Practical Uses of Chain Rule For Solving Problems

Overview

- Using Newton's Method to Solve Equations Iteratively
- Equivalence Between Chain Rule and Method of Least Squares
- Nonlinear Least Squares
  - Three Dimensional GPS Example

Using Newton's Method to Solve Equations Iteratively

Newton's Method - Mathematical Analysis

Solve For a Root of

$$y = 25x^2 - 10x + 1 = 0$$

Finding Root Means Measured y or y* is zero

$$\Delta y = y^* - \hat{y} = 0 - \hat{y} = -\hat{y} = -(25\hat{x}^2 - 10\hat{x} + 1) \quad \textbf{Since} \quad \hat{y} = 25\hat{x}^2 - 10\hat{x} + 1$$

Chain Rule From Calculus

$$\Delta y = \frac{dy}{dx}\Delta x$$

Inverting Previous Expression Yields

$$\Delta x = \frac{\Delta y}{\frac{dy}{dx}} \quad \textbf{Where} \quad \frac{dy}{dx} = 50x - 10$$

Therefore

$$\Delta x = \hat{x}_{k+1} - \hat{x}_k = \frac{-\hat{y}_k}{50\hat{x}_k - 10} = \frac{-\left(25\hat{x}_k^2 - 10\hat{x}_k + 1\right)}{50\hat{x}_k - 10}$$

Iterate

$$\hat{x}_{k+1} = \hat{x}_k - \frac{\left(25\hat{x}_k^2 - 10\hat{x}_k + 1\right)}{50\hat{x}_k - 10}$$

Newton Method Simulation

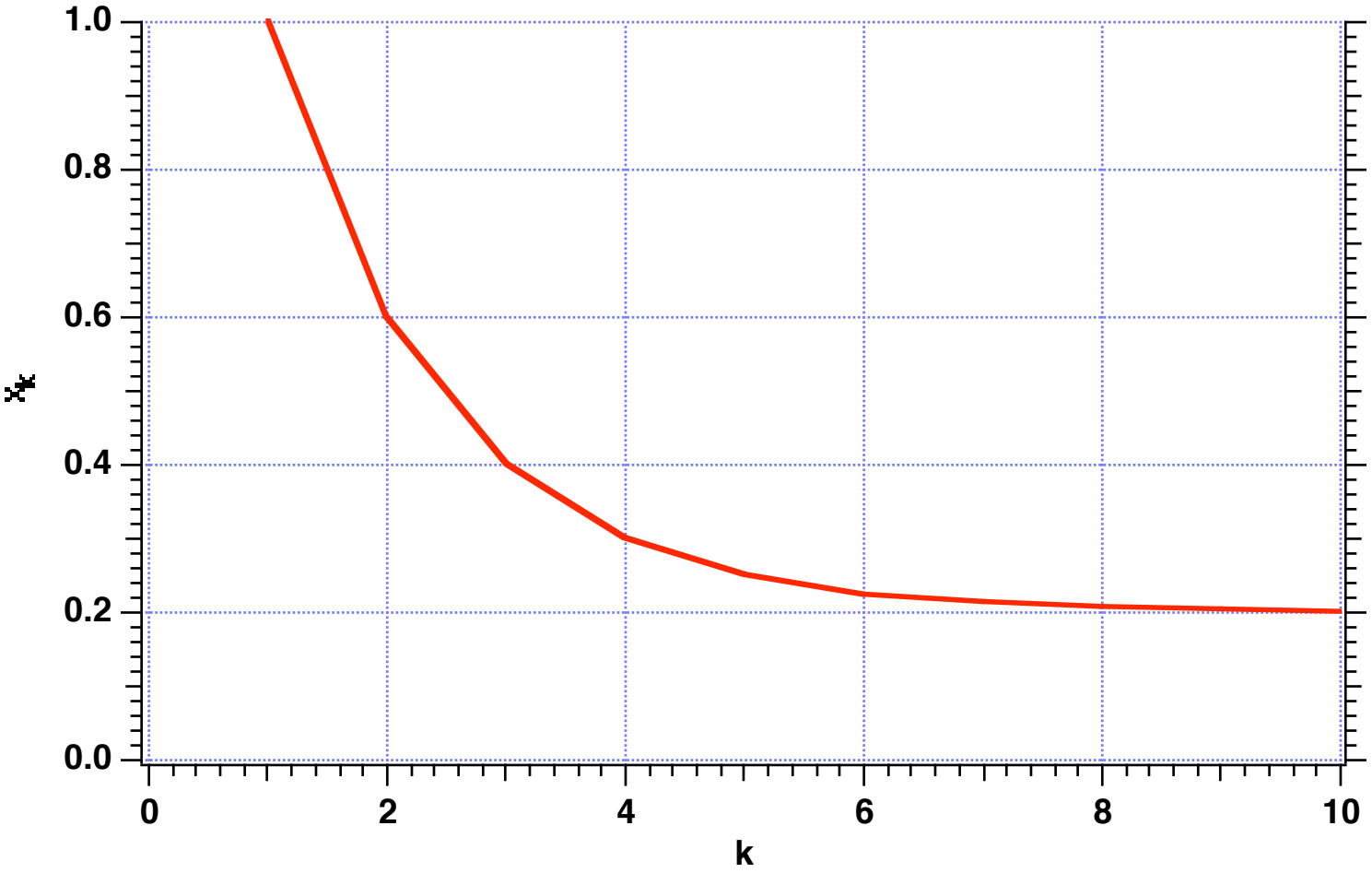
```
      IMPLICIT REAL*8 (A-H)
      IMPLICIT REAL*8 (O-Z)
      OPEN(1,STATUS='UNKNOWN',FILE='DATFIL')
      JJ=1
      XH=1.
      WRITE(9,*)JJ,XH
      WRITE(1,*)JJ,XH
      DO 10 JJ=2,10
      YH=25.*XH*XH-10.*XH+1.
      DELY=-YH
      DYDX=50.*XH-10.
      DELX=DELY/DYDX
      XH=DELX+XH
      WRITE(9,*)JJ,XH
      WRITE(1,*)JJ,XH
10    CONTINUE
      CLOSE(1)
      PAUSE
      END
```

Initial Guess

$$\hat{y} = 25\hat{x}^2 - 10\hat{x} + 1$$

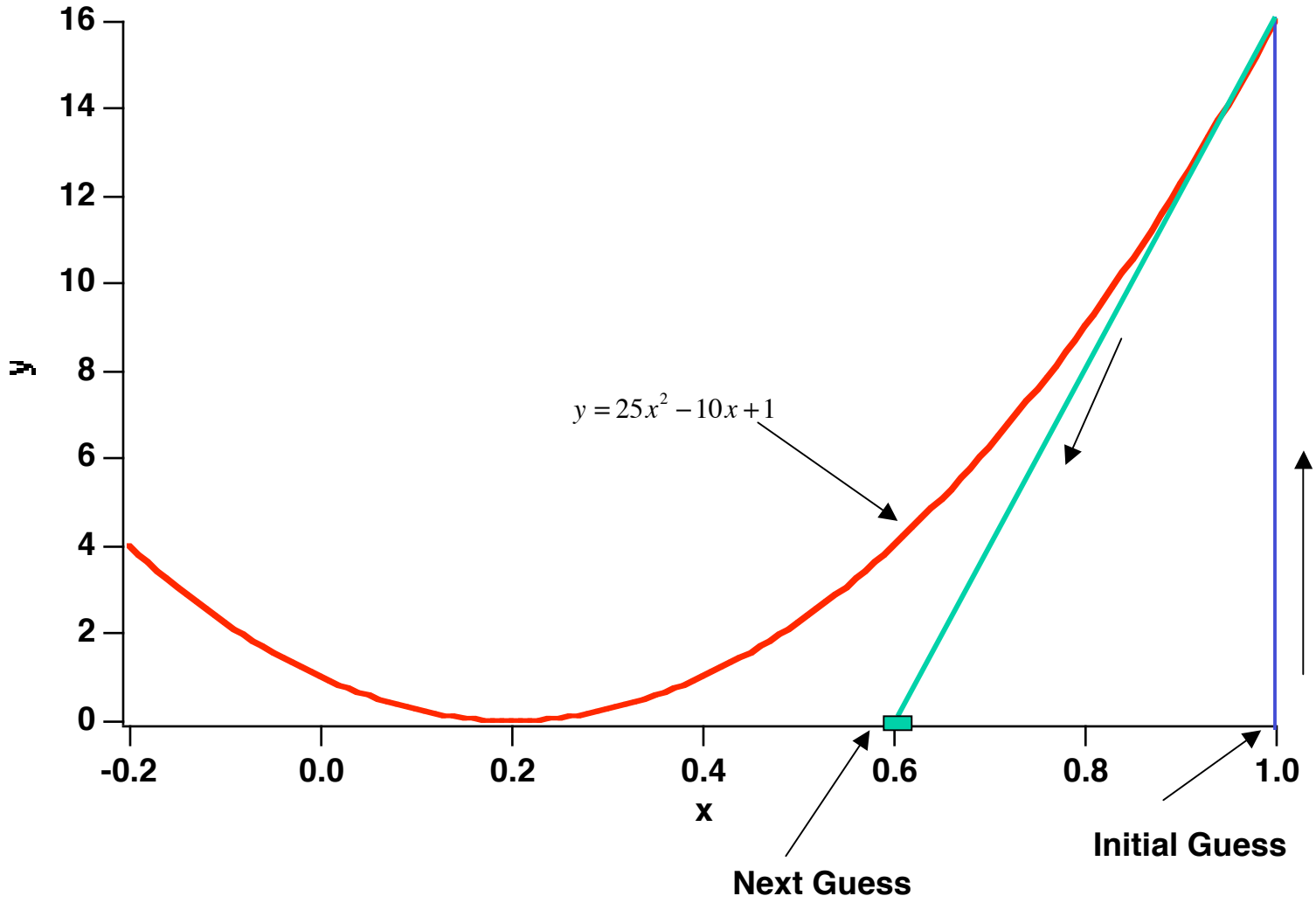
$$\Delta x = \frac{\Delta y}{\frac{dy}{dx}}$$

