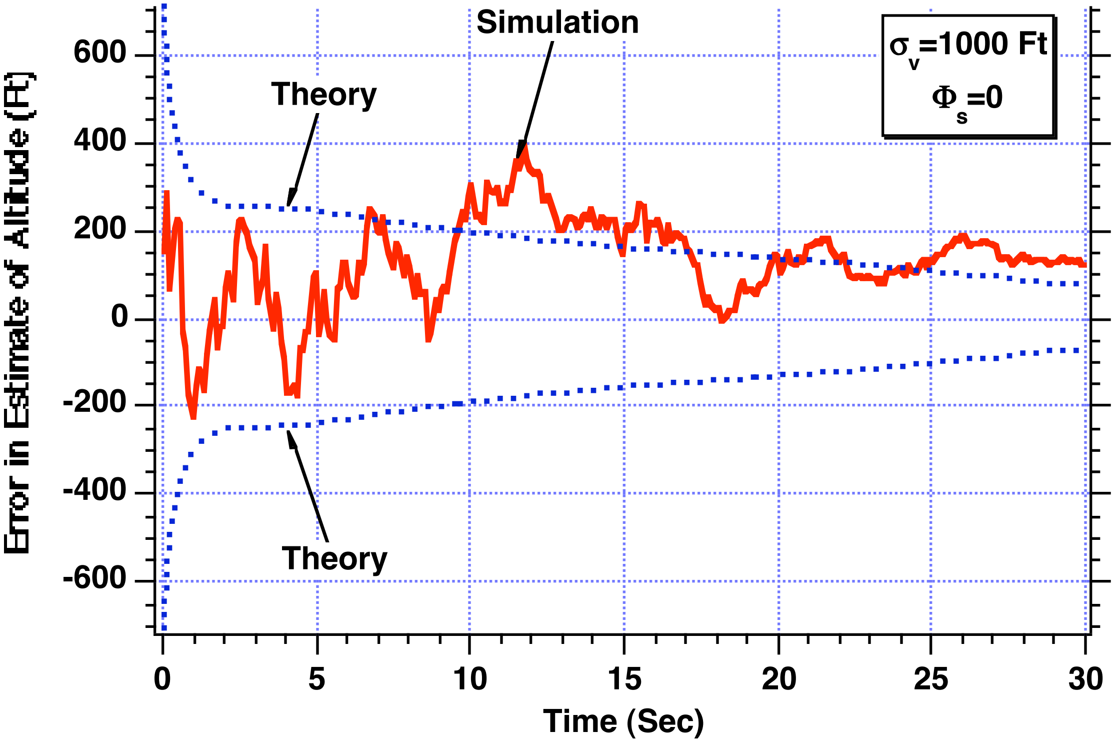
Theoretical and actual errors in estimates

```
count=count+1;
ArrayT(count)=T;
ArrayX(count)=X;
ArrayXH(count)=XH;
ArrayXD(count)=XD;
ArrayXDH(count)=XDH;
ArrayERRX(count)=ERRX;
ArraySP11(count)=SP11;
ArraySP11P(count)=SP11P;
ArrayERRXD(count)=ERRXD;
ArraySP22(count)=SP22;
ArraySP22P(count)=SP22P;
```

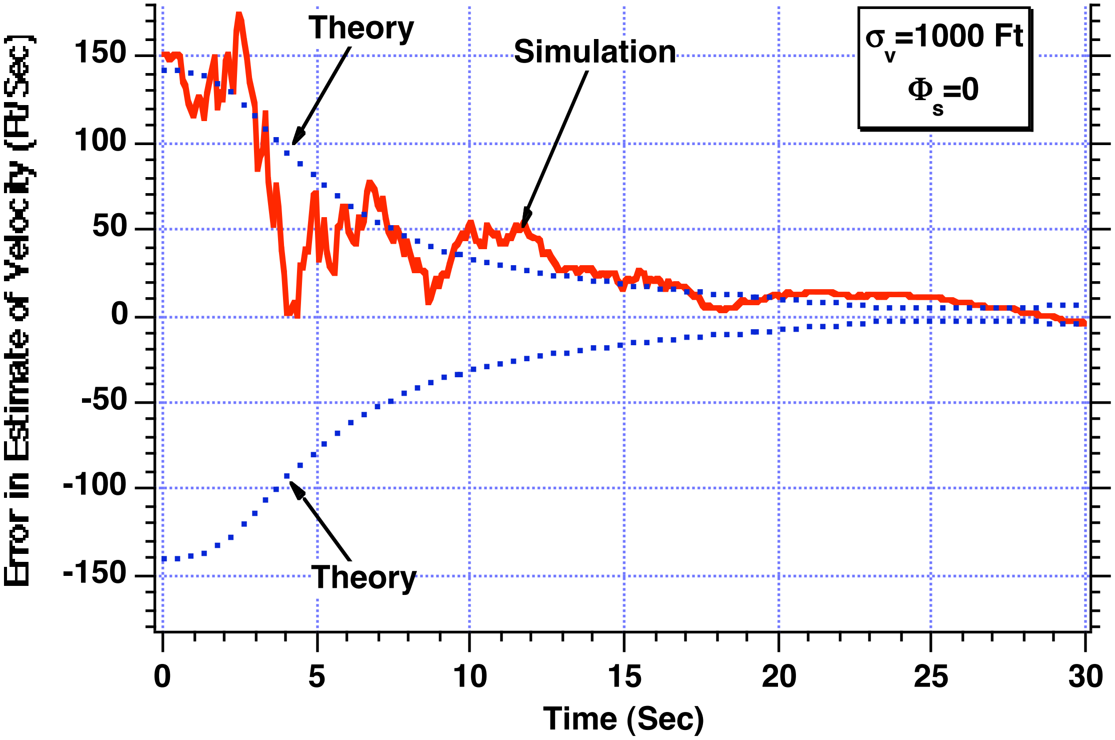
Save as arrays for plotting and writing to files

```
    end
end
```

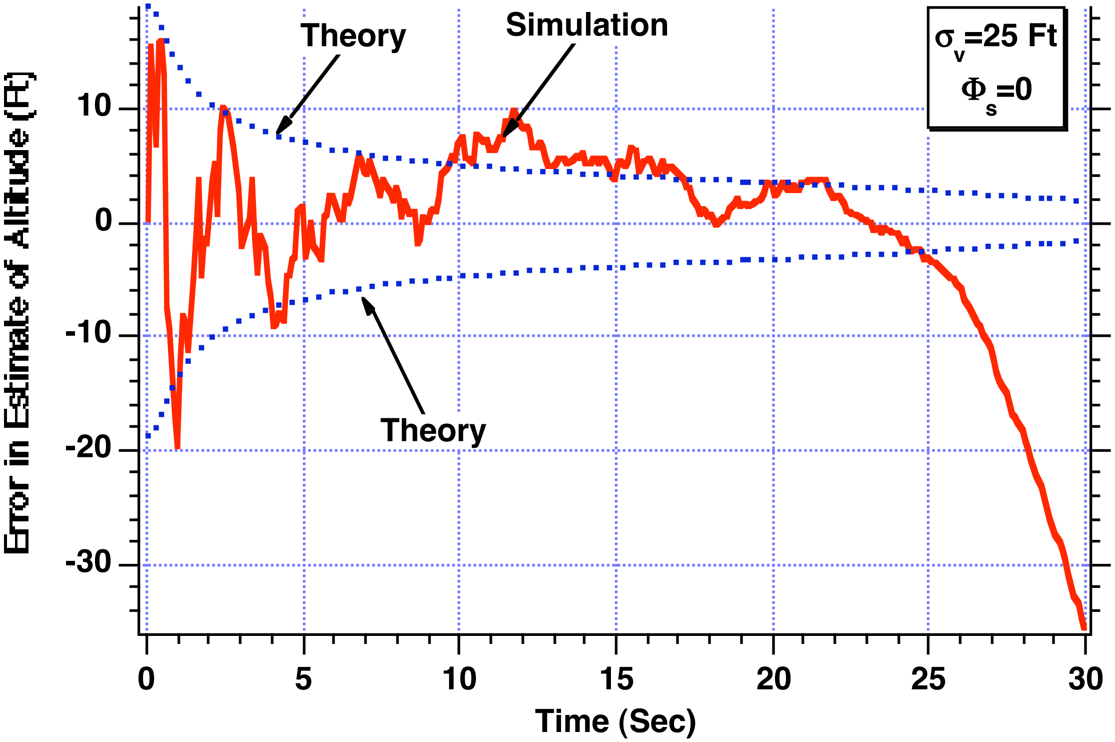
Error in the Estimate of Position Appears to be Within the Theoretical Bounds



