Kalman Filtering

Kevin Mead

May 23, 2014

1 Introduction

The task is to compute the coordinates and velocities of a moving vehicle. The calculations were done with a constant velocity kinematic model. Measurement values in the Excel document, *Measurement.xls*, give us the position and speed of the vehicle measured in 2 second intervals. Other values given are:

- PSD (power spectral density) of the random acceleration: 0.01 m^2s^3
- Standard deviation of measured coordinates (both components): 3 m
- Standard deviation of measured abs. velocity: 0.5 m/s
- Standard deviation of initial velocity (both components): 3 m/s
- Standard deviation of initial coordinates (both components): 10 m

2 Methodology

Matlab was used to solve for the following parameters and all of the following equations were taken from the technical report: Kalman Filtering.[1] The steps and equations from the Kalman filtering algorithm are documented in the Matlab code. We start off with the matrix differential equation,

$$\dot{\mathbf{x}}(t) = \mathbf{F}(t)x(t) + \mathbf{G}(t)\mathbf{u}(t) \tag{1}$$

where $\mathbf{x} = [\mathrm{e}\ \mathrm{n}\ \mathrm{v}_e\ \mathrm{v}_n]^{\mathrm{T}}$, the east and north positions and their velocity components. $\mathbf{F}$, $\mathbf{G}$, and $\mathbf{u}$ are given by,

$$\mathbf{F} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \mathbf{G} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \mathbf{u} = \begin{bmatrix} \omega_{ae} \\ \omega_{an} \end{bmatrix}$$

We do not know $\mathbf{u}$ because it represents white noise. The discrete solution to Equation (1) is

$$\mathbf{x}_k = \mathbf{T}_{k-1,k}\mathbf{x}_{k-1} + \mathbf{w}_{k-1,k}$$

where values of k represent discretization of time. The transition matrix, $\mathbf{T}_{k-1,k}$, can be approximated by,

$$\mathbf{T}_{k-1,k} = \mathbf{I} + \mathbf{F}_k \Delta t = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The covariance of $\mathbf{w}_k$, $\mathbf{Q}_k$, is evaluated by use of the power spectral density (PSD) of the random acceleration, which are initially given. We can now represent the covariance of $\mathbf{w}_k$ as,

$$\mathbf{Q}_k = \mathbf{Q}_G \Delta t + (\mathbf{F}\mathbf{Q}_G + \mathbf{Q}_G \mathbf{F}^{\mathrm{T}}) \frac{\Delta t^2}{2} + \mathbf{F}\mathbf{Q}_G \mathbf{F}^{\mathrm{T}} \frac{\Delta t^3}{3}$$

where

$$\mathbf{Q}_G = \mathbf{G}\mathbf{Q}\mathbf{G}^{\mathrm{T}} \quad \mathbf{Q} = \begin{bmatrix} q_{ae} & 0 \\ 0 & q_{an} \end{bmatrix}$$

2.1 Kalman Filter

The main steps of the discrete Kalman Filter algorithm are:

1. Initialization:

$$\mathbf{x}_0, \quad \mathbf{Q}_{x0} = \mathrm{var}[x_0]$$

2. Time propagation

$$\mathbf{x}_k^- = \mathbf{T}_{k-1,k} \mathbf{x}_{k-1}, \quad \mathbf{Q}_{x,k}^- = \mathbf{T}_{k-1,k} \mathbf{Q}_{x,k-1} \mathbf{T}_{k-1,k}^{\mathrm{T}} + \mathbf{Q}_k$$

3. Gain calculation:

$$\mathbf{K}_k = \mathbf{Q}_{x,k}^- \mathbf{H}_k^{\mathrm{T}} [\mathbf{R}_k + \mathbf{H}_k \mathbf{Q}_{x,k}^- \mathbf{H}_k^{\mathrm{T}}]^{-1}$$

4. Measurement update

$$\mathbf{x}_k = \mathbf{x}_k^- + \mathbf{K}_k [\tilde{L}_k - \mathbf{h}_k(x_k^-)]$$

5. Covariance update

$$\mathbf{Q}_{x,k} = [\mathbf{I} - \mathbf{K}_k \mathbf{H}_k] \mathbf{Q}_{x,k}^-$$