Iterative Method Approaches Exact Solution

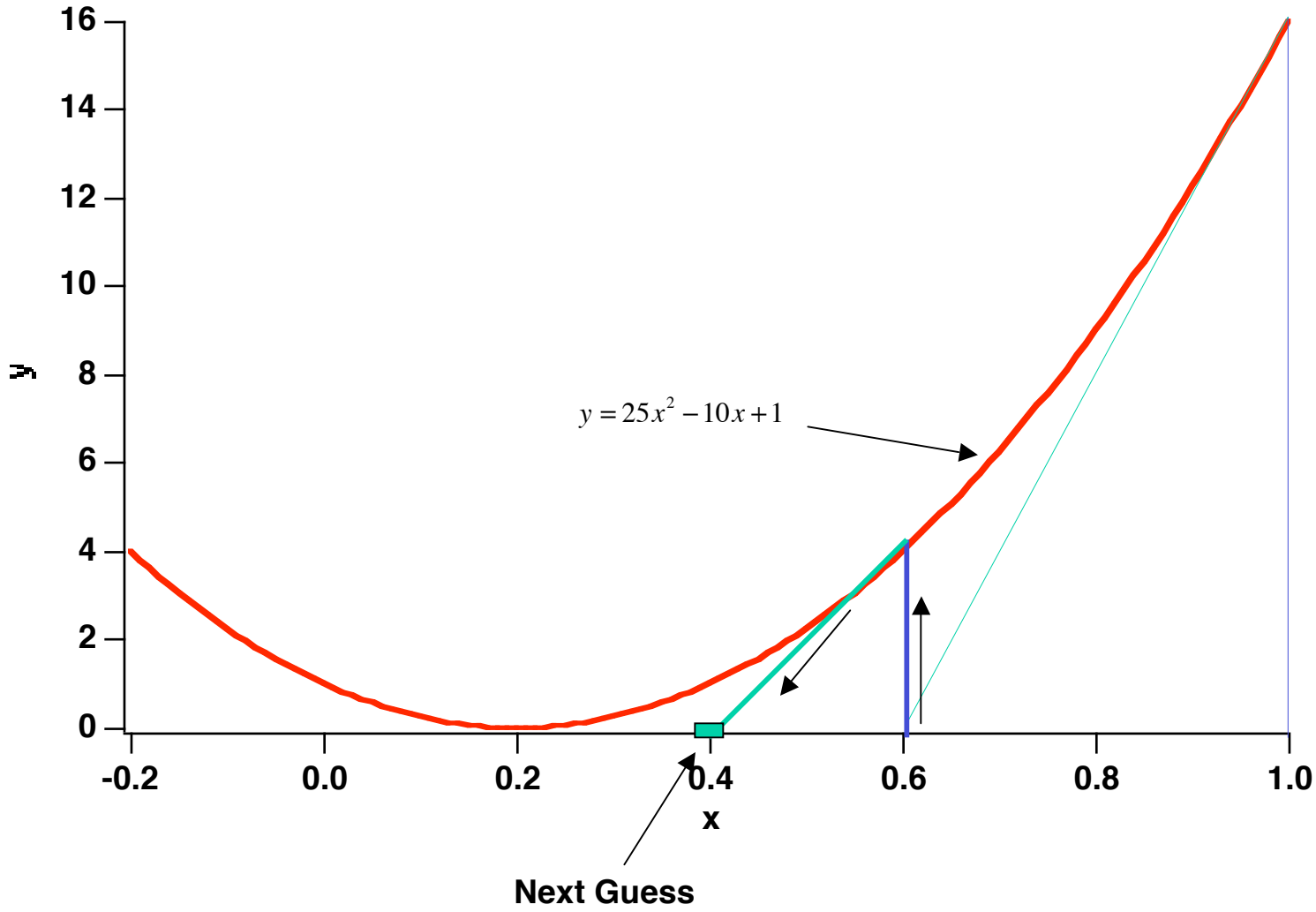


x=0.2 solves $y = 25x^2 - 10x + 1 = 0$

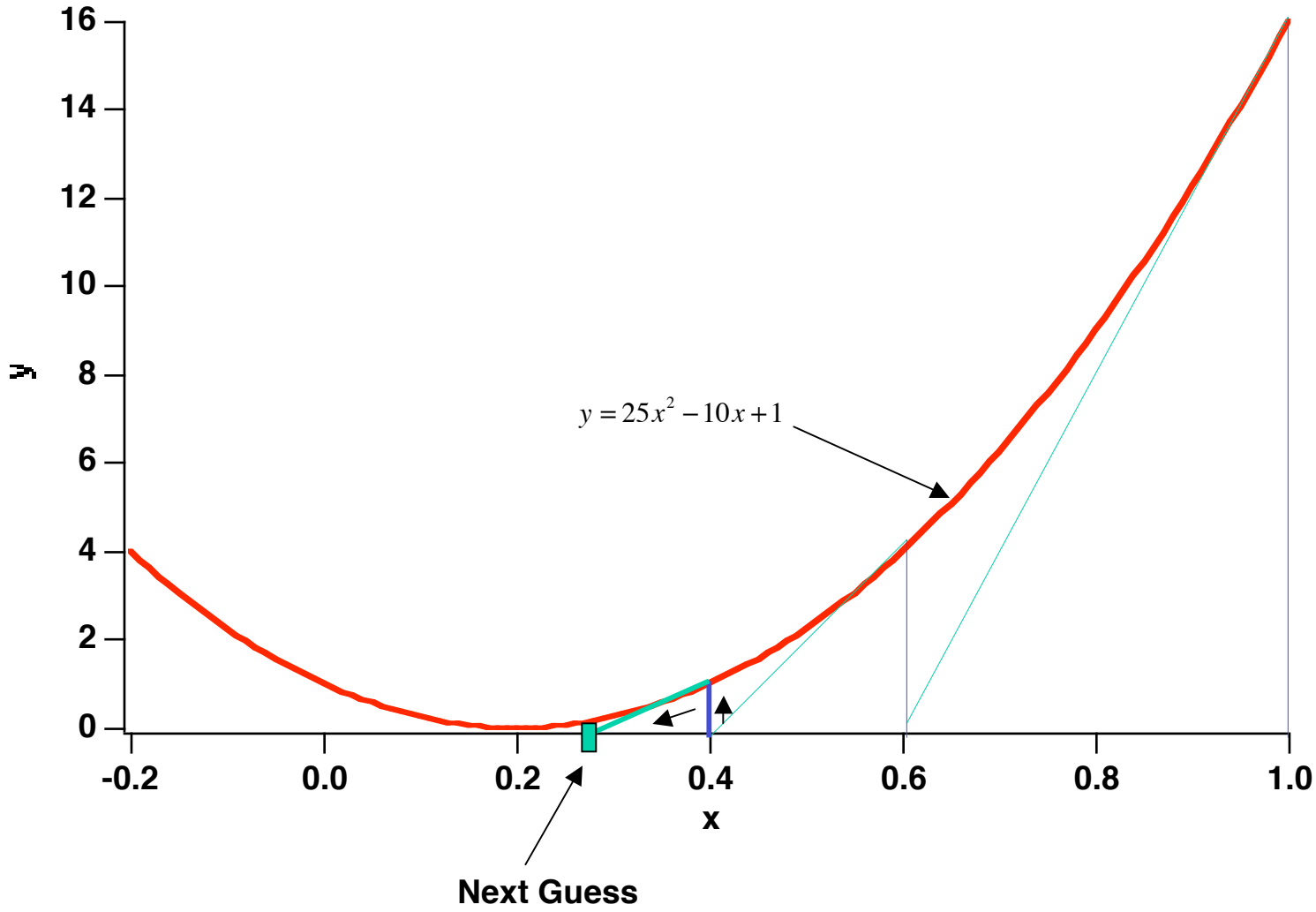
Newton's Method - Graphical Interpretation
What x Causes y to Go to Zero?



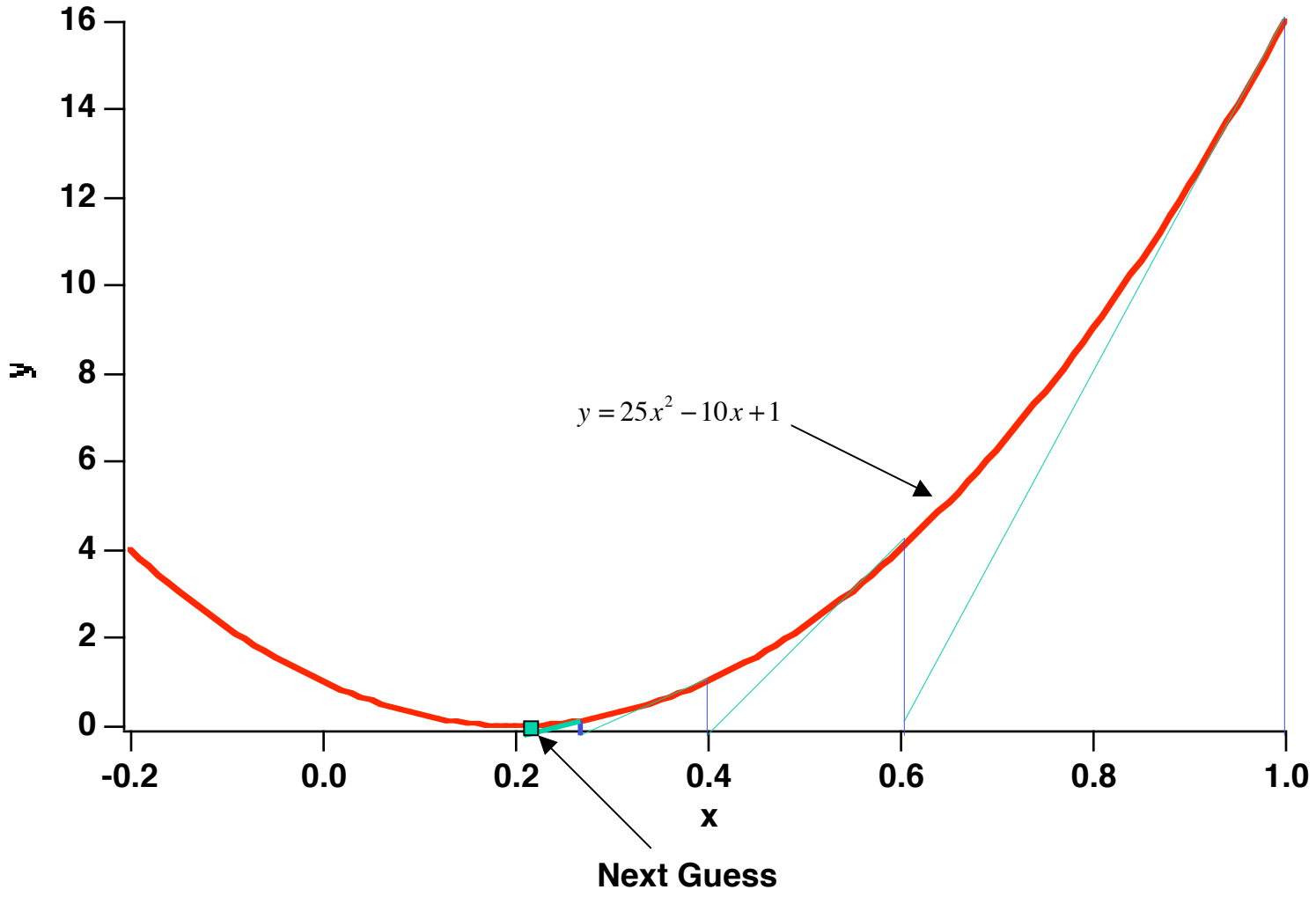
Newton's Method - Graphical Interpretation
What x Causes y to Go to Zero?



Newton's Method - Graphical Interpretation
What x Causes y to Go to Zero?

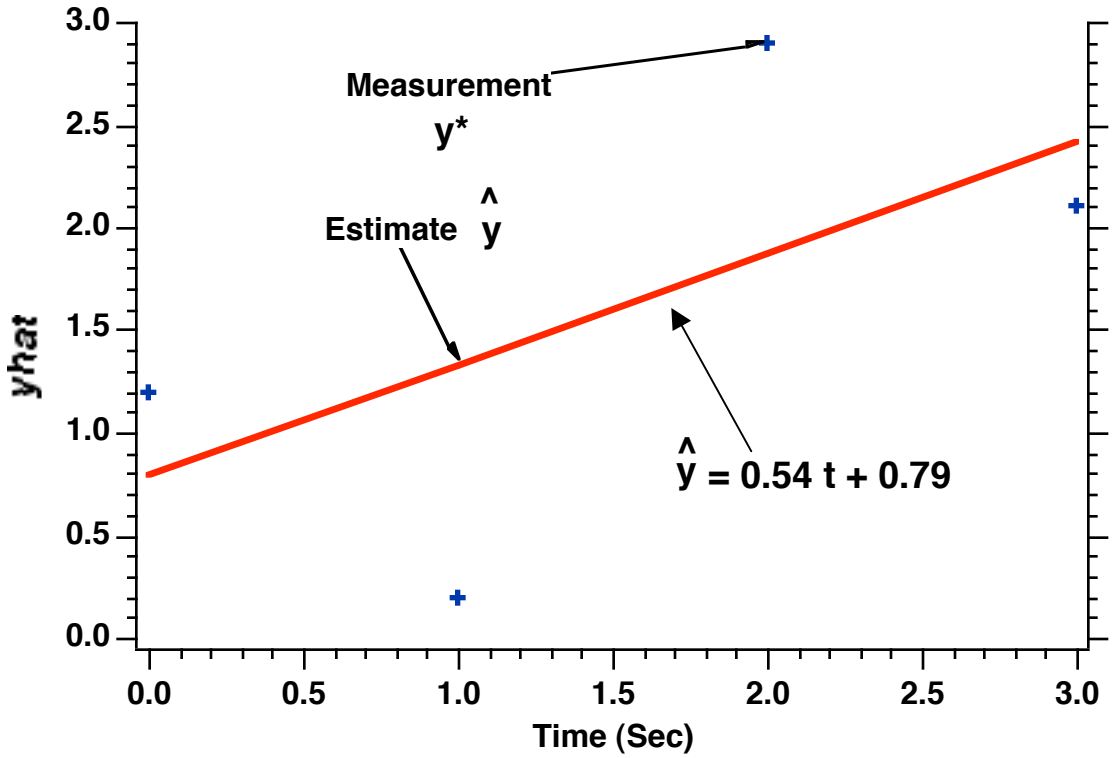


Newton's Method - Graphical Interpretation
What x Causes y to Go to Zero?



Equivalence Between Chain Rule and Method of Least Squares

Review of Method of Least Squares (Fit 4 Measurements With Straight Line)



t	y*
0	1.2
1	0.2
2	2.9
3	2.1

These results are from second lecture where method of least squares was introduced

Using Chain Rule From Calculus -1

Want to Fit the Four Measurements With Straight Line of Form

$$y = at + b$$

Chain Rule

$$\Delta y = \frac{\partial y}{\partial a}\Delta a + \frac{\partial y}{\partial b}\Delta b$$

For This Example

$$\frac{\partial y}{\partial a} = t$$

$$\frac{\partial y}{\partial b} = 1$$

If We Have Four Measurements We Can Say

t	y*	$\frac{\partial y}{\partial a}$	$\frac{\partial y}{\partial b}$
0	1.2	0	1
1	0.2	1	1
2	2.9	2	1
3	2.1	3	1

From 2nd Lecture

Using Chain Rule From Calculus -2

Expressing the Chain Rule With Matrix Notation We Obtain

$$\begin{bmatrix}\Delta y_1\\ \Delta y_2\\ \Delta y_3\\ \Delta y_4\end{bmatrix} = \underbrace{\begin{bmatrix}0 & 1\\ 1 & 1\\ 2 & 1\\ 3 & 1\end{bmatrix}}_{A}\begin{bmatrix}\Delta a\\ \Delta b\end{bmatrix}$$

We want to solve for Δa and Δb but can't take inverse of A since A is not square matrix

Use Pseudo Inverse

$$\begin{bmatrix}\Delta y_1\\ \Delta y_2\\ \Delta y_3\\ \Delta y_4\end{bmatrix} = A\begin{bmatrix}\Delta a\\ \Delta b\end{bmatrix}$$

Multiply Both Sides By A^T

$$A^T\begin{bmatrix}\Delta y_1\\ \Delta y_2\\ \Delta y_3\\ \Delta y_4\end{bmatrix} = A^T A\begin{bmatrix}\Delta a\\ \Delta b\end{bmatrix}$$

A^TA is a square matrix

Multiply Both Sides By $(A^TA)^{-1}$

$$\left(A^T A\right)^{-1}A^T\begin{bmatrix}\Delta y_1\\ \Delta y_2\\ \Delta y_3\\ \Delta y_4\end{bmatrix} = \begin{bmatrix}\Delta a\\ \Delta b\end{bmatrix}$$

Or

$$\boxed{\begin{bmatrix}\Delta a\\ \Delta b\end{bmatrix} = \left(A^T A\right)^{-1}A^T\begin{bmatrix}\Delta y_1\\ \Delta y_2\\ \Delta y_3\\ \Delta y_4\end{bmatrix}}$$

Numerical Details

Recall

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 1 \\ 2 & 1 \\ 3 & 1 \end{bmatrix}$$

Multiplying By Transpose Yields Matrix That is Invertible

$$A^T A = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 1 \\ 2 & 1 \\ 3 & 1 \end{bmatrix} = \begin{bmatrix} 14 & 6 \\ 6 & 4 \end{bmatrix}$$

A^TA is a square matrix

Taking Inverse Yields

$$\left(A^T A\right)^{-1} = \begin{bmatrix} 14 & 6 \\ 6 & 4 \end{bmatrix}^{-1} = \begin{bmatrix} 0.2 & -0.3 \\ -0.3 & 0.7 \end{bmatrix}$$

Use This Equation In Code

Finally We Obtain

$$\left(A^T A\right)^{-1} A^T = \begin{bmatrix} 0.2 & -0.3 \\ -0.3 & 0.7 \end{bmatrix} \begin{bmatrix} 0 & 1 & 2 & 3 \\ 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} -0.3 & -0.1 & 0.1 & 0.3 \\ 0.7 & 0.4 & 0.1 & -0.2 \end{bmatrix}$$

Therefore

$$\begin{bmatrix} \Delta a \\ \Delta b \end{bmatrix} = \left(A^T A\right)^{-1} A^T \begin{bmatrix} \Delta y_1 \\ \Delta y_2 \\ \Delta y_3 \\ \Delta y_4 \end{bmatrix} = \begin{bmatrix} -0.3 & -0.1 & 0.1 & 0.3 \\ 0.7 & 0.4 & 0.1 & -0.2 \end{bmatrix} \begin{bmatrix} \Delta y_1 \\ \Delta y_2 \\ \Delta y_3 \\ \Delta y_4 \end{bmatrix} = \begin{bmatrix} -0.3\Delta y_1 - 0.1\Delta y_2 + 0.1\Delta y_3 + 0.3\Delta y_4 \\ 0.7\Delta y_1 + 0.4\Delta y_2 + 0.1\Delta y_3 - 0.2\Delta y_4 \end{bmatrix}$$