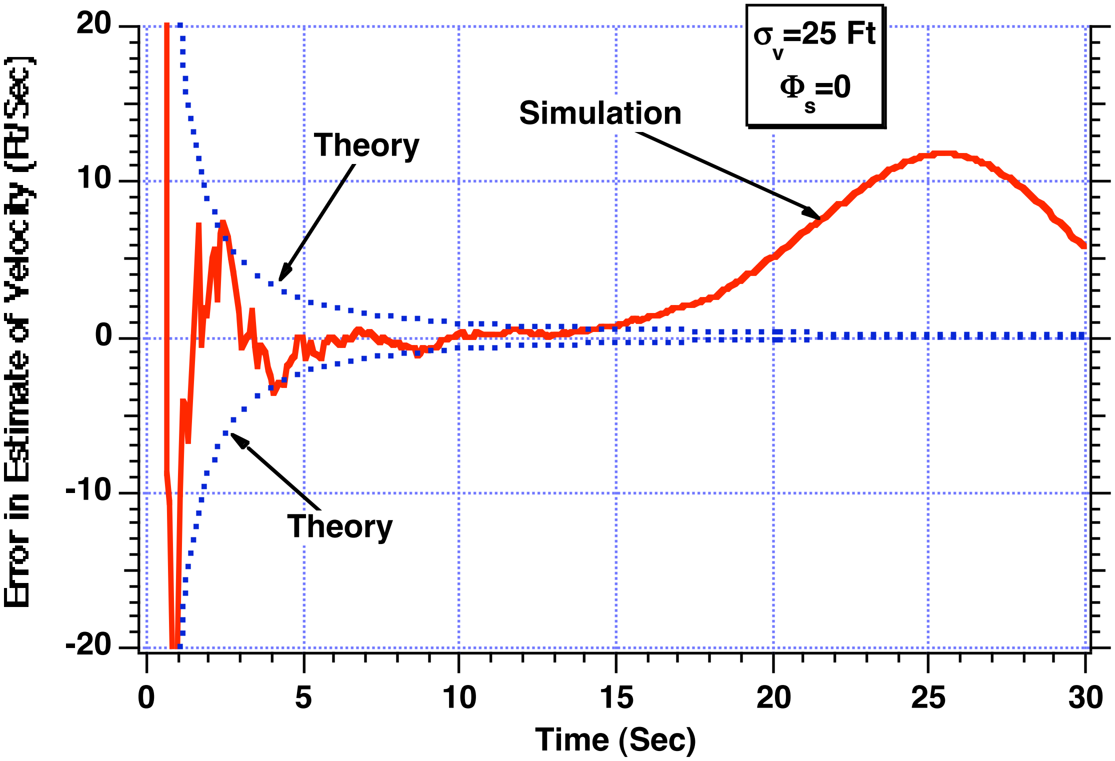
Error in the Estimate of Velocity Appears to be Within the Theoretical Bounds



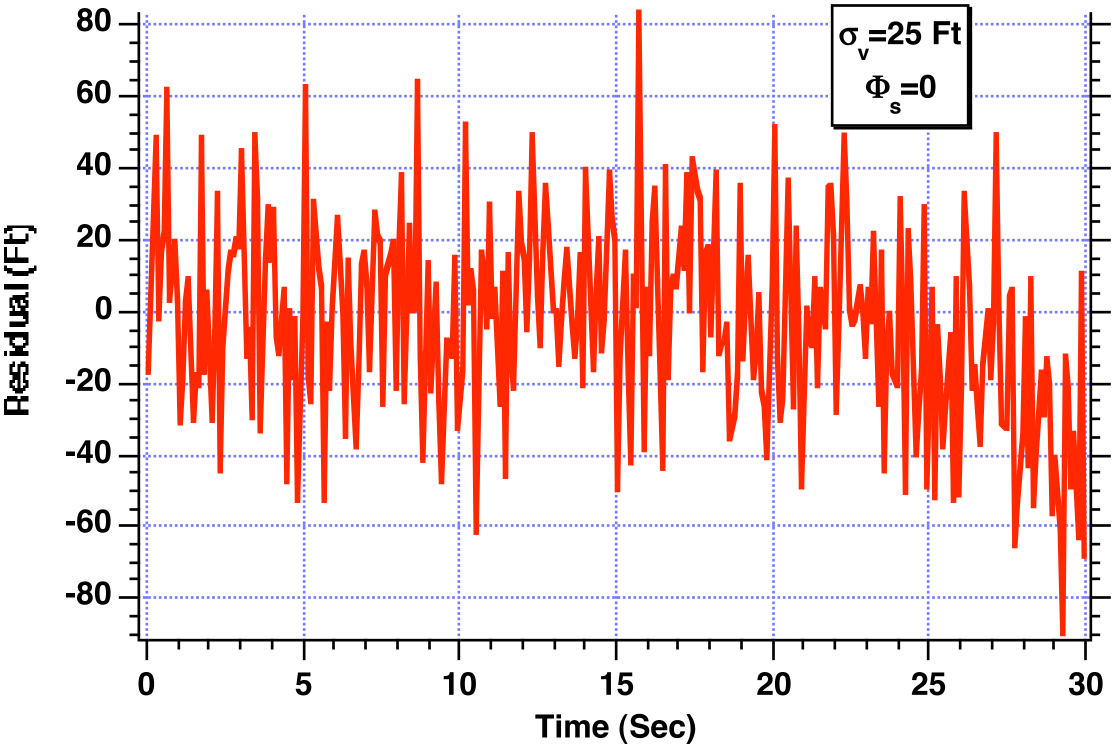
Reducing Measurement Noise Causes Error in Estimate of Position to Diverge



Reducing Measurement Noise Also Causes Error in Estimate of Velocity to Diverge

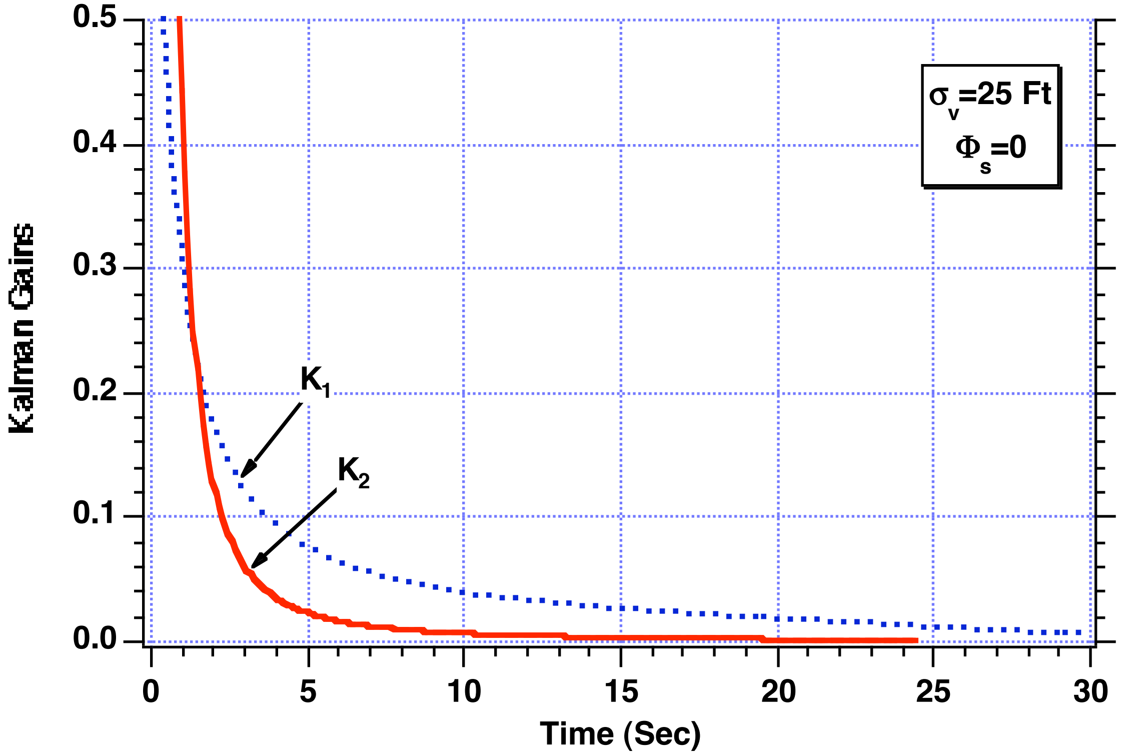


Residual Drifts From Zero After Twenty Seconds



$$\hat{\mathbf{x}}_k = \bar{\mathbf{x}}_k + \mathbf{K}_k \underbrace{[\mathbf{z}_k - \mathbf{h}(\bar{\mathbf{x}}_k)]}_{\text{Residual}}$$