2.2 Smoothing

After the Kalman Filter algorithm, we then smooth the results using the smoothed estimations for previous epochs,

$$\hat{\mathbf{x}}_k = \mathbf{x}_k + \mathbf{D}_k [\hat{\mathbf{x}}_{k+1} - \mathbf{x}_{k+1}^-]$$

$$\mathbf{D}_k = \mathbf{Q}_{x,k}\mathbf{T}_{k-1,k}^T(Q_{x,k+1}^-)^{-1}$$

with a covariance matrix of,

$$\hat{\mathbf{Q}}_{x,k} = \mathbf{Q}_{x,k} + \mathbf{D}_k[\hat{\mathbf{Q}}_{x,k+1} - \mathbf{Q}_{x,k+1}^-]\mathbf{D}_k^\mathrm{T}$$

3 Results

A comparison of the filtered and smoothed coordinates with the true measurements are shown in Figures 1 and 2. Table 1 and 2 are the coordinates after filtering and smoothing. Figures 3 and 4 show the difference between the true measurements and the filtered and smoothed results.

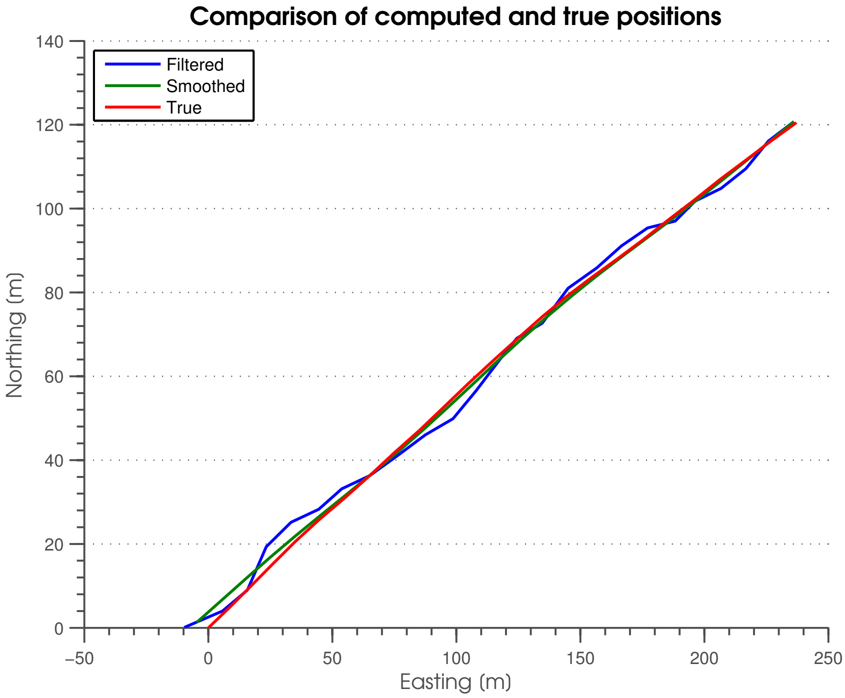

Figure 1: Comparison of filtered, smoothed, original, and true coordinates.

4 Analysis and Discussion

- What happens if we change the values of

  a) PSD?

  The value of the PSD changes the covariance matrix, $\mathbf{Q}_k$, which then affects the random accelerations of the vehicle measurements. Increasing the PSD would increase the standard deviations of our filter and smoothing results.