Also Recall

Straight Line

$$y = at + b$$

Therefore

$$\hat{y} = \hat{a}t + \hat{b}$$

At The Four Different Measurement Times (t=0,1,2,3)

$$\hat{y}(0) = \hat{y}_1 = \hat{b}$$

$$\hat{y}(1) = \hat{y}_2 = \hat{a} + \hat{b}$$

$$\hat{y}(2) = \hat{y}_3 = 2\hat{a} + \hat{b}$$

$$\hat{y}(3) = \hat{y}_4 = 3\hat{a} + \hat{b}$$

Use These Equations In Code

Or in Matrix Form

$$\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \\ \hat{y}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \\ 2 & 1 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} \hat{a} \\ \hat{b} \end{bmatrix} = A \begin{bmatrix} \hat{a} \\ \hat{b} \end{bmatrix}$$

Code For Solving Straight Line By Iteration

```
      IMPLICIT REAL*8(A-H,O-Z)
      Y1S=1.2
      Y2S=.2
      Y3S=2.9
      Y4S=2.1
      AH=2.
      BH=-2.
      DO 10 I=1,5
      Y1H=BH
      Y2H=AH+BH
      Y3H=2.*AH+BH
      Y4H=3.*AH+BH
      DELY1=Y1S-Y1H
      DELY2=Y2S-Y2H
      DELY3=Y3S-Y3H
      DELY4=Y4S-Y4H
      AH=AH-.3*DELY1-.1*DELY2+.1*DELY3+.3*DELY4
      BH=BH+.7*DELY1+.4*DELY2+.1*DELY3-.2*DELY4
      WRITE(9,*)I,AH,BH
10    CONTINUE
      PAUSE
      END
```

Measurements

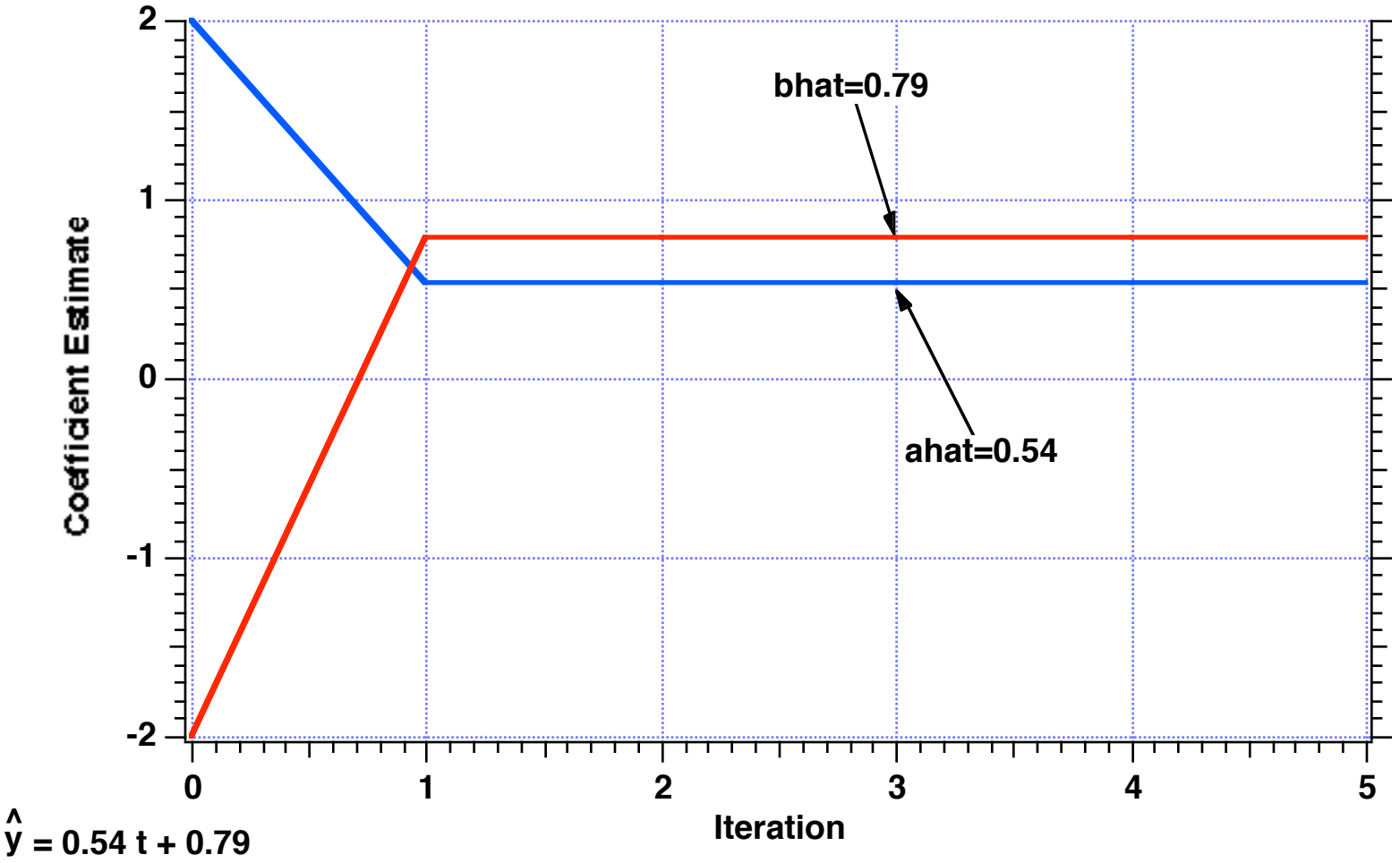
Initial Guess at Coefficients

Estimates of y Based On Coefficient Estimates

Compute Δy For Each Measurement

Update Coefficient Estimates

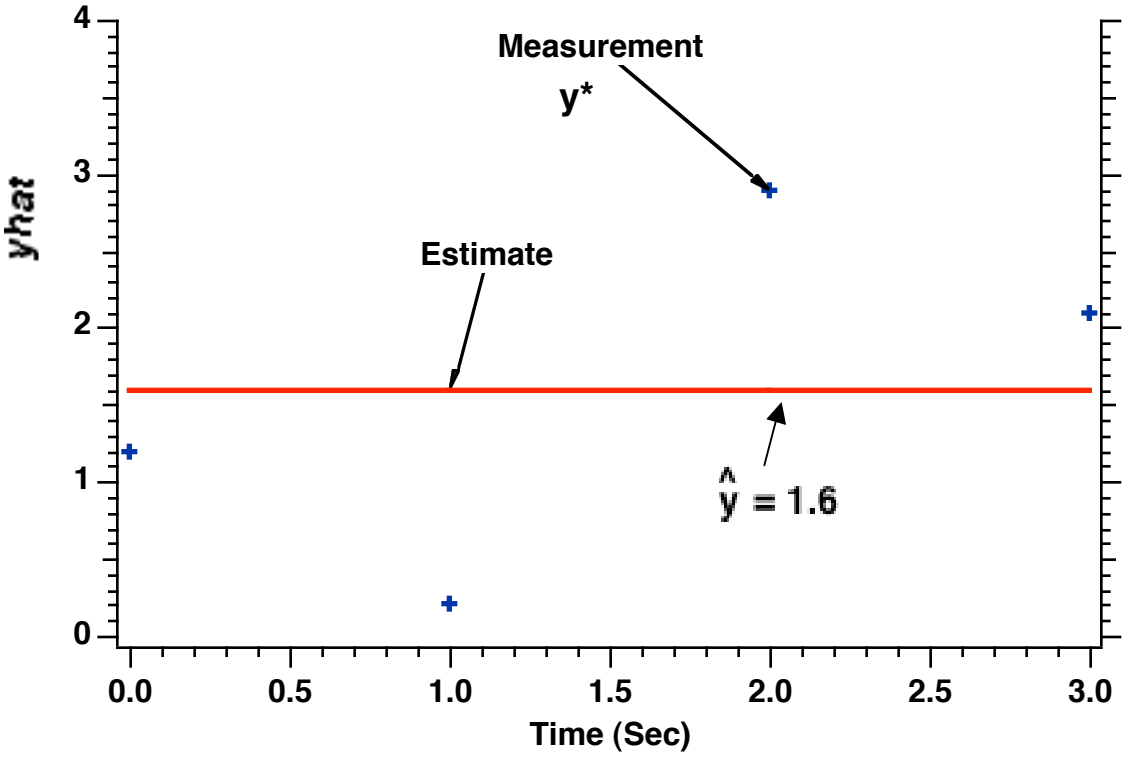
Straight Line Coefficient Estimates Converge Immediately and Are Independent of Initial Estimates



$\hat{y} = 0.54\ t + 0.79$

This is the same equation as obtained by the method of least squares in Lecture 2

Review of Method of Least Squares (Fit 4 Measurements With Constant)



t	y*
0	1.2
1	0.2
2	2.9
3	2.1

These results are from second lecture where method of least squares was introduced

Using Chain Rule From Calculus -1

Want to Fit Four Measurements With Constant of Form

$$y = a$$

Chain Rule

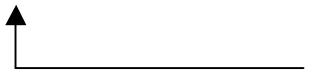
$$\Delta y = \frac{\partial y}{\partial a}\Delta a$$

For This Example

$$\frac{\partial y}{\partial a} = 1$$

If We Have Four Measurements We Can Say

t	y*	$\frac{\partial y}{\partial a}$
0	1.2	1
1	0.2	1
2	2.9	1
3	2.1	1



From 2nd Lecture

Using Chain Rule From Calculus -2

Therefore From Chain Rule

$$\begin{bmatrix}\Delta y_1\\ \Delta y_2\\ \Delta y_3\\ \Delta y_4\end{bmatrix} = \underbrace{\begin{bmatrix}1\\ 1\\ 1\\ 1\end{bmatrix}}_{\mathbf{A}} \Delta a$$

We Have Also Shown

$$\Delta a = \left(A^T A\right)^{-1} A^T \begin{bmatrix}\Delta y_1\\ \Delta y_2\\ \Delta y_3\\ \Delta y_4\end{bmatrix}$$

Therefore

$$\Delta a = \left\{\begin{bmatrix}1 & 1 & 1 & 1\end{bmatrix}\begin{bmatrix}1\\ 1\\ 1\\ 1\end{bmatrix}\right\}^{-1} \begin{bmatrix}1 & 1 & 1 & 1\end{bmatrix}\begin{bmatrix}\Delta y_1\\ \Delta y_2\\ \Delta y_3\\ \Delta y_4\end{bmatrix} = .25\begin{bmatrix}1 & 1 & 1 & 1\end{bmatrix}\begin{bmatrix}\Delta y_1\\ \Delta y_2\\ \Delta y_3\\ \Delta y_4\end{bmatrix} = .25\left(\Delta y_1 + \Delta y_2 + \Delta y_3 + \Delta y_4\right)$$

Use This Equation In Code

Also Recall

Constant

$$y = a$$

Therefore

$$\hat{y} = \hat{a}$$

At The Four Different Measurement Times (t=0,1,2,3)

$$\hat{y}(0) = \hat{y}_1 = \hat{a}$$

$$\hat{y}(1) = \hat{y}_2 = \hat{a}$$

$$\hat{y}(2) = \hat{y}_3 = \hat{a}$$

$$\hat{y}(3) = \hat{y}_4 = \hat{a}$$

Use These Equations In Code

Or in Matrix Form

$$\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \\ \hat{y}_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \hat{a} = A\hat{a}$$

Code For Solving Constant By Iteration

```
      IMPLICIT REAL*8(A-H,O-Z)
      OPEN(1,STATUS='UNKNOWN',FILE='DATFIL')
      Y1S=1.2
      Y2S=.2
      Y3S=2.9
      Y4S=2.1
      AH=2.
      I=0
      WRITE(9,*)I,AH
      WRITE(1,*)I,AH
      DO 10 I=1,5
      Y1H=AH
      Y2H=AH
      Y3H=AH
      Y4H=AH
      DELY1=Y1S-Y1H
      DELY2=Y2S-Y2H
      DELY3=Y3S-Y3H
      DELY4=Y4S-Y4H
      AH=AH+.25*(DELY1+DELY2+DELY3+DELY4)
      WRITE(9,*)I,AH
      WRITE(1,*)I,AH
10    CONTINUE
      PAUSE
      CLOSE(1)
      END
```