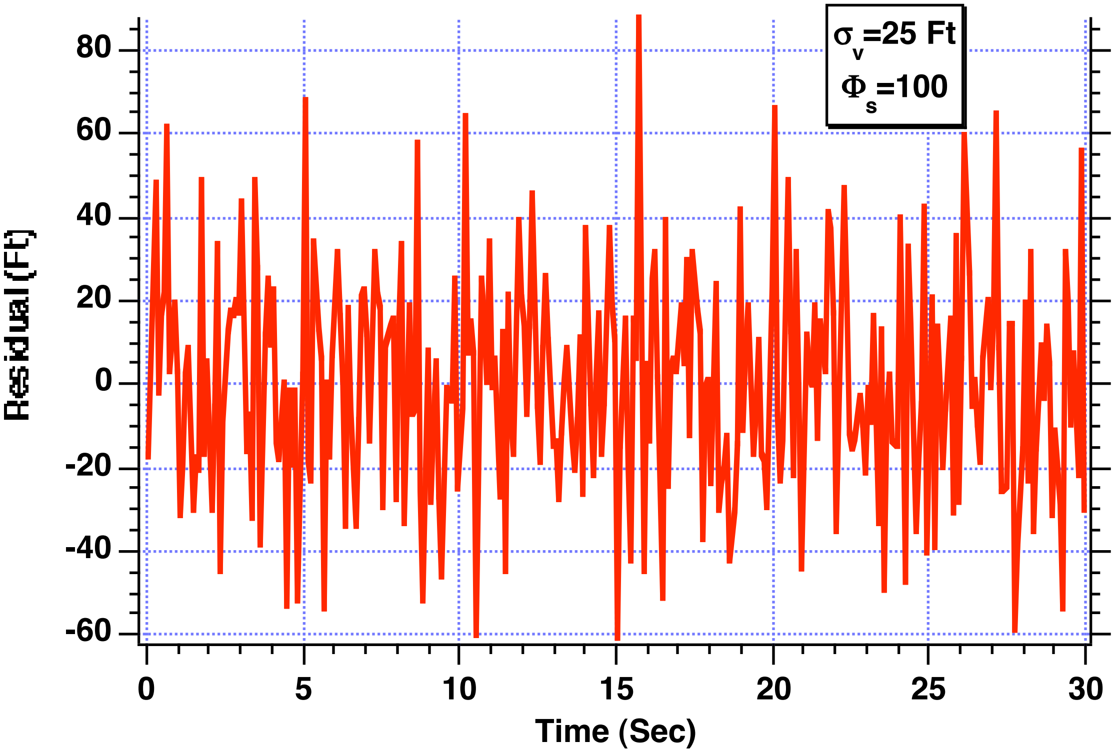
Both Kalman Gains Approach Zero



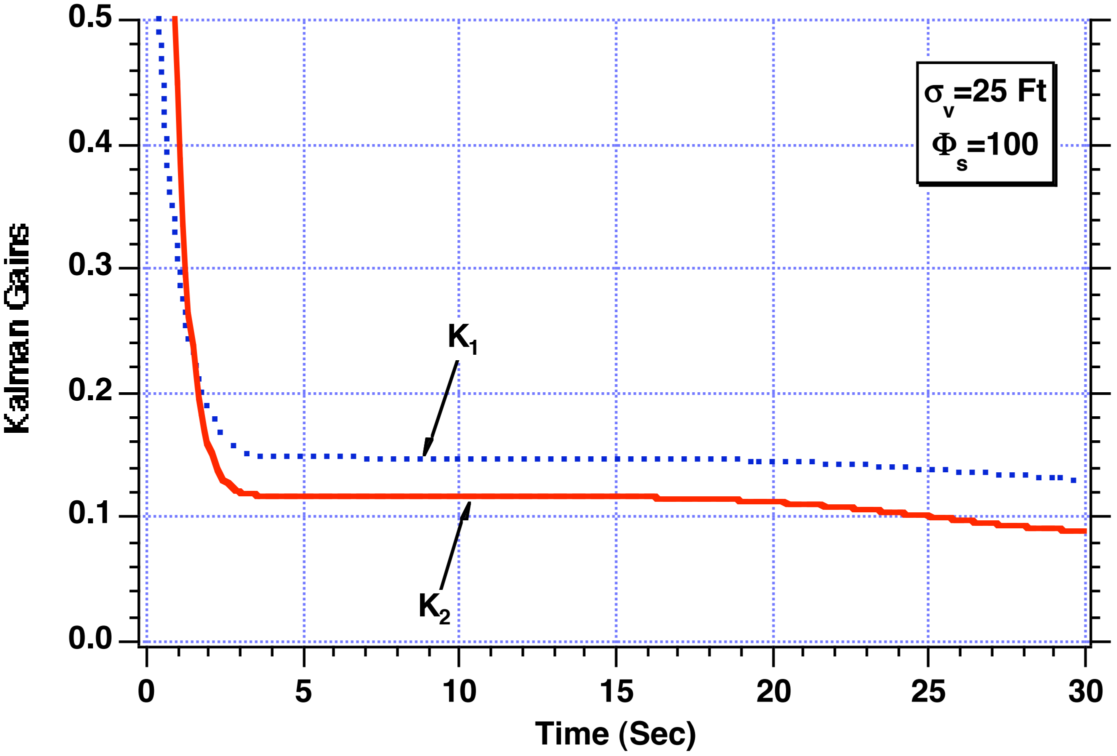
$\hat{\mathbf{x}}_k = \bar{\mathbf{x}}_k + \mathbf{K}_k[\mathbf{z}_k - \mathbf{h}(\bar{\mathbf{x}}_k)]$

When gains are zero measurements are no longer important

Adding Process Noise Prevents Residual From Drifting Away From Zero



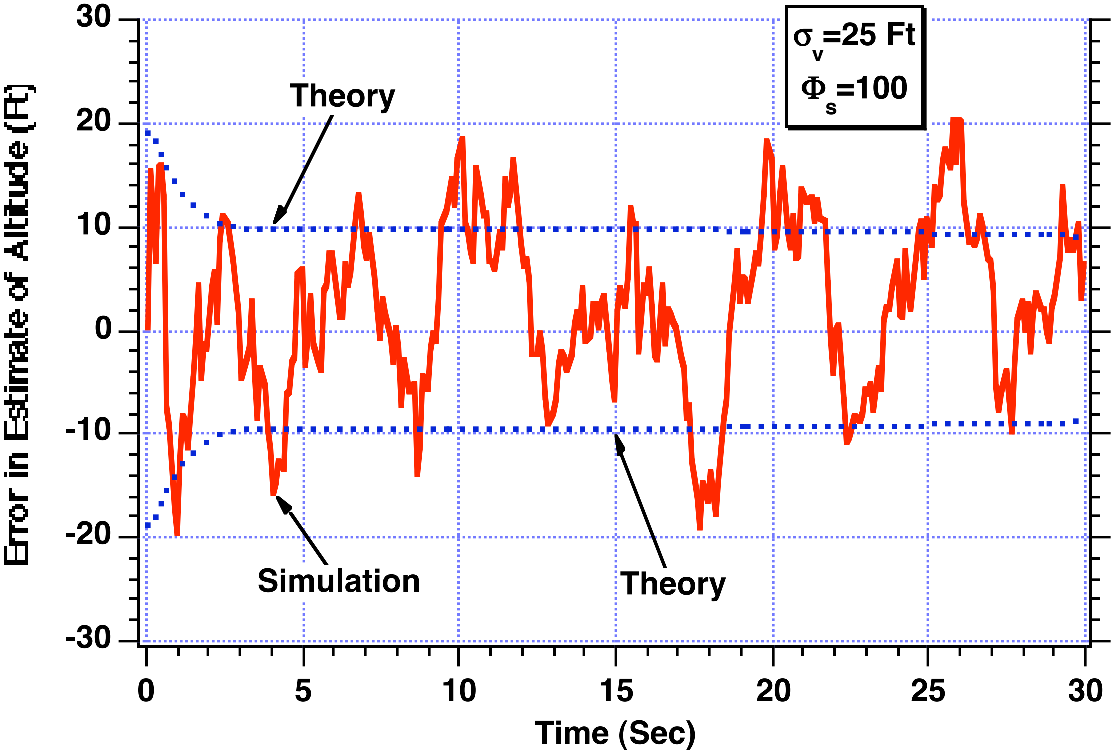
Process Noise Prevents Kalman Gains From Going to Zero



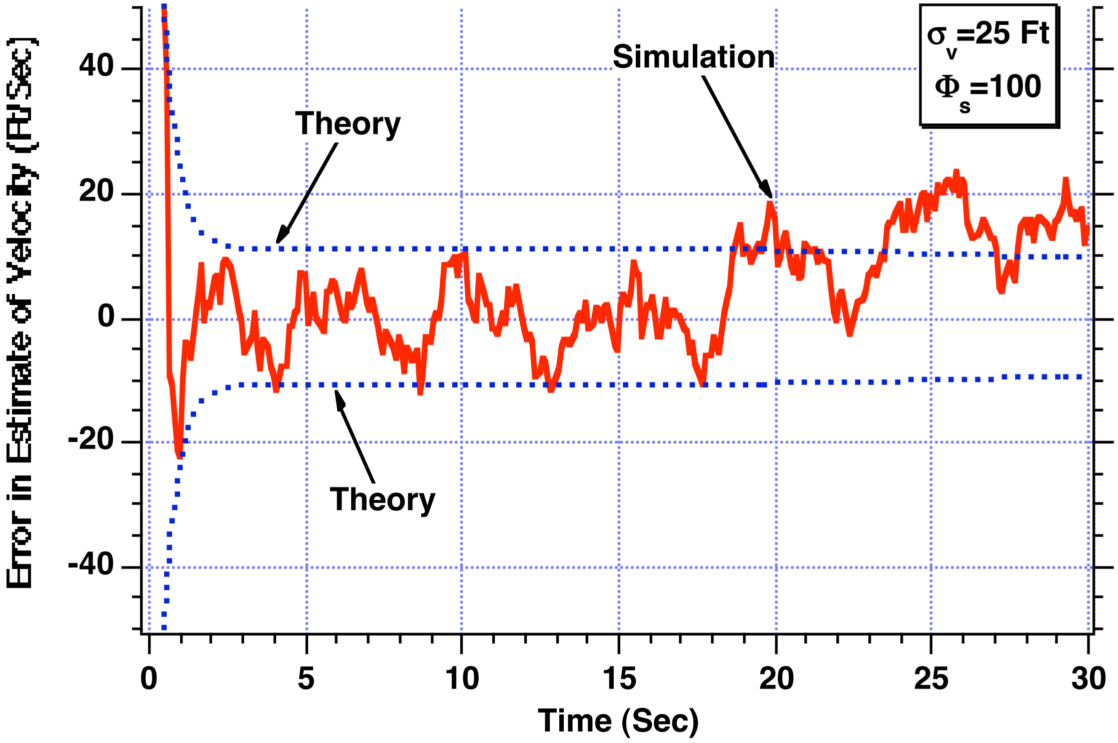
$\hat{\mathbf{x}}_k = \bar{\mathbf{x}}_k + \mathbf{K}_k[\mathbf{z}_k - \mathbf{h}(\bar{\mathbf{x}}_k)]$