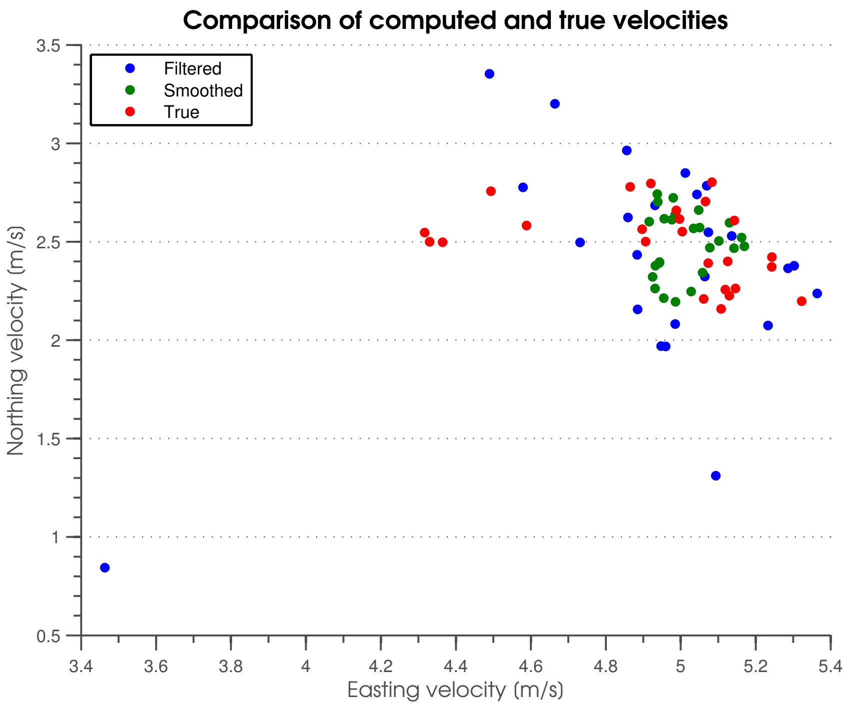

Figure 2: Comparison of filtered, smoothed, and true velocities.

b) Standard deviations of measurements?

Changing the standard deviations of the measurements increases the standard deviations of the filtered results but doesn't affect the smoothing results.

c) Standard deviations of initial state variables?

Increasing the standard deviations of the initial state variables only affects our ability to calculate the precise location of the car at that time. The more measurements we have, the less important the initial state becomes.

- Can we draw any conclusion/implications from the results?

  We can conclude that the smoothed coordinates are a better approximation to the true values than the filtered coordinates.

- Are results reliable and accurate?

  I believe the results are reliable and accurate because the standard deviations dropped when the coordinates were smoothed and the state vector was a better approximation of the true values. Figures 3 and 4 shows that the smoothed values are closer to $(0, 0)$ than the filtered values.

References

[1] M. Horemuž, "Kalman filtering," Royal Institute of Technology, Tech. Rep., 2014.

Table of Contents

Kalman Filter

```
clear all;clc;close all
addpath(['/Users/kevin/SkyDrive/KTH Work/',...
    'Period 4 2014/GNSS/Labs/L4 ',...
    '- Kalman filtering/']);
```

Import numbers from excel

Measured values from sheet 1

```
data.s1 = xlsread('Measurement.xlsx');
meas.time = data.s1(1:25,1);
meas.east = data.s1(1:25,2);
meas.north = data.s1(1:25,3);
meas.speed = data.s1(1:25,4);
% True values from sheet 2
data.s2 = xlsread('Measurement.xlsx','True values');
true.time = data.s2(1:25,1);
true.east = data.s2(1:25,2);
true.north = data.s2(1:25,3);
true.vel_east = data.s2(1:25,4);
true.vel_north = data.s2(1:25,5);
```

A priori statistics

```
PSD = 0.01; % - PSD (power spectral density) of the
% random acceleration
meas.sd_coord = 3; % m - Standard error of measured coordinates
meas.sd_abs_vel = 0.5; % m/s - Standard error of measured
% abs. velocity
sd_ini_vel = 3; % m/s - Standard error of initial velocity
sd_ini_coord = 10; % m - Standard error of initial coordinates
ve = 3.53; % m/s
vn = 0.86; % m/s
dt = 2; % time difference -> 2 sec between measurements
```

State variables and Equations

```
xk = [meas.east(1) meas.north(1) ve vn];
```

```
xk = padarray(xk,24,0,'post');
xk = xk';
% Equation 4
F = zeros(4);
F(1,3) = 1;
F(2,4) = 1;
G = zeros(4,2);
G(3,1) = 1;
G(4,2) = 1;
% Equation 5
Tk = eye(length(F)) + dt * F;
% Equation 9
Q = [ PSD 0 ; 0 PSD];
% Equation 11
QG = G*Q*G';
% Equation 12
Qk = QG * dt + (F*QG + QG*F')*dt^2/2 + F*QG*F'*dt^3/3;
% Equation 15
% covariance matrix of initial state
Qx(:,:,1) = diag([sd_ini_coord^2 ...
    sd_ini_coord^2 sd_ini_vel^2 sd_ini_vel^2]);
Rk = diag([meas.sd_coord meas.sd_coord meas.sd_abs_vel]);
Qxm_predicted = zeros(4,4,25);
```