Measurements

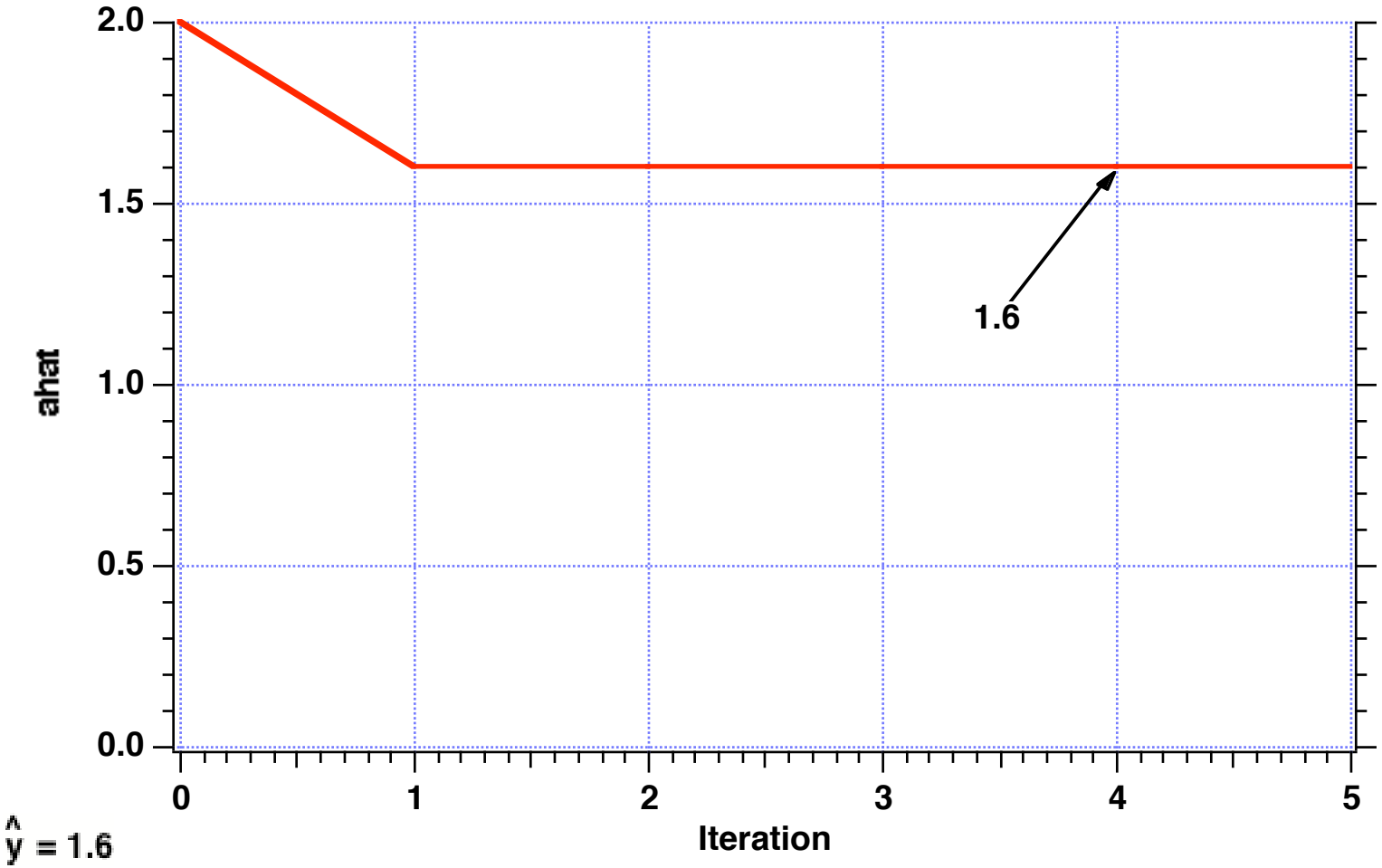
Initial Guess at Coefficient

Estimates of y Based On Initial Coefficient Estimates

Compute Δy For Each Measurement

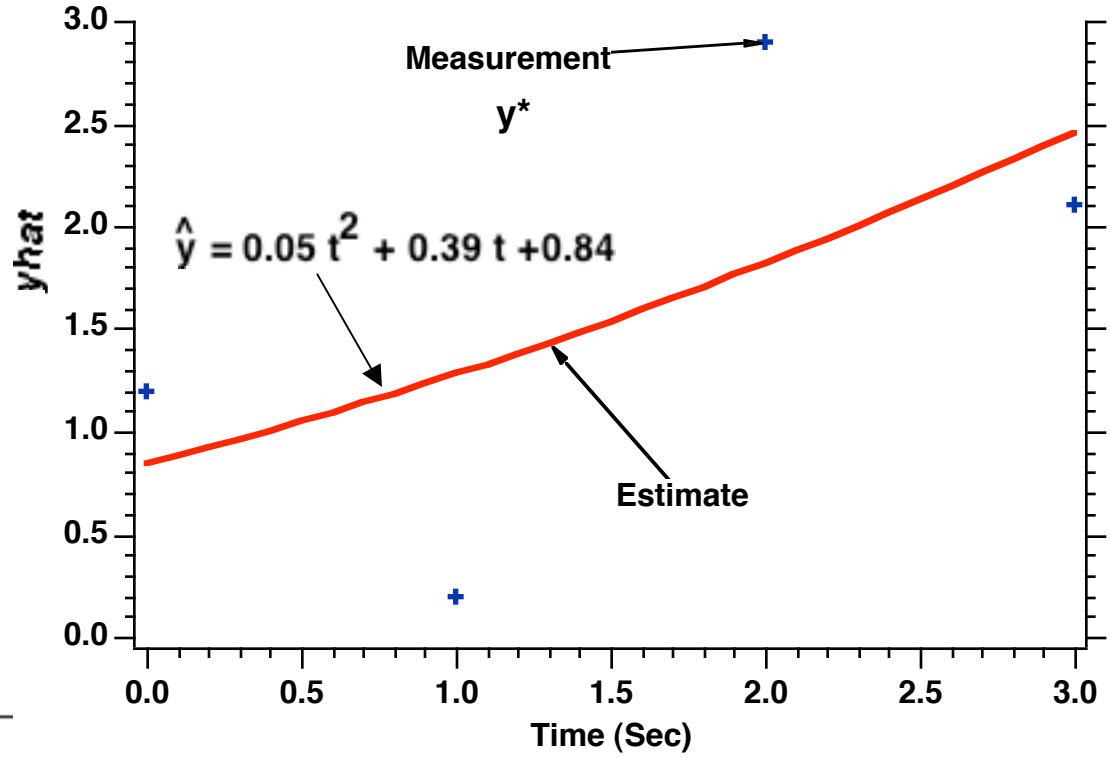
Update Coefficient Estimates

Constant Coefficient Estimate Converges Immediately and is Independent of Initial Estimate



This is the same equation as obtained by the method of least squares in Lecture 2

Review of Method of Least Squares (Fit 4 Measurements With Parabola)



t	y*
0	1.2
1	0.2
2	2.9
3	2.1

These results are from second lecture where method of least squares was introduced

Using Chain Rule From Calculus

Want to Fit Four Measurements With Parabola of Form

$$y = at^2 + bt + c$$

Chain Rule

$$\Delta y = \frac{\partial y}{\partial a}\Delta a + \frac{\partial y}{\partial b}\Delta b + \frac{\partial y}{\partial c}\Delta c$$

For This Example

$$\frac{\partial y}{\partial a} = t^2 \qquad \frac{\partial y}{\partial c} = 1$$

$$\frac{\partial y}{\partial b} = t$$

If We Have Four Measurements We Can Say

t	y*	$\frac{\partial y}{\partial a}$	$\frac{\partial y}{\partial b}$	$\frac{\partial y}{\partial c}$
0	1.2	0	0	1
1	0.2	1	1	1
2	2.9	4	2	1
3	2.1	9	3	1

From 2nd Lecture

Recall

Parabola

$$y = at^2 + bt + c$$

Therefore

$$\hat{y} = \hat{a}t^2 + \hat{b}t + \hat{c}$$

At The Four Different Measurement Times (t=0,1,2,3)

$$\hat{y}(0) = \hat{y}_1 = \hat{c}$$

$$\hat{y}(1) = \hat{y}_2 = \hat{a} + \hat{b} + \hat{c}$$

$$\hat{y}(2) = \hat{y}_3 = 4\hat{a} + 2\hat{b} + \hat{c}$$

$$\hat{y}(3) = \hat{y}_4 = 9\hat{a} + 3\hat{b} + \hat{c}$$

Use These Equations In Code

Or in Matrix Form

$$\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \\ \hat{y}_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 4 & 2 & 1 \\ 9 & 3 & 1 \end{bmatrix} \begin{bmatrix} \hat{a} \\ \hat{b} \\ \hat{c} \end{bmatrix} = A \begin{bmatrix} \hat{a} \\ \hat{b} \\ \hat{c} \end{bmatrix}$$

Iteration Scheme

Recall

$$A = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 4 & 2 & 1 \\ 9 & 3 & 1 \end{bmatrix} \quad \& \quad \begin{bmatrix} y_1^* \\ y_2^* \\ y_3^* \\ y_4^* \end{bmatrix} = \begin{bmatrix} 1.2 \\ 0.2 \\ 2.9 \\ 2.1 \end{bmatrix}$$

Start with $\begin{bmatrix} \hat{a} \\ \hat{b} \\ \hat{c} \end{bmatrix} = \begin{bmatrix} 2 \\ -2 \\ 5 \end{bmatrix}$

Could be Anything

Loop

$$\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \\ \hat{y}_4 \end{bmatrix} = A \begin{bmatrix} \hat{a} \\ \hat{b} \\ \hat{c} \end{bmatrix}$$

$$\begin{bmatrix} \Delta y_1 \\ \Delta y_2 \\ \Delta y_3 \\ \Delta y_4 \end{bmatrix} = \begin{bmatrix} y_1^* \\ y_2^* \\ y_3^* \\ y_4^* \end{bmatrix} - \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \\ \hat{y}_4 \end{bmatrix}$$

Iterate

Recognizing That

$$\begin{bmatrix} \Delta a \\ \Delta b \\ \Delta c \end{bmatrix} = \left(A^T A\right)^{-1} A^T \begin{bmatrix} \Delta y_1 \\ \Delta y_2 \\ \Delta y_3 \\ \Delta y_4 \end{bmatrix}$$

Update Coefficient Estimates

$$\begin{bmatrix} \hat{a} \\ \hat{b} \\ \hat{c} \end{bmatrix} = \begin{bmatrix} \Delta a \\ \Delta b \\ \Delta c \end{bmatrix} + \begin{bmatrix} \hat{a} \\ \hat{b} \\ \hat{c} \end{bmatrix}$$

Code For Solving Parabola By Iteration - 1

```
IMPLICIT REAL*8(A-H,O-Z)
REAL*8 A(4,3),AINV(3,4),B(4,1),ANS(3,1),AT(3,4),CMAT(3,3)
REAL*8 CINV(3,3)
OPEN(1,STATUS='UNKNOWN',FILE='DATFIL')
Y1S=1.2
Y2S=.2
Y3S=2.9
Y4S=2.1
AH=2.
BH=-2.
CH=5.
I=0
WRITE(9,*)I,AH,BH,CH
WRITE(1,*)I,AH,BH,CH
DO 10 I=1,5
A(1,1)=0.
A(1,2)=0.
A(1,3)=1.
A(2,1)=1.
A(2,2)=1.
A(2,3)=1.
A(3,1)=4.
A(3,2)=2.
A(3,3)=1.
A(4,1)=9.
A(4,2)=3.
A(4,3)=1.
```

Measurements

Initial Guess at Coefficient

A

Code For Solving Parabola By Iteration - 2

```
      Y1H=CH
      Y2H=AH+BH+CH
      Y3H=4.*AH+2.*BH+CH
      Y4H=9.*AH+3.*BH+CH
      DELY1=Y1S-Y1H
      DELY2=Y2S-Y2H
      DELY3=Y3S-Y3H
      DELY4=Y4S-Y4H
      B(1,1)=DELY1
      B(2,1)=DELY2
      B(3,1)=DELY3
      B(4,1)=DELY4
      CALL MATTRN(A,4,3,AT)
      CALL MATMUL(AT,3,4,A,4,3,CMAT)
      CALL MTINV(CMAT,3,CINV)
      CALL MATMUL(CINV,3,3,AT,3,4,AINV)
      CALL MATMUL(AINV,3,4,B,4,1,ANS)
      DELA=ANS(1,1)
      DELB=ANS(2,1)
      DELC=ANS(3,1)
      AH=DELA+AH
      BH=DELB+BH
      CH=DELC+CH
      WRITE(9,*)I,AH,BH,CH
      WRITE(1,*)I,AH,BH,CH
10    CONTINUE
      PAUSE
      CLOSE(1)
      END
```

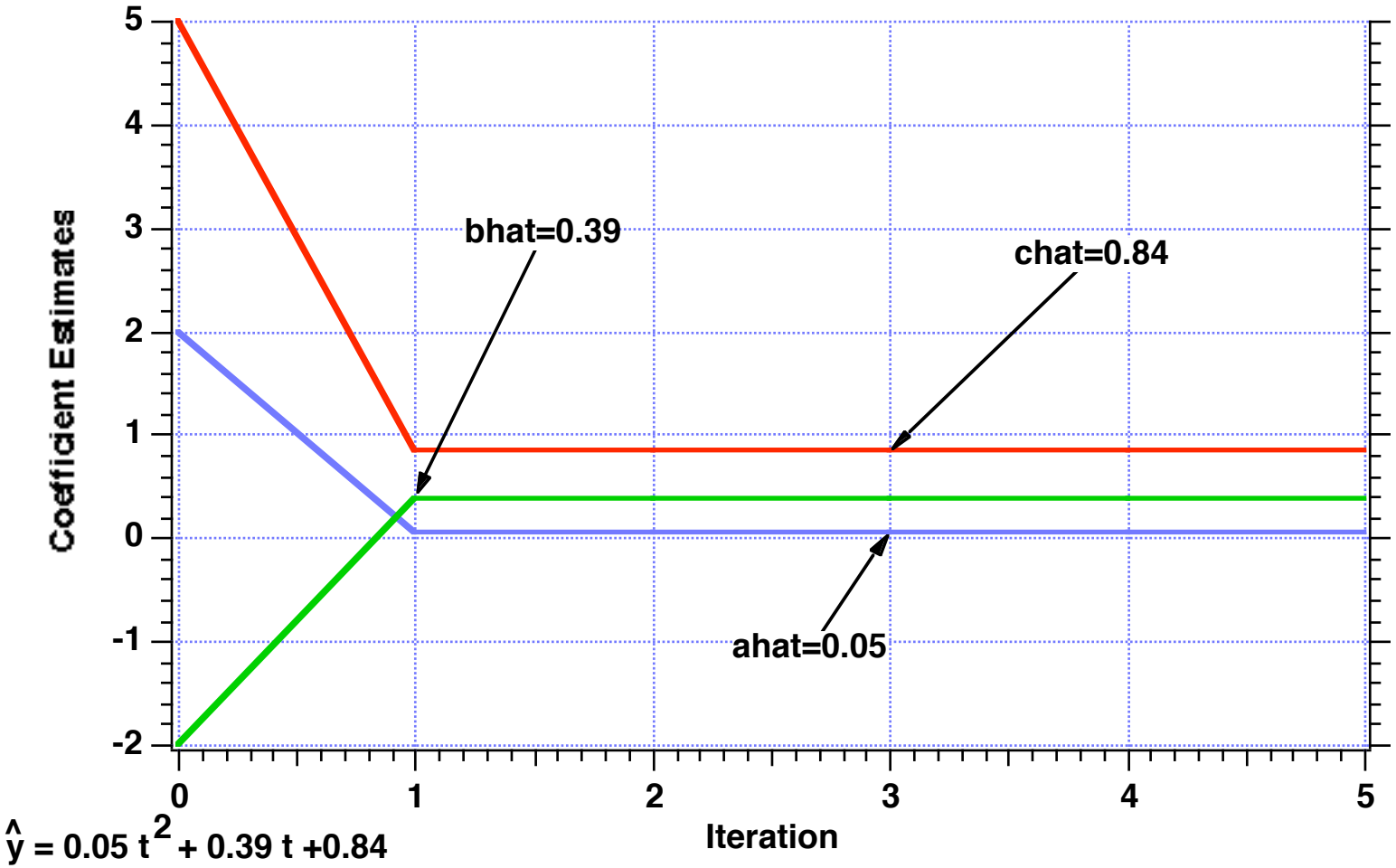
Estimates of y Based On Initial Coefficient Estimates

Compute Δy For Each Measurement

$$\begin{bmatrix}\Delta a\\ \Delta b\\ \Delta c\end{bmatrix} = \left(A^T A\right)^{-1} A^T \begin{bmatrix}\Delta y_1\\ \Delta y_2\\ \Delta y_3\\ \Delta y_4\end{bmatrix}$$

Update Coefficient Estimates

Parabola Coefficient Estimates Converge Immediately and Are Independent of Initial Estimates