When gains are non zero measurements are important

Adding Process Noise Eliminates Filter Divergence in Position



Adding Process Noise Eliminates Filter Divergence in Velocity



Second Attempt at Extended Kalman Filter

Add More Terms to Fundamental Matrix-1

Fundamental matrix can be represented by Taylor series

$$\Phi_k = I + \mathbf{F}T_s + \frac{\mathbf{F}^2 T_s^2}{2!} + \frac{\mathbf{F}^3 T_s^3}{3!} + ...$$

Systems dynamics matrix given by

$$\mathbf{F}(t) = \begin{bmatrix} 0 & 1 \\ f_{21} & f_{22} \end{bmatrix} \quad \textbf{Where} \quad f_{21} = \frac{-\hat{\rho} g \hat{\dot{x}}^2}{44000\beta} \qquad f_{22} = \frac{\hat{\rho}\hat{\dot{x}} g}{\beta}$$

Two term expansion

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

Squaring and cubing

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

$$\mathbf{F}^3 = \begin{bmatrix} 0 & 1 \\ f_{21} & f_{22} \end{bmatrix} \begin{bmatrix} f_{21} & f_{22} \\ f_{22}f_{21} & f_{21}+f_{22}^2 \end{bmatrix} = \begin{bmatrix} f_{22}f_{21} & f_{21}+f_{22}^2 \\ f_{21}^2+f_{22}^2 f_{21} & 2f_{22}f_{21}+f_{22}^3 \end{bmatrix}$$

Add More Terms to Fundamental Matrix-2

Three term expansion

$$\Phi_{k\,3\ \mathrm{Term}} = \begin{bmatrix} 1 & T_s \\ f_{21}T_s & 1+f_{22}T_s \end{bmatrix} + \begin{bmatrix} f_{21} & f_{22} \\ f_{22}f_{21} & (f_{21}+f_{22}^2) \end{bmatrix} \frac{T_s^2}{2}$$

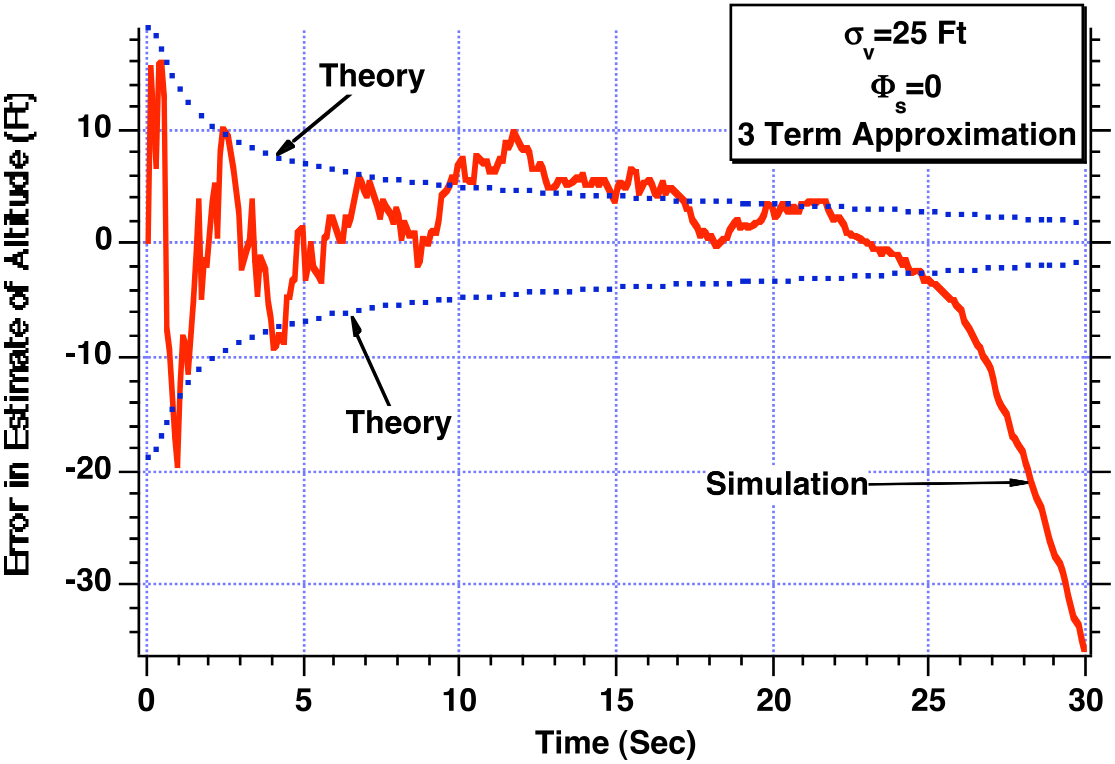
$$\Phi_{k\,3\ \mathrm{Term}} = \begin{bmatrix} 1+f_{21}\frac{T_s^2}{2} & T_s+f_{22}\frac{T_s^2}{2} \\ f_{21}T_s+f_{22}f_{21}\frac{T_s^2}{2} & 1+f_{22}T_s+(f_{21}+f_{22}^2)\frac{T_s^2}{2} \end{bmatrix}$$

Four term expansion

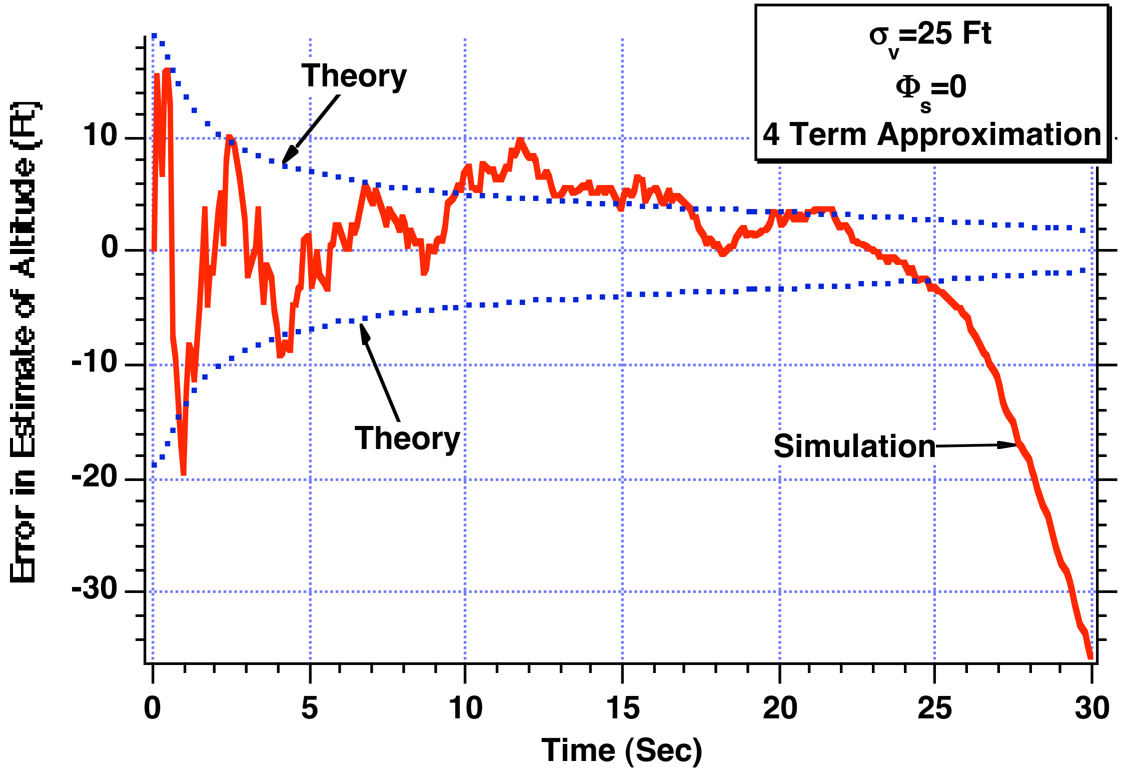
$$\Phi_{k\,4\ \mathrm{Term}} = \begin{bmatrix} 1+f_{21}\frac{T_s^2}{2} & T_s+f_{22}\frac{T_s^2}{2} \\ f_{21}T_s+f_{22}f_{21}\frac{T_s^2}{2} & 1+f_{22}T_s+(f_{21}+f_{22}^2)\frac{T_s^2}{2} \end{bmatrix} + \begin{bmatrix} f_{22}f_{21} & f_{21}+f_{22}^2 \\ f_{21}^2+f_{22}^2f_{21} & 2f_{22}f_{21}+f_{22}^3 \end{bmatrix} \frac{T_s^3}{6}$$

$$\Phi_{k\,4\ \mathrm{Term}} = \begin{bmatrix} 1+f_{21}\frac{T_s^2}{2}+f_{22}f_{21}\frac{T_s^3}{6} & T_s+f_{22}\frac{T_s^2}{2}+(f_{21}+f_{22}^2)\frac{T_s^3}{6} \\ f_{21}T_s+f_{22}f_{21}\frac{T_s^2}{2}+(f_{21}^2+f_{22}^2f_{21})\frac{T_s^3}{6} & 1+f_{22}T_s+(f_{21}+f_{22}^2)\frac{T_s^2}{2}+(2f_{22}f_{21}+f_{22}^3)\frac{T_s^3}{6} \end{bmatrix}$$

Having Three-Term Approximation for Fundamental Matrix Does Not Remove Filter Divergence When There is No Process Noise

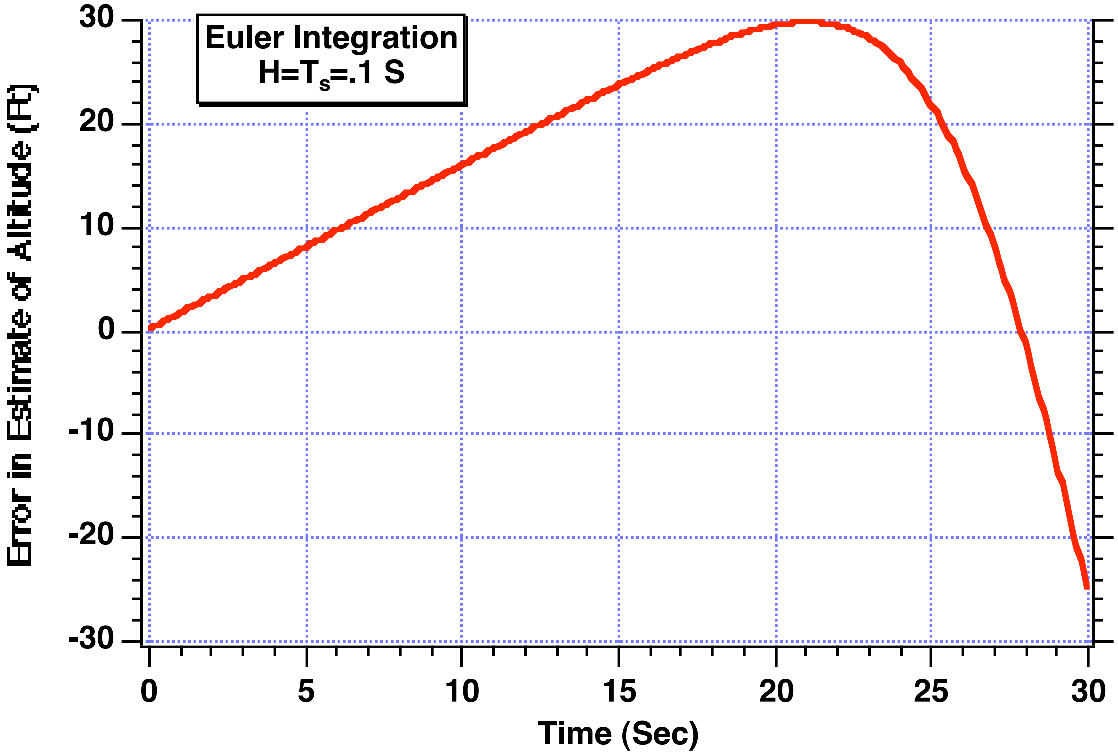


Having Four-Term Approximation for Fundamental Matrix Does Not Remove Filter Divergence When There is No Process Noise



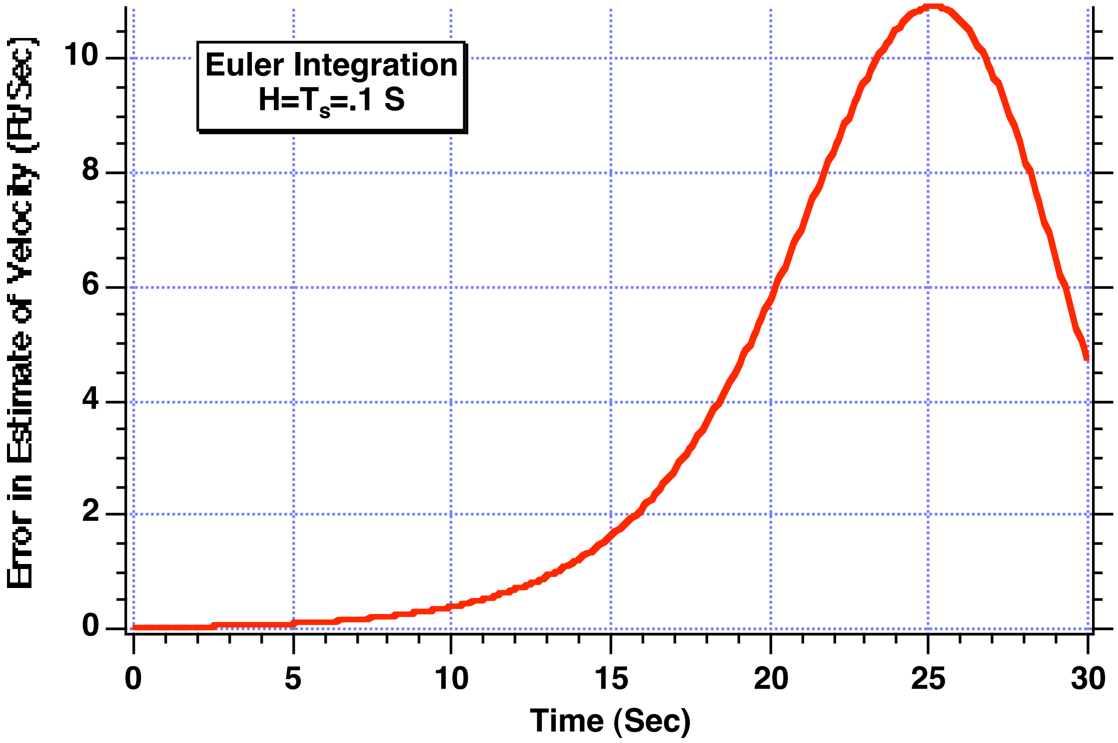
Third Attempt at Extended Kalman Filter

Euler Integration With an Integration Interval of .1 s is Not Adequate For Eliminating Altitude Errors



- Initial state estimates perfect
- Kalman gains set to zero

Euler Integration With an Integration Interval of .1 s is Not Adequate For Eliminating Velocity Errors



- Initial state estimates perfect
- Kalman gains set to zero

True BASIC Simulation to Test State Propagation-1

```
OPTION NOLET
REM UNSAVE "DATFIL"
OPEN #1:NAME "DATFIL",ACCESS OUTPUT,CREATE NEW, ORGANIZATION TEXT
SET #1: MARGIN 1000
X=200000.
XD=-6000.
```

Actual initial conditions

```
BETA=500.
XH=X
XDH=XD
```

Estimated initial conditions are perfect

```
TS=.1
TF=30.
T=0.
S=0.
H=.001
HP=.1
```

Integration intervals for actual and estimated equations

```
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)
```

Second-order Runge-Kutta integration for actual equations

```
        S=S+H
        IF S>=(TS-.00001) THEN
                S=0.
                CALL PROJECT(T,TS,XH,XDH,BETA,XB,XDB,XDDB,HP)
```

Call routine to integrate estimated equations

```
                XH=XB
                XDH=XDB
                ERRX=X-XH
                ERRXD=XD-XDH
                PRINT T,ERRX,ERRXD
                PRINT #1:T,ERRX,ERRXD
        END IF
LOOP
CLOSE #1
END
```