$\hat{y} = 0.05\ t^2 + 0.39\ t + 0.84$

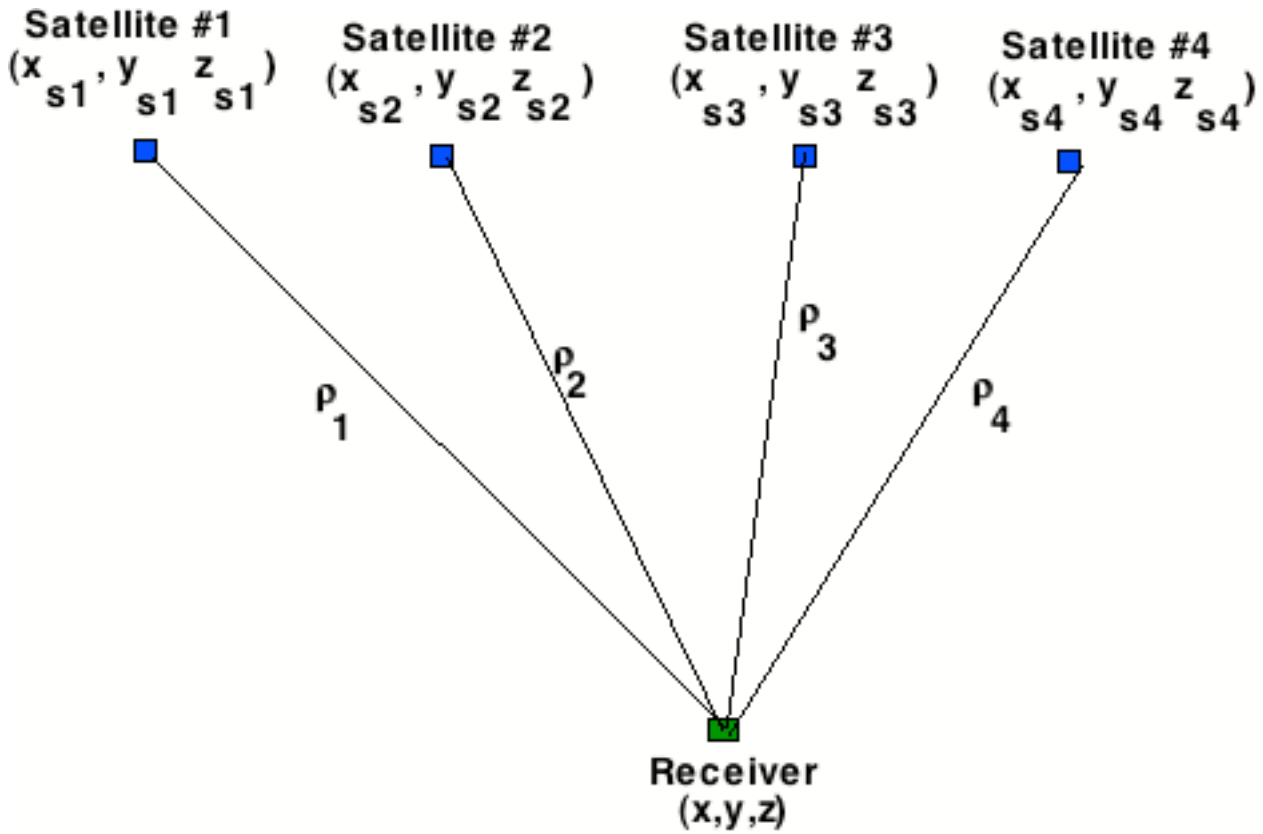
This is the same equation as obtained by the method of least squares in Lecture 2

Nonlinear Least Squares (Three Dimensional GPS Example)

Recall

- In 2 dimensional flat earth GPS example we had closed-form solutions for receiver position in terms of two range measurements and we some how initialized the Extended Kalman Filter so that the initial receiver location was known to a few thousand feet in the presence of 300 ft of range measurement noise
- We would like to see how to solve for receiver position when many satellites are available and how this information can be used to initialize an Extended Kalman filter

Satellite Geometry For GPS



Satellite locations known
Receiver location unknown
Distances from receiver to each satellite measured

Pseudorange Measurements to 4 Satellites

4 Pseudorange Measurements

$$\rho_1 = \sqrt{(x_{s1} - x)^2 + (y_{s1} - y)^2 + (z_{s1} - z)^2} + c\tau$$

$$\rho_2 = \sqrt{(x_{s2} - x)^2 + (y_{s2} - y)^2 + (z_{s2} - z)^2} + c\tau$$

$$\rho_3 = \sqrt{(x_{s3} - x)^2 + (y_{s3} - y)^2 + (z_{s3} - z)^2} + c\tau$$

$$\rho_4 = \sqrt{(x_{s4} - x)^2 + (y_{s4} - y)^2 + (z_{s4} - z)^2} + c\tau$$

We Have 4 Equations With 4 Unknowns (x, y, z ,τ). How Can We Solve These Equations?

Chain Rule From Calculus

$$\Delta\rho_1 = \frac{\partial\rho_1}{\partial x}\Delta x + \frac{\partial\rho_1}{\partial y}\Delta y + \frac{\partial\rho_1}{\partial z}\Delta z + \frac{\partial\rho_1}{\partial\tau}\Delta\tau$$

$$\Delta\rho_2 = \frac{\partial\rho_2}{\partial x}\Delta x + \frac{\partial\rho_2}{\partial y}\Delta y + \frac{\partial\rho_2}{\partial z}\Delta z + \frac{\partial\rho_2}{\partial\tau}\Delta\tau$$

$$\Delta\rho_3 = \frac{\partial\rho_3}{\partial x}\Delta x + \frac{\partial\rho_3}{\partial y}\Delta y + \frac{\partial\rho_3}{\partial z}\Delta z + \frac{\partial\rho_3}{\partial\tau}\Delta\tau$$

$$\Delta\rho_4 = \frac{\partial\rho_4}{\partial x}\Delta x + \frac{\partial\rho_4}{\partial y}\Delta y + \frac{\partial\rho_4}{\partial z}\Delta z + \frac{\partial\rho_4}{\partial\tau}\Delta\tau$$

Taking Partial Derivatives

Recall

$$\rho_1 = \sqrt{(x_{s1} - x)^2 + (y_{s1} - y)^2 + (z_{s1} - z)^2} + c\tau$$

Therefore

$$\frac{\partial \rho_1}{\partial x} = \frac{1}{2}\left[(x_{s1} - x)^2 + (y_{s1} - y)^2 + (z_{s1} - z)^2\right]^{-5} 2(x_{s1} - x)(-1) = \frac{-(x_{s1} - x)}{\sqrt{(x_{s1} - x)^2 + (y_{s1} - y)^2 + (z_{s1} - z)^2}} = \frac{-(x_{s1} - x)}{\rho_1 - c\tau}$$

Similarly

$$\frac{\partial \rho_1}{\partial y} = \frac{-(y_{s1} - y)}{\rho_1 - c\tau}$$

$$\frac{\partial \rho_1}{\partial z} = \frac{-(z_{s1} - z)}{\rho_1 - c\tau}$$

By Inspection

$$\frac{\partial \rho_1}{\partial \tau} = c$$

Expressing Total Differential in Matrix Form

Therefore We Can Say

$$\begin{bmatrix}\Delta\rho_1\\ \Delta\rho_2\\ \Delta\rho_3\\ \Delta\rho_4\end{bmatrix}=\begin{bmatrix}\frac{-(x_{s1}-x)}{\rho_1-c\tau} & \frac{-(y_{s1}-y)}{\rho_1-c\tau} & \frac{-(z_{s1}-z)}{\rho_1-c\tau} & c\\ \frac{-(x_{s2}-x)}{\rho_2-c\tau} & \frac{-(y_{s2}-y)}{\rho_2-c\tau} & \frac{-(z_{s2}-z)}{\rho_2-c\tau} & c\\ \frac{-(x_{s3}-x)}{\rho_3-c\tau} & \frac{-(y_{s3}-y)}{\rho_3-c\tau} & \frac{-(z_{s3}-z)}{\rho_3-c\tau} & c\\ \frac{-(x_{s4}-x)}{\rho_4-c\tau} & \frac{-(y_{s4}-y)}{\rho_4-c\tau} & \frac{-(z_{s4}-z)}{z_4-c\tau} & c\end{bmatrix}\begin{bmatrix}\Delta x\\ \Delta y\\ \Delta z\\ \Delta\tau\end{bmatrix}$$

Or

$$\begin{bmatrix}\Delta\rho_1\\ \Delta\rho_2\\ \Delta\rho_3\\ \Delta\rho_4\end{bmatrix}=A\begin{bmatrix}\Delta x\\ \Delta y\\ \Delta z\\ \Delta\tau\end{bmatrix}$$

Where

$$A=\begin{bmatrix}\frac{-(x_{s1}-x)}{\rho_1-c\tau} & \frac{-(y_{s1}-y)}{\rho_1-c\tau} & \frac{-(z_{s1}-z)}{\rho_1-c\tau} & c\\ \frac{-(x_{s2}-x)}{\rho_2-c\tau} & \frac{-(y_{s2}-y)}{\rho_2-c\tau} & \frac{-(z_{s2}-z)}{\rho_2-c\tau} & c\\ \frac{-(x_{s3}-x)}{\rho_3-c\tau} & \frac{-(y_{s3}-y)}{\rho_3-c\tau} & \frac{-(z_{s3}-z)}{\rho_3-c\tau} & c\\ \frac{-(x_{s4}-x)}{\rho_4-c\tau} & \frac{-(y_{s4}-y)}{\rho_4-c\tau} & \frac{-(z_{s4}-z)}{z_4-c\tau} & c\end{bmatrix}$$

Therefore

$$\begin{bmatrix}\Delta x\\ \Delta y\\ \Delta z\\ \Delta\tau\end{bmatrix}=A^{-1}\begin{bmatrix}\Delta\rho_1\\ \Delta\rho_2\\ \Delta\rho_3\\ \Delta\rho_4\end{bmatrix}$$

How Can We Solve For Δx, Δy, Δz and Δτ?

Proposed Iteration Algorithm

Start With 4 Pseudorange Measurements $\rho_1^*,\ \rho_2^*,\ \rho_3^*,\ \rho_4^*$

Choose $\hat{x} = \hat{y} = \hat{z} = \hat{\tau} = 0$

Iterate

$$\begin{bmatrix} \Delta\rho_1 \\ \Delta\rho_2 \\ \Delta\rho_3 \\ \Delta\rho_4 \end{bmatrix} = \begin{bmatrix} \rho_1^* - \hat{\rho}_1 \\ \rho_2^* - \hat{\rho}_2 \\ \rho_3^* - \hat{\rho}_3 \\ \rho_4^* - \hat{\rho}_4 \end{bmatrix}$$

Where

$$\hat{\rho}_1 = \sqrt{(x_{s1} - \hat{x})^2 + (y_{s1} - \hat{y})^2 + (z_{s1} - \hat{z})^2} + c\hat{\tau}$$

$$\hat{\rho}_2 = \sqrt{(x_{s2} - \hat{x})^2 + (y_{s2} - \hat{y})^2 + (z_{s2} - \hat{z})^2} + c\hat{\tau}$$

$$\hat{\rho}_3 = \sqrt{(x_{s3} - \hat{x})^2 + (y_{s3} - \hat{y})^2 + (z_{s3} - \hat{z})^2} + c\hat{\tau}$$

$$\hat{\rho}_4 = \sqrt{(x_{s4} - \hat{x})^2 + (y_{s4} - \hat{y})^2 + (z_{s4} - \hat{z})^2} + c\hat{\tau}$$

$$\begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \Delta\tau \end{bmatrix} = \begin{bmatrix} \dfrac{-(x_{s1} - \hat{x})}{\rho_1^* - c\hat{\tau}} & \dfrac{-(y_{s1} - \hat{y})}{\rho_1^* - c\hat{\tau}} & \dfrac{-(z_{s1} - \hat{z})}{\rho_1^* - c\hat{\tau}} & c \\ \dfrac{-(x_{s2} - \hat{x})}{\rho_2^* - c\hat{\tau}} & \dfrac{-(y_{s2} - \hat{y})}{\rho_2^* - c\hat{\tau}} & \dfrac{-(z_{s2} - \hat{z})}{\rho_2^* - c\hat{\tau}} & c \\ \dfrac{-(x_{s3} - \hat{x})}{\rho_3^* - c\hat{\tau}} & \dfrac{-(y_{s3} - \hat{y})}{\rho_3^* - c\hat{\tau}} & \dfrac{-(z_{s3} - \hat{z})}{\rho_3^* - c\hat{\tau}} & c \\ \dfrac{-(x_{s4} - \hat{x})}{\rho_4^* - c\hat{\tau}} & \dfrac{-(y_{s4} - \hat{y})}{\rho_4^* - c\hat{\tau}} & \dfrac{-(z_{s4} - \hat{z})}{\rho_4^* - c\hat{\tau}} & c \end{bmatrix}^{-1} \begin{bmatrix} \Delta\rho_1 \\ \Delta\rho_2 \\ \Delta\rho_3 \\ \Delta\rho_4 \end{bmatrix}$$

$$\begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ \hat{\tau} \end{bmatrix} = \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \Delta\tau \end{bmatrix} + \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ \hat{\tau} \end{bmatrix}$$

Sample Satellite Locations From Reference*

Satellite	x_s (m)	y_s (m)	z_s (m)
SV 01	22,808,161	-12,005,867	-6,609,527
SV 02	21,141,180	-2,355,056	-15,985,716
SV 08	20,438,959	-4,238,967	16,502,090
SV 14	18,432,296	-18,613,383	-4,672,401
SV 17	21,772,118	13,773,270	6,656,636
SV 23	15,561,524	3,469,099	-21,303,596
SV 24	13,773,317	15,929,331	-16,266,254

*Parkinson, et. al., "Global Positioning System Theory and Applications Volume 1," AIAA Progress in Aeronautics and Astronautics, Washington, DC, pp. 415-417

Simulation-1

```
GLOBAL DEFINE
                INCLUDE 'quickdraw.inc'
END
IMPLICIT REAL*8 (A-H,O-Z)
REAL*8 A(4,4),AINV(4,4),B(4,1),ANS(4,1)
OPEN(1,STATUS='UNKNOWN',FILE='DATFIL')
TAU=.000001D0
C=300000000.D0
SIGNOISE=100.D0
RADEARTH=2.0926E7/3.28D0
RADDEG=57.3D0
XLATFDEG=0.D0
XLONGFDEG=0.D0
X0=RADEARTH*COS(XLATFDEG/RADDEG)*COS(XLONGFDEG/RADDEG)
Y0=RADEARTH*COS(XLATFDEG/RADDEG)*SIN(XLONGFDEG/RADDEG)
Z0=RADEARTH*SIN(XLATFDEG/RADDEG)
XH=0.D0
YH=0.D0
ZH=0.D0
TAUH=0.D0
XS1=22808161.D0
YS1=-12005867.D0
ZS1=-6609527.D0
XS2=21141180.D0
YS2=-2355056.D0
ZS2=-15985716.D0
XS3=20438959.D0
YS3=-4238967.D0
ZS3=16502090.D0
XS4=18432296.D0
YS4=-18613383.D0
ZS4=-4672401.D0
R1=SQRT((XS1-X0)**2+(YS1-Y0)**2+(ZS1-Z0)**2)
R2=SQRT((XS2-X0)**2+(YS2-Y0)**2+(ZS2-Z0)**2)
R3=SQRT((XS3-X0)**2+(YS3-Y0)**2+(ZS3-Z0)**2)
R4=SQRT((XS4-X0)**2+(YS4-Y0)**2+(ZS4-Z0)**2)
```

Timing Error

Receiver Location

Initially We Guess Receiver is at Center of Earth!

Location of 4 GPS Satellites

True Range From Each Satellite To Receiver

Simulation-2

```
CALL GAUSS(R1NOISE,SIGNOISE)
CALL GAUSS(R2NOISE,SIGNOISE)
CALL GAUSS(R3NOISE,SIGNOISE)
CALL GAUSS(R4NOISE,SIGNOISE)
BIAS1=0.D0
BIAS2=0.D0
BIAS3=0.D0
BIAS4=0.D0
R1NOISE=0.D0
R2NOISE=0.D0
R3NOISE=0.D0
R4NOISE=0.D0
R1S=R1+R1NOISE+C*TAU+BIAS1
R2S=R2+R2NOISE+C*TAU+BIAS2
R3S=R3+R3NOISE+C*TAU+BIAS3
R4S=R4+R4NOISE+C*TAU+BIAS4
DO 10 I=1,30
R1H=SQRT((XS1-XH)**2+(YS1-YH)**2+(ZS1-ZH)**2)+C*TAUH
R2H=SQRT((XS2-XH)**2+(YS2-YH)**2+(ZS2-ZH)**2)+C*TAUH
R3H=SQRT((XS3-XH)**2+(YS3-YH)**2+(ZS3-ZH)**2)+C*TAUH
R4H=SQRT((XS4-XH)**2+(YS4-YH)**2+(ZS4-ZH)**2)+C*TAUH
DELR1=R1S-R1H
DELR2=R2S-R2H
DELR3=R3S-R3H
DELR4=R4S-R4H
DELR=SQRT(DELR1**2+DELR2**2+DELR3**2+DELR4**2)
B(1,1)=DELR1
B(2,1)=DELR2
B(3,1)=DELR3
B(4,1)=DELR4
```

Pseudorange Measurement With Noise, Bias and Timing Errors

Estimated Pseudorange

Difference Between Pseudorange Measurement And Estimate

Simulation-3

```
A(1,1)=-(XS1-XH)/(R1H-C*TAUH)
A(1,2)=-(YS1-YH)/(R1H-C*TAUH)
A(1,3)=-(ZS1-ZH)/(R1H-C*TAUH)
A(1,4)=C
A(2,1)=-(XS2-XH)/(R2H-C*TAUH)
A(2,2)=-(YS2-YH)/(R2H-C*TAUH)
A(2,3)=-(ZS2-ZH)/(R2H-C*TAUH)
A(2,4)=C
A(3,1)=-(XS3-XH)/(R3H-C*TAUH)
A(3,2)=-(YS3-YH)/(R3H-C*TAUH)
A(3,3)=-(ZS3-XH)/(R3H-C*TAUH)
A(3,4)=C
A(4,1)=-(XS4-XH)/(R4H-C*TAUH)
A(4,2)=-(YS4-YH)/(R4H-C*TAUH)
A(4,3)=-(ZS4-XH)/(R4H-C*TAUH)
A(4,4)=C
CALL MTINV(A,4,AINV)
CALL MATMUL(AINV,4,4,B,4,1,ANS)
DELX=ANS(1,1)
DELY=ANS(2,1)
DELZ=ANS(3,1)
DELTAU=ANS(4,1)
XH=DELX+XH
YH=DELY+YH
ZH=DELZ+ZH
TAUH=DELTAU+TAUH
XERR=ABS(X0-XH)
YERR=ABS(Y0-YH)
ZERR=ABS(Z0-ZH)
TAUERR=ABS(TAU-TAUH)
WRITE(9,*)I,XERR,YERR,ZERR,TAUERR
WRITE(1,*)I,XERR,YERR,ZERR,TAUERR
IF(DELR<.0001)EXIT
10 CONTINUE
CLOSE(1)
PAUSE
END
```

Calculate Delta Errors

Update State Estimates

Compute Errors in Estimates

Exit Iteration Loop When Error Criteria Satisfied

Iteration Loop

Simulation-4 (Matrix Inverse)

```
                SUBROUTINE MTINV(A,N,AINV)
                IMPLICIT REAL*8                 (A-H,O-Z)
                PARAMETER    (ZERO=0.,ONE=1.)
                DIMENSION A(N,N),AINV(N,N),PIVOT(15),IPIVOT(15),INDEX(15,2)

C  MATRIX INVERSION - GAUSS-JORDAN ELIMINATION METHOD
C
C  INPUT:
C               A(N,N)          = "N" BY "N" MATRIX TO BE INVERTED.
C               N               = ORDER (NUMBER OF ROWS AND COLUMNS) OF MATRIX "A".

C  OUTPUT:
C               AINV(N,N) = INVERSE OF MATRIX "A".

C  TEST FOR ERRORS
                IF(N .LE. 1) THEN
                  WRITE(3,*) 'MTINV CALLED WITH N ≤ 1'
                  STOP
                ELSE IF(N .GT. 15) THEN
                  WRITE(3,*) 'MTINV CALLED WITH N > 15'
                  STOP
                END IF
                DO (I=1,N)
                  DO (J=1,N)
                                AINV(I,J) = A(I,J)
                  END DO
                END DO
C
C  INITIALIZATION
C
                DO (J=1,N)
                  IPIVOT(J) = 0
                END DO

                DO (I=1,N)
```

Simulation-5 (Matrix Inverse)

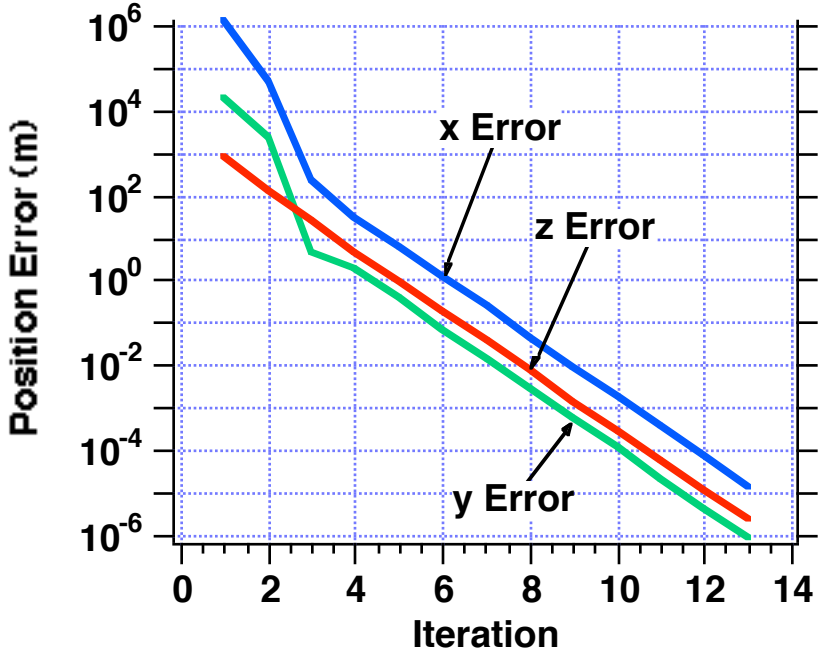
```
C  SEARCH FOR PIVOT ELEMENT
C
                    BIG = ZERO
                    DO (J=1,N)
                     IF(IPIVOT(J) .NE. 1) THEN
                                DO (K=1,N)
                                  IF(IPIVOT(K) .LT. 1) THEN
                                              IF(ABS(AINV(J,K)) .GE. ABS(BIG)) THEN
                                  IROW   = J
                                  ICOLUM = K
                                                 BIG   = AINV(J,K)
                                              END IF
                                  ELSE IF(IPIVOT(K) .GT. 1) THEN
                                  GO TO 999
                                  END IF
                                END DO
                     END IF
                    END DO
                    IPIVOT(ICOLUM) = IPIVOT(ICOLUM) + 1
C  INTERCHANGE ROWS TO PUT PIVOT ELEMENT ON DIAGONAL
C
                    IF(IROW .NE. ICOLUM) THEN
                                DO (L=1,N)
                          SWAP       = AINV(IROW,L)
                          AINV(IROW,L)   = AINV(ICOLUM,L)
                          AINV(ICOLUM,L) = SWAP
                                END DO
                    END IF
                    INDEX(I,1)     = IROW
                    INDEX(I,2)     = ICOLUM
                    PIVOT(I)       = AINV(ICOLUM,ICOLUM)
C  TEST FOR SINGULARITY
                    IF( PIVOT(I) .EQ. ZERO) GO TO 999
C  DIVIDE PIVOT ROW BY PIVOT ELEMENT
C
                    AINV(ICOLUM,ICOLUM) = ONE
                    DO (L=1,N)
                                AINV(ICOLUM,L) = AINV(ICOLUM,L)/PIVOT(I)
                    END DO
```

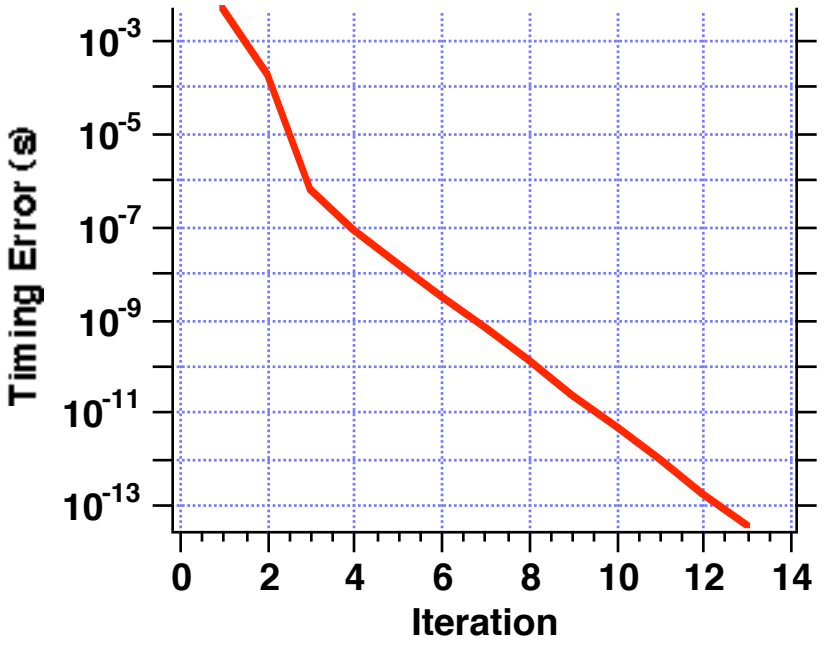
Simulation-6 (Matrix Inverse)

```
C
C  REDUCE NON-PIVOT ROWS
C
                DO (L1=1,N)
                   IF(L1 .NE. ICOLUM) THEN
                      T                          = AINV(L1,ICOLUM)
                      AINV(L1,ICOLUM) = ZERO
                      DO (L=1,N)
                                                 AINV(L1,L)  = AINV(L1,L) - AINV(ICOLUM,L)*T
                         END DO
                   END IF
                END DO
             END DO
C
C  INTERCHANGE COLUMNS
C
             DO (I=1,N)
               L = N - I + 1
               IF(INDEX(L,1) .NE. INDEX(L,2)) THEN
                         JROW   = INDEX(L,1)
                         JCOLUM = INDEX(L,2)
                         DO (K=1,N)
                            SWAP          = AINV(K,JROW)
                      AINV(K,JROW)   = AINV(K,JCOLUM)
                      AINV(K,JCOLUM) = SWAP
                         END DO
               END IF
             END DO

 999         CONTINUE
             RETURN
             END
```

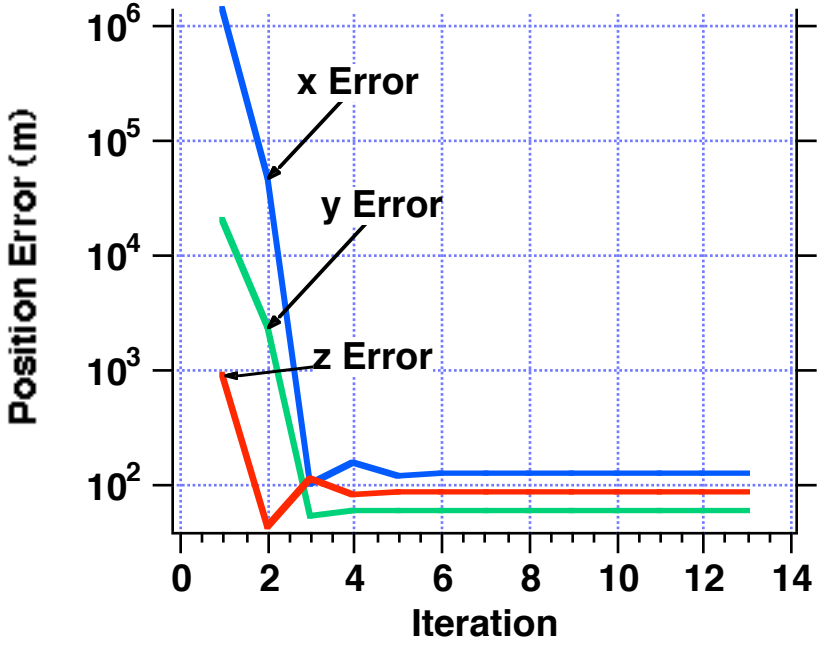
In Absence of Noise and Biases, Errors Converge to Zero Very Quickly