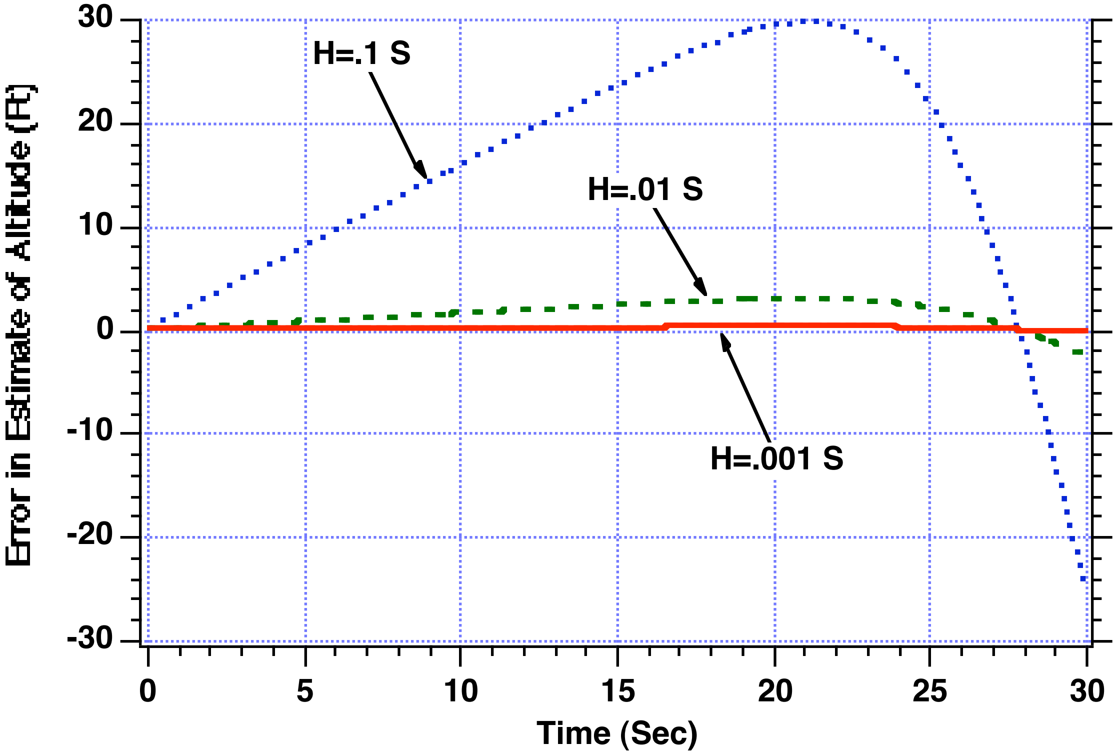
True BASIC Simulation to Test State Propagation-2

```
SUB PROJECT(TP,TS,XP,XDP,BETA,XH,XDH,XDDH,HP)
T=0.
X=XP
XD=XDP
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
```

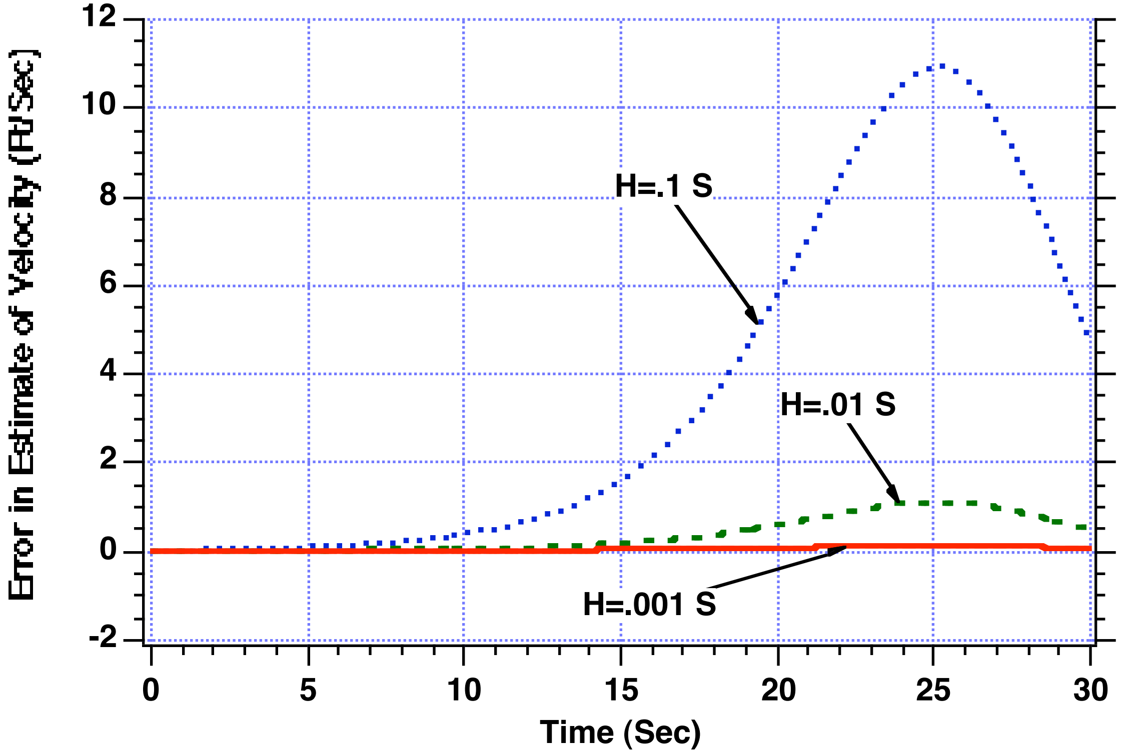
Integration step size

Euler integration for actual equations

Integration Step Size in Propagation Subroutine Must be Reduced to .001 Sec to Keep Errors in Estimate of Altitude Near Zero



Integration Step Size in Propagation Subroutine Must be Reduced to .001 Sec to Keep Errors in Estimate of Velocity Near Zero



True BASIC Version of Extended Kalman Filter with Accurate Propagation Subroutine-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(2,2),P(2,2),M(2,2),PHIP(2,2),PHIPPHIT(2,2),GAIN(2,1)
DIM Q(2,2),HMAT(1,2),HM(1,2),MHT(2,1)
DIM PHIT(2,2)
DIM HMHT(1,1),HT(2,1),KH(2,2),IDNP(2,2),IKH(2,2)
SIGNOISE=25.
X=200000.
XD=-6000.
```

Initial conditions for real world

```
BETA=500.
XH=200025.
XDH=-6150.
```

Initial state estimates

```
ORDER=2
TS=.1
TF=30.
PHIS=0.
T=0.
S=0.
H=.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.
```

Initial covariance matrix

```
MAT HMAT=ZER(1,ORDER)
MAT HT=ZER(ORDER,1)
HMAT(1,1)=1.
HT(1,1)=1.
```

True BASIC Version of Extended Kalman Filter with Accurate Propagation Subroutine-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.*BETA)
                F22=RHOH*32.2*XDH/BETA
                PHI(1,1)=1.
                PHI(1,2)=TS
                PHI(2,1)=F21*TS
                PHI(2,2)=1.+F22*TS
                Q(1,1)=PHIS*TS*TS*TS/3.
                Q(1,2)=PHIS*(TS*TS/2.+F22*TS*TS*TS/3.)
                Q(2,1)=Q(1,2)
                Q(2,2)=PHIS*(TS+F22*TS*TS+F22*F22*TS*TS*TS/3.)
                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
                CALL GAUSS(XNOISE,SIGNOISE)
                CALL PROJECT(T,TS,XH,XDH,BETA,XB,XDB,XDDB)
```

Second-order Runge-Kutta integration for model of real world

Fundamental matrix

Process noise matrix

Riccati equations

Project states ahead

True BASIC Version of Extended Kalman Filter with Accurate Propagation Subroutine-3

```
                RES=X+XNOISE-XB
                XH=XB+GAIN(1,1)*RES
                XDH=XDB+GAIN(2,1)*RES
```

Kalman filter

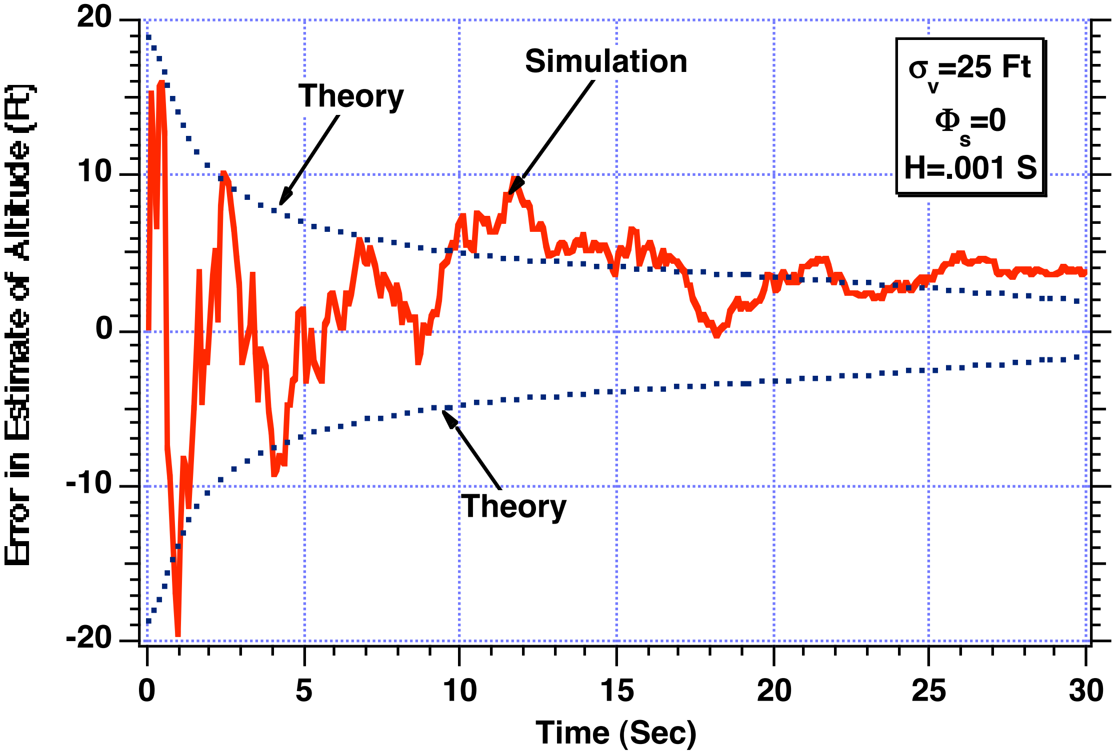
```
                ERRX=X-XH
                SP11=SQR(P(1,1))
                ERRXD=XD-XDH
                SP22=SQR(P(2,2))
                PRINT T,X,XH,XD,XDH
                PRINT #1:T,X,XH,XD,XDH
                PRINT #2:T,ERRX,SP11,-SP11,ERRXD,SP22,-SP22
        END IF
LOOP
CLOSE #1
CLOSE #2
END

SUB PROJECT(TP,TS,XP,XDP,BETA,XH,XDH,XDDH)
T=0.
X=XP
XD=XDP
H=.001
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
```

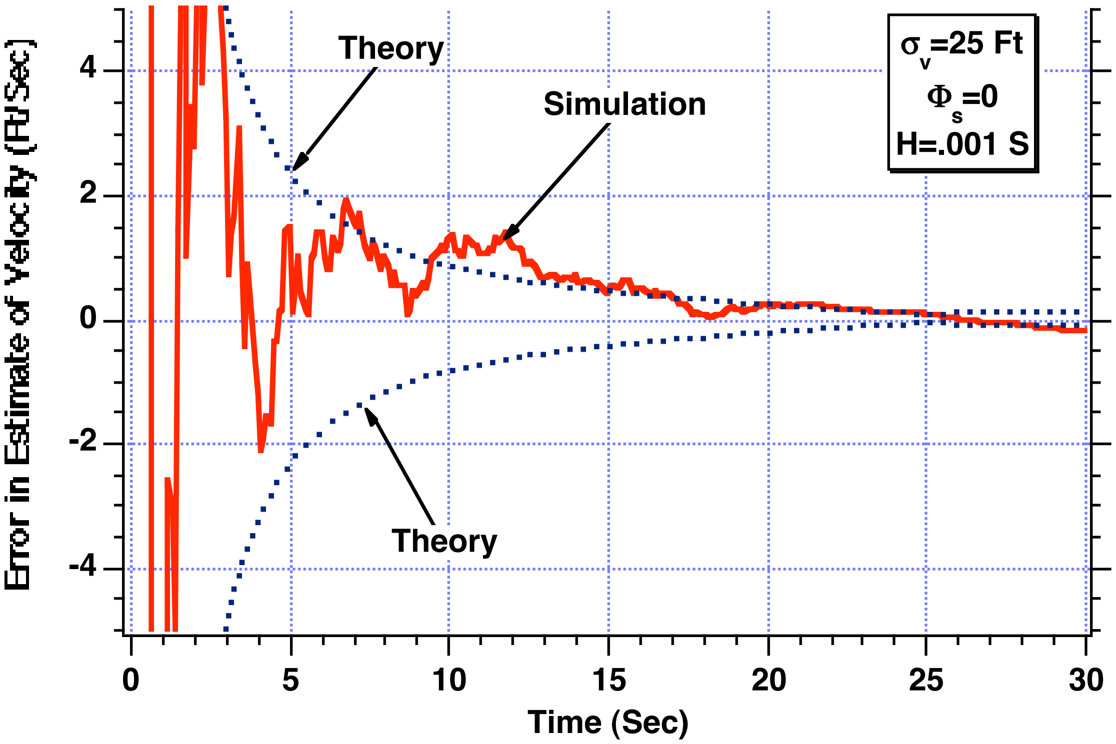
Routine for propagating states ahead one sampling interval

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

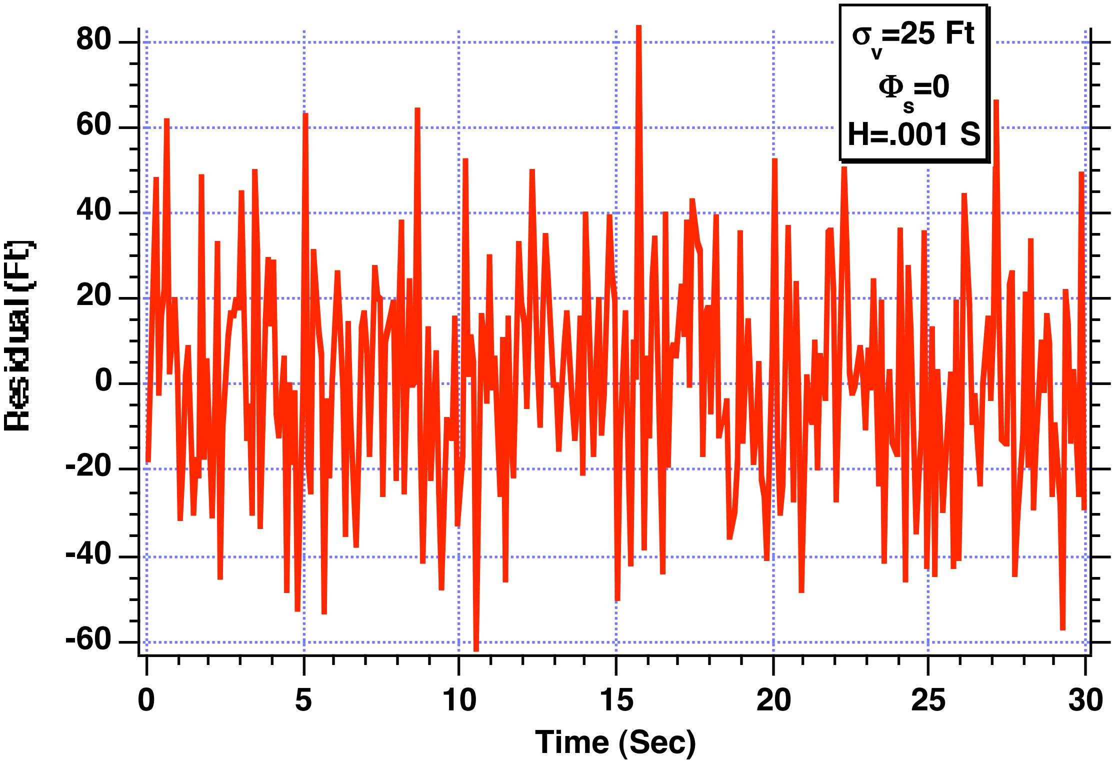
Divergence Has Been Eliminated in Altitude Estimate by Use of More Accurate State Propagation Methods



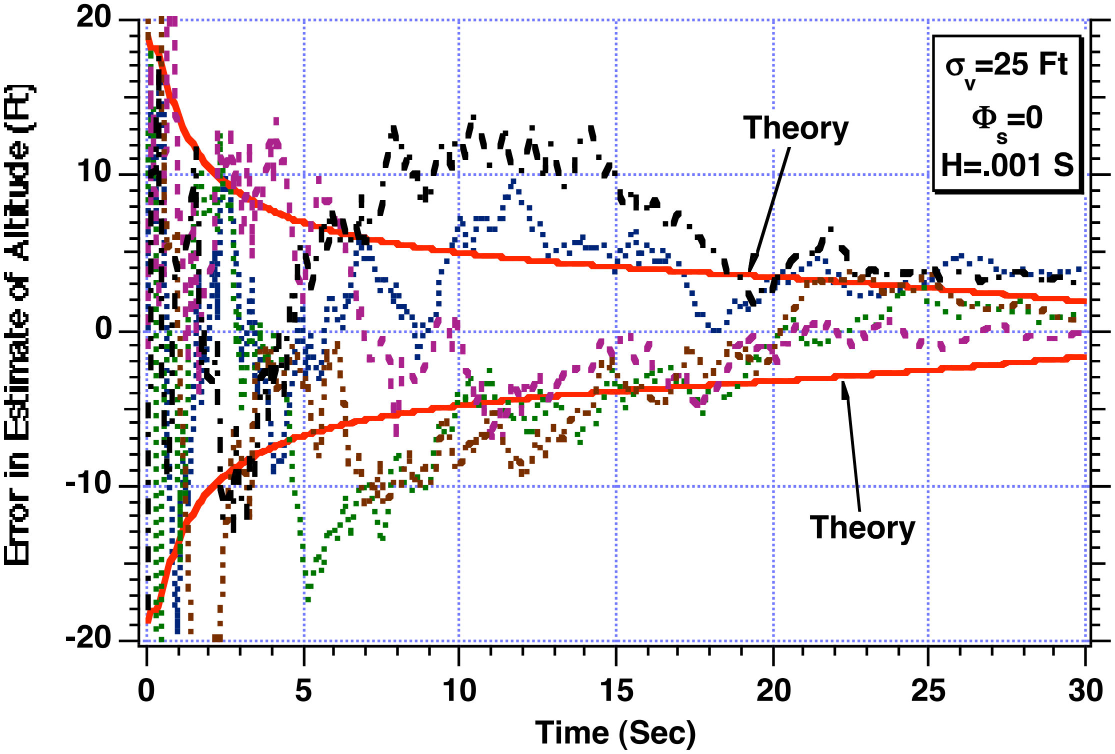
Divergence Has Been Eliminated in Velocity Estimate by Use of More Accurate State Propagation Methods