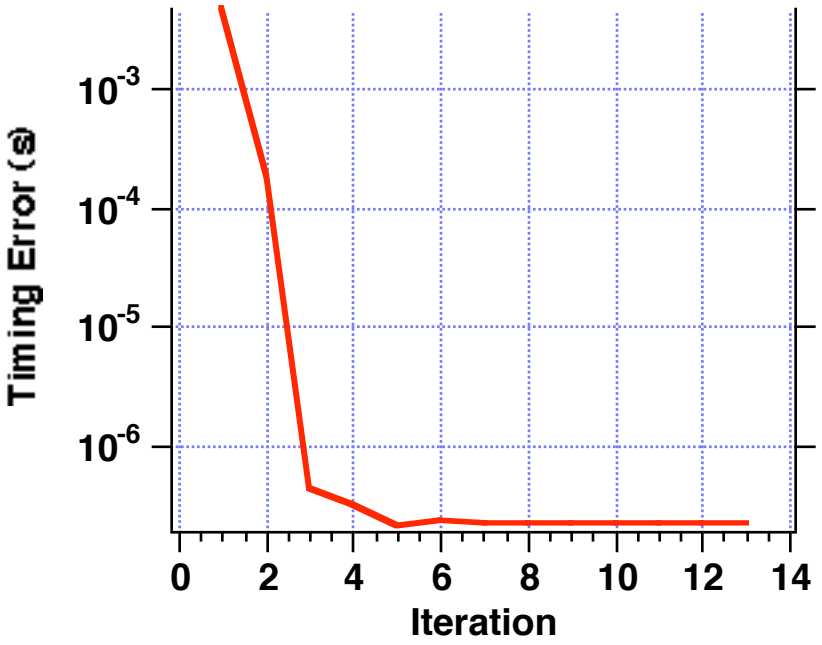




If pseudorange measurements are exact we have an iterative method which gives us the receiver location and timing errors without having to derive closed-form solutions

With 100 m of Measurement Noise (No Biases) Errors Converge to Less Than 150 m Very Quickly

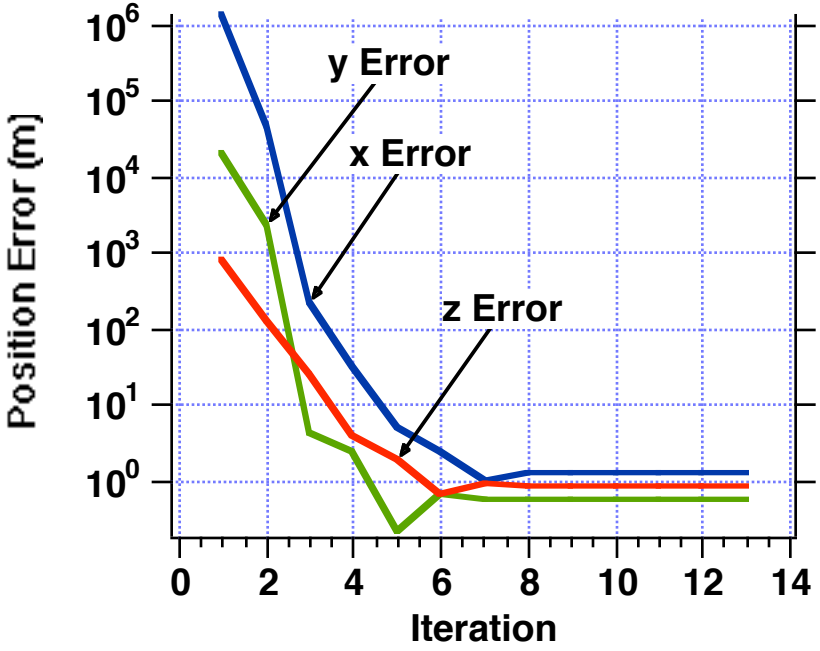


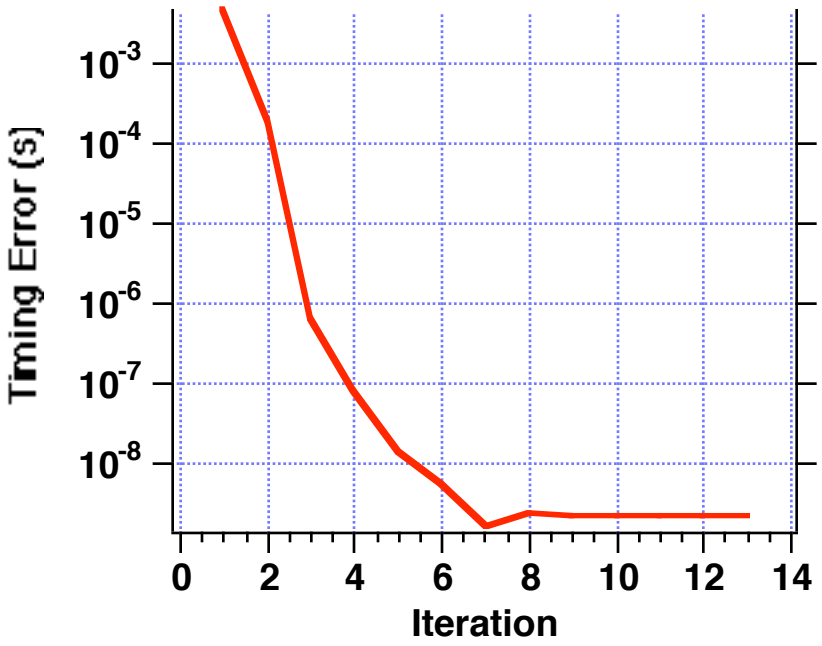


We have already shown that an Extended Kalman filter needs to know the initial receiver location to within a few thousand feet.

The iterative method provides the required accuracy to initialize the Extended Kalman filter

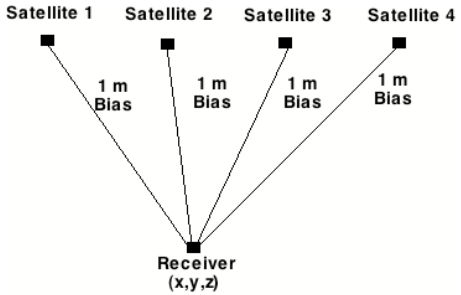
With 1 m of Measurement Noise (No Biases) Errors Converge to Less Than 2 m Very Quickly

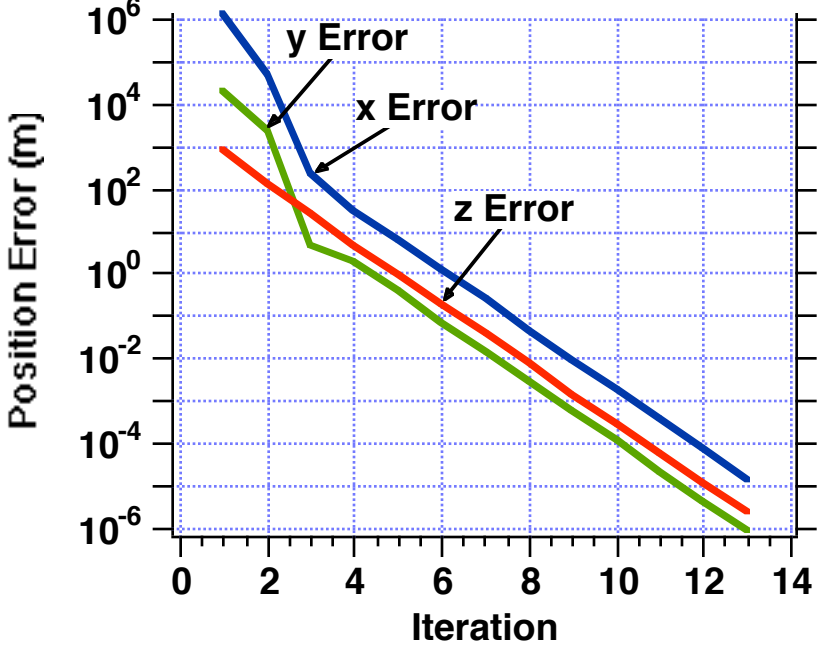


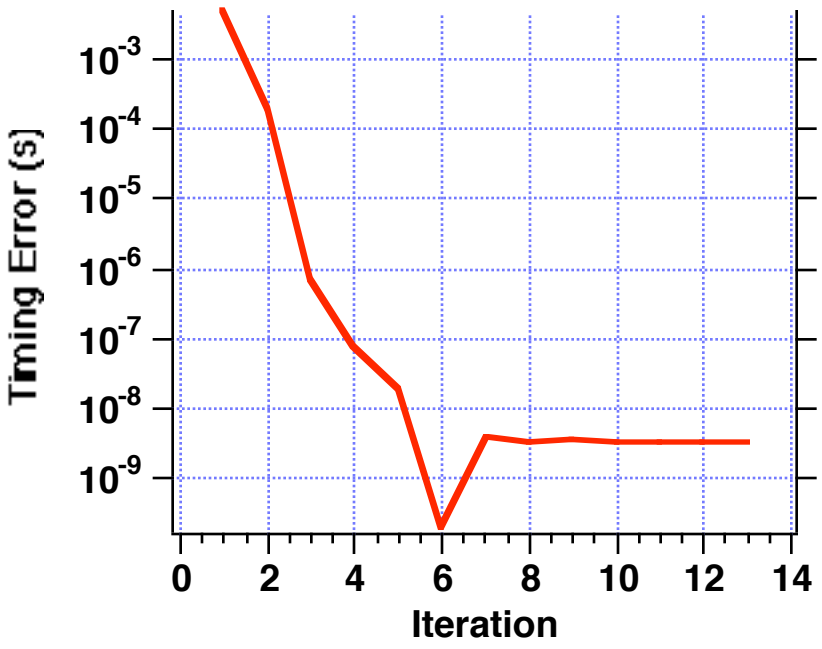


If measurement noise is 1 m then filtering may not be required

In Absence of Noise and 1 m Bias Error on Each Range Measurement Errors Converge to Zero Very Quickly

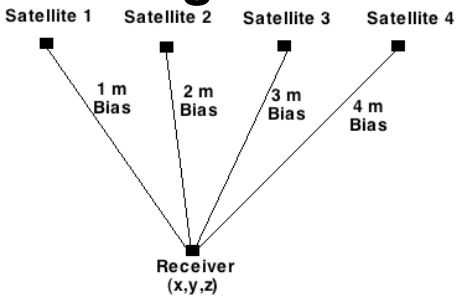


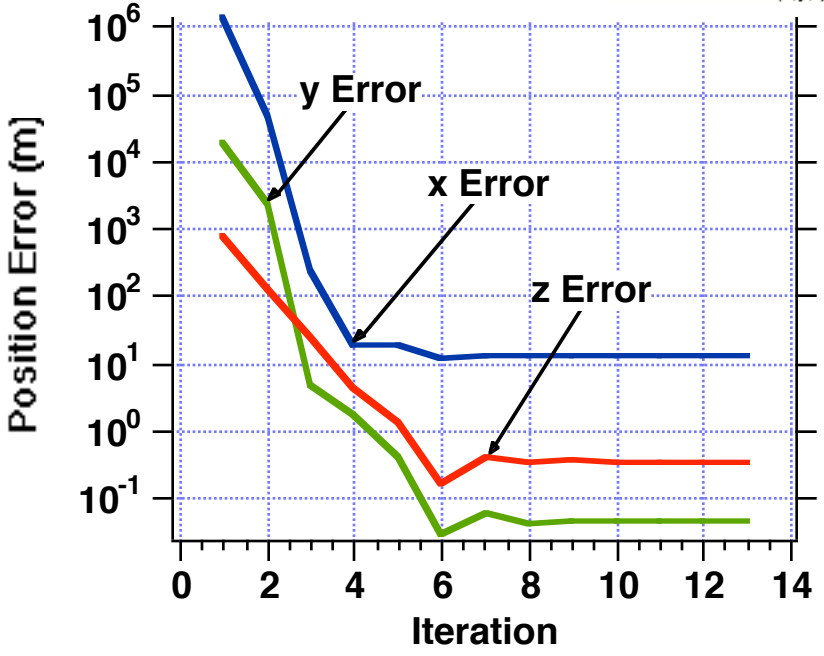


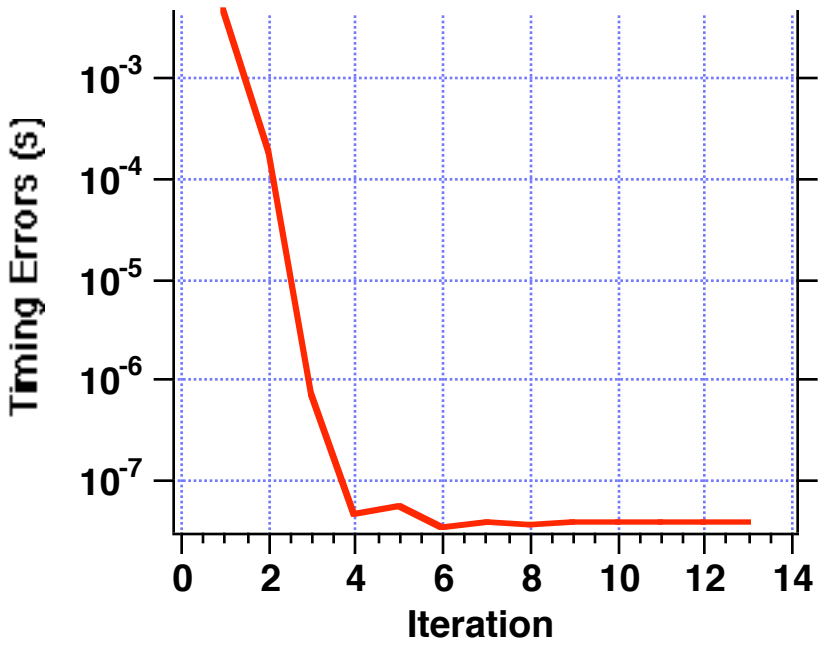


If pseudorange measurements have same bias we have an iterative method which gives us the exact receiver location and small timing errors

In Absence of Noise and Unequal Bias Errors (1,2,3,4) on Each Range Measurement Largest Error Converges to 13 m







Unequal bias errors on pseudorange measurements can lead to large position and timing errors

How Can We Use More Than 4 Satellites?