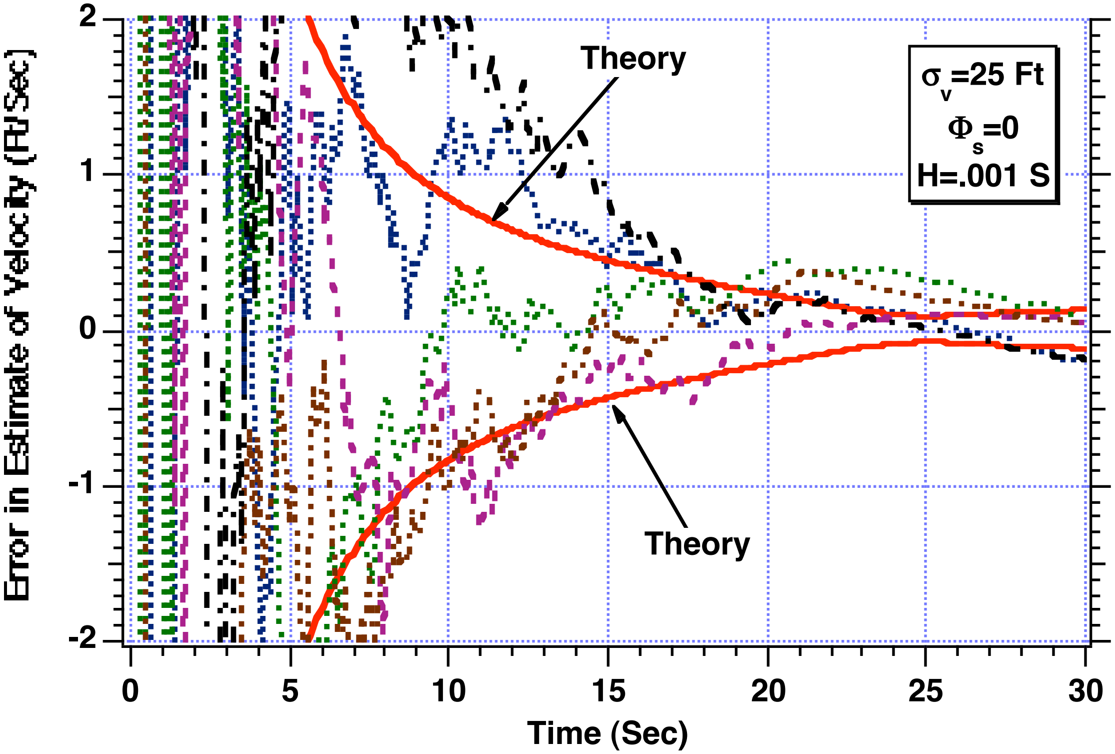
More Accurate State Propagation Ensures Residual has Zero Mean Even Though Kalman Gains Approach Zero



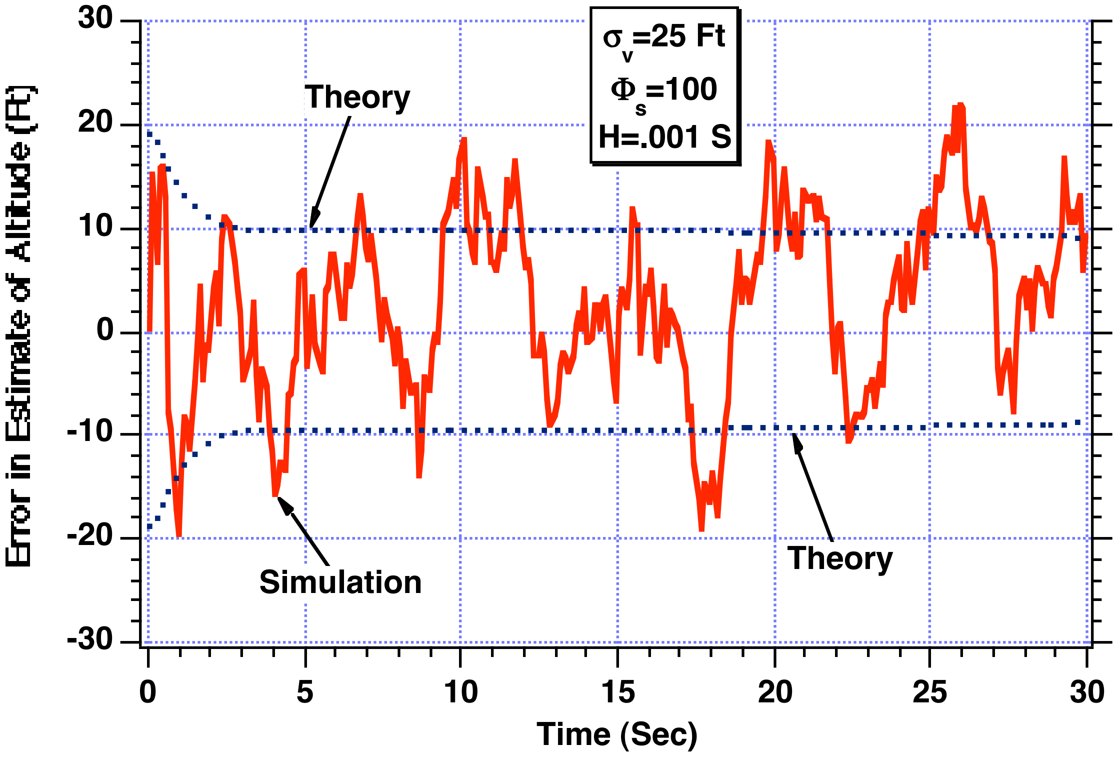
Monte Carlo Results are Within Theoretical Bounds for Error in Estimate of Position



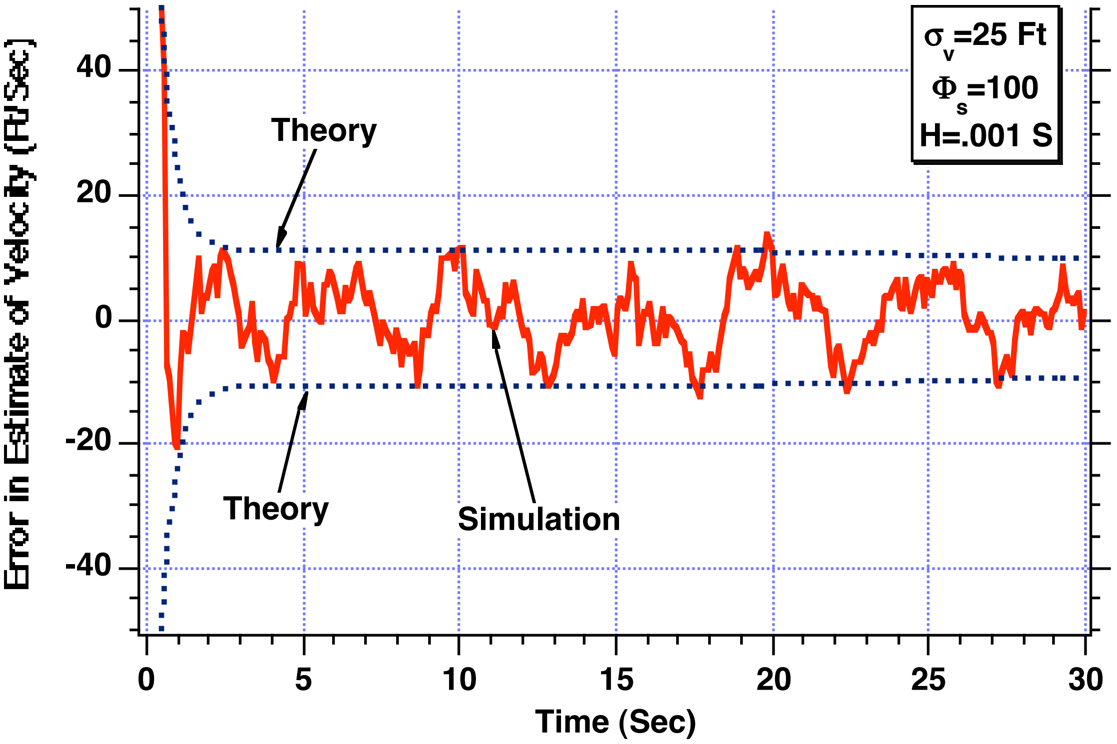
Monte Carlo Results are Within Theoretical Bounds for Error in Estimate of Velocity



Adding Process Noise Increases Error in Estimate of Position



Adding Process Noise Increases Error in Estimate of Velocity



Extended Kalman Filtering
Summary

- Extended Kalman filtering equations presented along with a numerical example
- Divergence problem was examined
    - Adding process noise removed divergence
    - Adding more terms to fundamental matrix approximation did not help
    - Using more accurate integration for state propagation helped but was more costly in terms of throughput