With 4 Satellites We Solved

$$\Delta pos = A^{-1}\Delta range$$

With 4 Pseudorange Measurements A Matrix Was 4 by 4 and Invertible

With 6 Pseudorange Measurements A Matrix is 6 by 4 and Not Invertible We Can Use Pseudo Inverse

$$\Delta range = A\Delta pos$$

Multiply Each Side by A^T

$$A^T\Delta range = A^T A\Delta pos$$

Multiply Each Side by $(A^TA)^{-1}$ Where A^TA is a 4 by 4 Matrix and is Invertible

$$\left(A^T A\right)^{-1} A^T\Delta range = \left(A^T A\right)^{-1}(A^T A)\Delta pos = \Delta pos$$

Or

$$\Delta pos = \left(A^T A\right)^{-1} A^T\Delta range$$

Sample Satellite Locations From Reference*

Satellite	x_s (m)	y_s (m)	z_s (m)
SV 01	22,808,161	-12,005,867	-6,609,527
SV 02	21,141,180	-2,355,056	-15,985,716
SV 08	20,438,959	-4,238,967	16,502,090
SV 14	18,432,296	-18,613,383	-4,672,401
SV 17	21,772,118	13,773,270	6,656,636
SV 23	15,561,524	3,469,099	-21,303,596
SV 24	13,773,317	15,929,331	-16,266,254

*Parkinson, et. al., "Global Positioning System Theory and Applications Volume 1," AIAA Progress in Aeronautics and Astronautics, Washington, DC,1996, pp. 415-417

6 Satellite Simulation - 1

```
GLOBAL DEFINE
                INCLUDE 'quickdraw.inc'
END
IMPLICIT REAL*8 (A-H,O-Z)
REAL*8 A(6,4),AINV(4,6),B(6,1),ANS(4,1),AT(4,6),CMAT(6,6)
REAL*8 CINV(4,4)
OPEN(1,STATUS='UNKNOWN',FILE='DATFIL')
TAU=.000001D0
C=300000000.D0
SIGNOISE=100.D0
RADEARTH=2.0926E7/3.28D0
RADDEG=57.3D0
XLATFDEG=0.D0
XLONGFDEG=0.D0
X0=RADEARTH*COS(XLATFDEG/RADDEG)*COS(XLONGFDEG/RADDEG)
Y0=RADEARTH*COS(XLATFDEG/RADDEG)*SIN(XLONGFDEG/RADDEG)
Z0=RADEARTH*SIN(XLATFDEG/RADDEG)
XH=0.D0
YH=0.D0
ZH=0.D0
TAUH=0.D0
XS1=22808161.D0
YS1=-12005867.D0
ZS1=-6609527.D0
XS2=21141180.D0
YS2=-2355056.D0
ZS2=-15985716.D0
XS3=20438959.D0
YS3=-4238967.D0
ZS3=16502090.D0
XS4=18432296.D0
YS4=-18613383.D0
ZS4=-4672401.D0
XS5=21772118.D0
YS5=13773270.D0
ZS5=6656636.D0
XS6=15561524.D0
YS6=3469099.D0
ZS6=-21303596.D0
```

Timing Error

Receiver Location

Algorithm States Initialized to Zero

Location of 6 GPS Satellites

6 Satellite Simulation - 2

```
R1=SQRT((XS1-X0)**2+(YS1-Y0)**2+(ZS1-Z0)**2)
R2=SQRT((XS2-X0)**2+(YS2-Y0)**2+(ZS2-Z0)**2)
R3=SQRT((XS3-X0)**2+(YS3-Y0)**2+(ZS3-Z0)**2)
R4=SQRT((XS4-X0)**2+(YS4-Y0)**2+(ZS4-Z0)**2)
R5=SQRT((XS5-X0)**2+(YS5-Y0)**2+(ZS5-Z0)**2)
R6=SQRT((XS6-X0)**2+(YS6-Y0)**2+(ZS6-Z0)**2)
CALL GAUSS(R1NOISE,SIGNOISE)
CALL GAUSS(R2NOISE,SIGNOISE)
CALL GAUSS(R3NOISE,SIGNOISE)
CALL GAUSS(R4NOISE,SIGNOISE)
CALL GAUSS(R5NOISE,SIGNOISE)
CALL GAUSS(R6NOISE,SIGNOISE)
R1NOISE=0.D0
R2NOISE=0.D0
R3NOISE=0.D0
R4NOISE=0.D0
R5NOISE=0.D0
R6NOISE=0.D0
R1S=R1+R1NOISE+C*TAU
R2S=R2+R2NOISE+C*TAU
R3S=R3+R3NOISE+C*TAU
R4S=R4+R4NOISE+C*TAU
R5S=R5+R5NOISE+C*TAU
R6S=R6+R6NOISE+C*TAU
```

Pseudorange Measurement With Noise and Timing Error

```
DO 10 I=1,30
R1H=SQRT((XS1-XH)**2+(YS1-YH)**2+(ZS1-ZH)**2)+C*TAUH
R2H=SQRT((XS2-XH)**2+(YS2-YH)**2+(ZS2-ZH)**2)+C*TAUH
R3H=SQRT((XS3-XH)**2+(YS3-YH)**2+(ZS3-ZH)**2)+C*TAUH
R4H=SQRT((XS4-XH)**2+(YS4-YH)**2+(ZS4-ZH)**2)+C*TAUH
R5H=SQRT((XS5-XH)**2+(YS5-YH)**2+(ZS5-ZH)**2)+C*TAUH
R6H=SQRT((XS6-XH)**2+(YS6-YH)**2+(ZS6-ZH)**2)+C*TAUH
```

Estimated Pseudorange

```
DELR1=R1S-R1H
DELR2=R2S-R2H
DELR3=R3S-R3H
DELR4=R4S-R4H
DELR5=R5S-R5H
DELR6=R6S-R6H
DELR=SQRT(DELR1**2+DELR2**2+DELR3**2+DELR4**2+DELR5**2+DELR6**2)
```

Difference Between Pseudorange Measurement And Estimate

6 Satellite Simulation - 3

```
B(1,1)=DELR1
B(2,1)=DELR2
B(3,1)=DELR3
B(4,1)=DELR4
B(5,1)=DELR5
B(6,1)=DELR6
A(1,1)=-(XS1-XH)/(R1H-C*TAUH)
A(1,2)=-(YS1-YH)/(R1H-C*TAUH)
A(1,3)=-(ZS1-ZH)/(R1H-C*TAUH)
A(1,4)=C
A(2,1)=-(XS2-XH)/(R2H-C*TAUH)
A(2,2)=-(YS2-YH)/(R2H-C*TAUH)
A(2,3)=-(ZS2-ZH)/(R2H-C*TAUH)
A(2,4)=C
A(3,1)=-(XS3-XH)/(R3H-C*TAUH)
A(3,2)=-(YS3-YH)/(R3H-C*TAUH)
A(3,3)=-(ZS3-XH)/(R3H-C*TAUH)
A(3,4)=C
A(4,1)=-(XS4-XH)/(R4H-C*TAUH)
A(4,2)=-(YS4-YH)/(R4H-C*TAUH)
A(4,3)=-(ZS4-XH)/(R4H-C*TAUH)
A(4,4)=C
A(5,1)=-(XS5-XH)/(R5H-C*TAUH)
A(5,2)=-(YS5-YH)/(R5H-C*TAUH)
A(5,3)=-(ZS5-XH)/(R5H-C*TAUH)
A(5,4)=C
A(6,1)=-(XS6-XH)/(R6H-C*TAUH)
A(6,2)=-(YS6-YH)/(R6H-C*TAUH)
A(6,3)=-(ZS6-XH)/(R6H-C*TAUH)
A(6,4)=C
CALL MATTRN(A,6,4,AT)
CALL MATMUL(AT,4,6,A,6,4,CMAT)
CALL MTINV(CMAT,4,CINV)
CALL MATMUL(CINV,4,4,AT,4,6,AINV)
CALL MATMUL(AINV,4,6,B,6,1,ANS)
```

Pseudo Inverse

Calculate Delta Errors

Iteration Loop

6 Satellite Simulation - 4

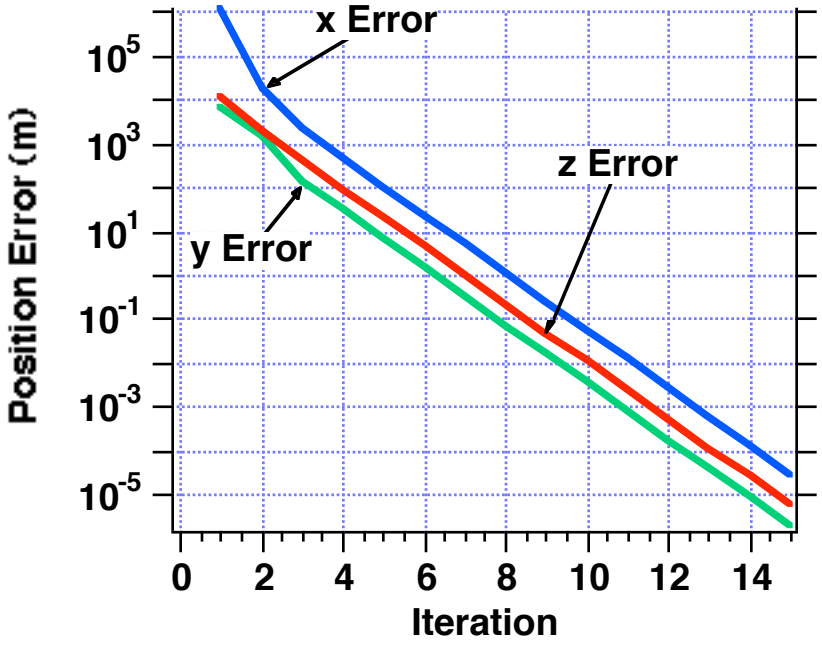
```
      DELX=ANS(1,1)
      DELY=ANS(2,1)
      DELZ=ANS(3,1)
      DELTAU=ANS(4,1)
      XH=DELX+XH
      YH=DELY+YH
      ZH=DELZ+ZH
      TAUH=DELTAU+TAUH
      XERR=ABS(X0-XH)
      YERR=ABS(Y0-YH)
      ZERR=ABS(Z0-ZH)
      TAUERR=ABS(TAU-TAUH)
      WRITE(9,*)I,XERR,YERR,ZERR,TAUERR
      WRITE(1,*)I,XERR,YERR,ZERR,TAUERR
      IF(DELR<.0001)EXIT
10    CONTINUE
      CLOSE(1)
      PAUSE
      END
```

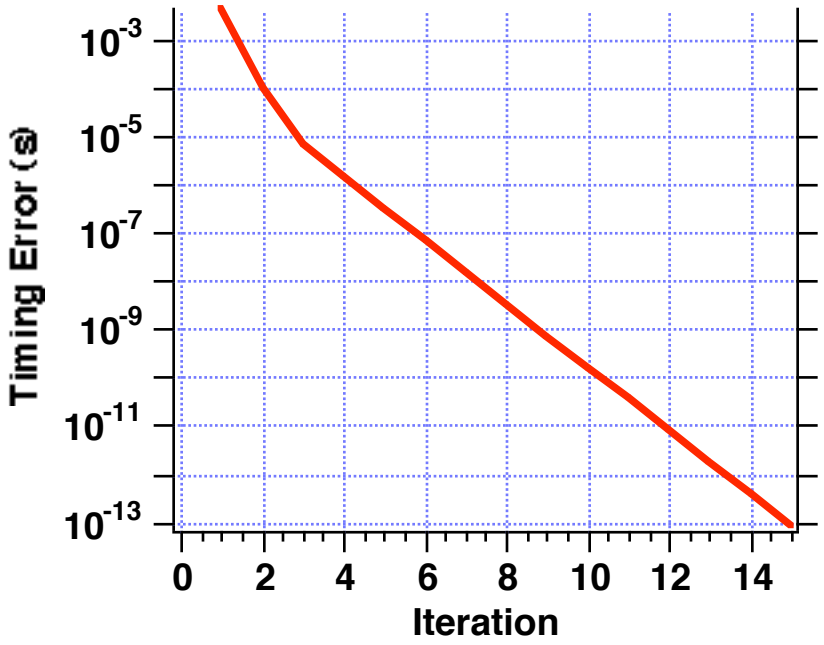
Update State Estimates

Compute Errors in Estimates

Exit Iteration Loop When Error Criteria Satisfied

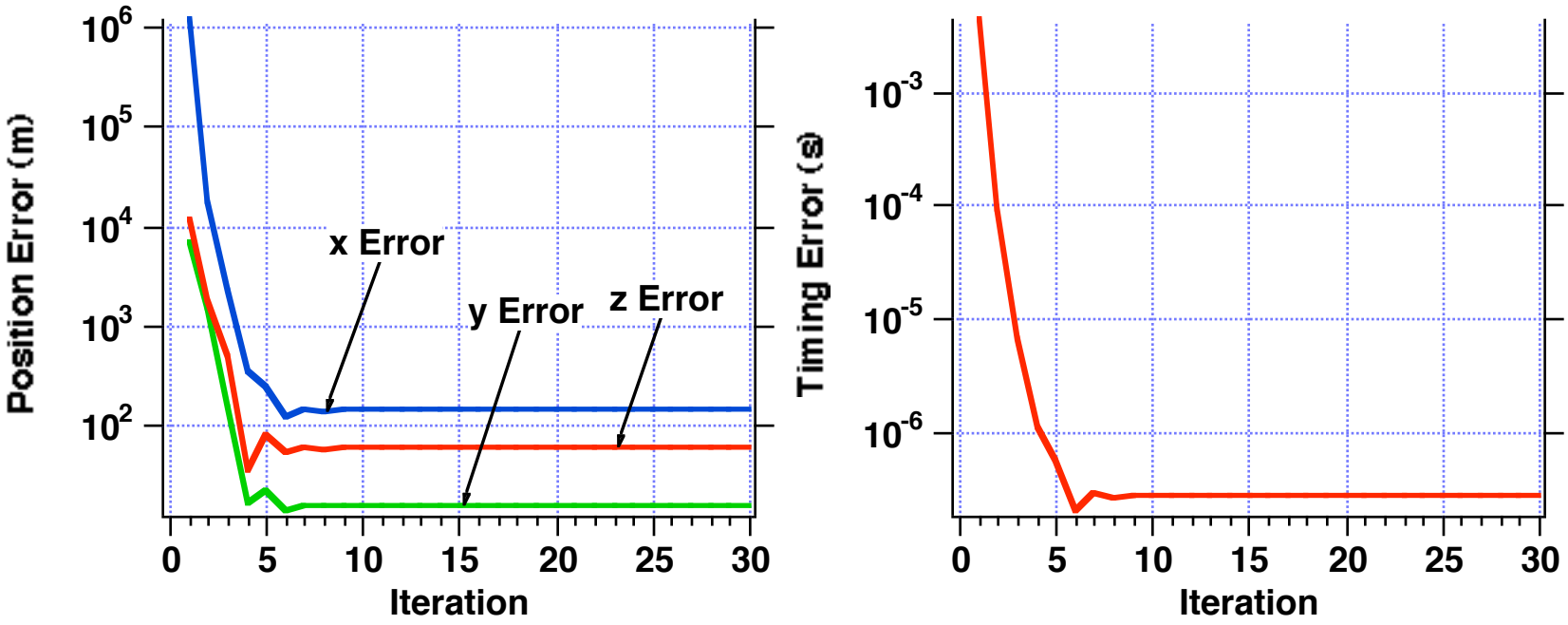
In Absence of Noise and Biases Errors Converge to Zero Very Quickly With 6 Satellites





If pseudorange measurements are exact we have an iterative method which gives us the receiver location and timing errors without having to derive closed-form solutions

With 100 m of Measurement Noise (No Biases) Errors Converge to Less Than 150 m Very Quickly With 6 Satellites



We have already shown that an Extended Kalman filter needs to know the initial receiver location to within a few thousand feet.

The iterative method provides the required accuracy to initialize the Extended Kalman filter