FOR LOOP

```
for i=1:25
    % Equation 16 Time propagation
    xkm_predicted(:,i) = Tk * xk(:,i);
    %     Qx = cov(xk(:,i));
    Qxm_predicted(:,:,i) = Tk * Qx(:,:,i) * Tk' + Qk;
    vm_predicted = sqrt(xkm_predicted(3,i)^2 +  ...
        xkm_predicted(4,i)^2); % should be equal to speed_meas(1)
    Hk(:,:,i) = [1,0, 0,        0;...
        0, 1, 0,        0;...
        0, 0, xkm_predicted(3,i)/vm_predicted,   ...
        xkm_predicted(4,i) /vm_predicted];
    % Equation 17 Gain
    Kk(:,:,i) = Qxm_predicted(:,:,i) * Hk(:,:,i)'*inv([Rk +  ...
        Hk(:,:,i) * Qxm_predicted(:,:,i) * Hk(:,:,i)']);
    Lk(:,i) = [meas.east(i) meas.north(i)...
        meas.speed(i)]';
    hkm_predicted = [xkm_predicted(1,i) xkm_predicted(2,i) ...
        sqrt(xkm_predicted(3,i)^2 + xkm_predicted(4,i)^2)]';

    xk(:,i+1) = xkm_predicted(:,i) + Kk(:,:,i)*[ Lk(:,i) ...
        - hkm_predicted ]; % Equation18
    %     Measurement update
    % Equation 19
    Qx(:,:,i+1) = [eye(length(Kk(:,:,i)*...
        Hk(:,:,i)))-Kk(:,:,i)*Hk(:,:,i)]*Qxm_predicted(:,:,i);
    % Equation 22
    %     Lk = [meas.east(i) meas.north(i)...
```

```
    %         sqrt((xk(3,i))^2 + (xk(4,i))^2)]';

    %     Hk = inv(xk(:,i))*Lk;
    final.xplot(:,i+1) = xk(:,i);
end
```

Smoothing

```
xkhat(:,25) = xk(:,end);
nStep = 25;
count = nStep;
Qxkhat(:,:,25) = Qx(:,:,end);
for i = 1:(nStep-1)
    Dk = Qx(:,:,count+1)*Tk'*inv(Qxm_predicted(:,:,count));
    xkhat(:,count-1) = xk(:,count) + Dk*[xkhat(:,count) ...
        - xkm_predicted(:,count)];
    Qxkhat(:,:,count-1) = Qx(:,:,count+1) ...
        + Dk*[Qxkhat(:,:,count) - Qxm_predicted(:,:,count)]*Dk';
    count = count - 1;
end
```

Plot

```
final.x1 = xk(1,:)';  % final x values
final.y1 = xk(2,:)';  % final y values
meas.x2 = meas.east;  % original x
meas.y2 = meas.north;  % original y
true.x3 = true.east;  % true x
true.y3 = true.north;  % true y
figure('Units', 'pixels', ...
    'Position', [100 100 500 375]);
hold on;
y1_plot = plot(final.x1,final.y1, ...  % Before Smoothing
xkhat(1,:),xkhat(2,:), ...             % Smoothing
meas.x2,meas.y2, ...                   % Original
true.x3,true.y3)                       % True Plot
hTitle = title('Kalman filtering');
hXLabel = xlabel('Easting [m]');
hYLabel = ylabel('Northing [m]');
hLegend = legend(...
    'Before Smoothing',...
    'Smoothing',...
    'Original',...
    'True Plot',...
    'location','best');
set( gca                         , ...
    'FontName'   , 'Helvetica' );
set([hTitle, hXLabel, hYLabel], ...
    'FontName'   , 'AvantGarde');
set([hLegend, gca]               , ...
    'FontSize'   , 10             );
set([hXLabel, hYLabel]  , ...
    'FontSize'   , 11           );
```

```
set( hTitle                    , ...
    'FontSize'   , 13          , ...
    'FontWeight' , 'bold'      );
set(gca, ...
  'Box'         , 'off'         , ...
  'TickDir'     , 'out'         , ...
  'TickLength'  , [.02 .02]     , ...
  'XMinorTick'  , 'on'          , ...
  'YMinorTick'  , 'on'          , ...
  'YGrid'       , 'on'          , ...
  'XColor'      , [.3 .3 .3]    , ...
  'YColor'      , [.3 .3 .3]    , ...
  'LineWidth'   , 1             );
hold off;
```

Published with MATLAB® R2013a

Filtered Estimations and standard deviations				
t	$x_e \pm \sigma_e$	$x_n \pm \sigma_n$	$v_e \pm \sigma_{v_e}$	$v_n \pm \sigma_{v_n}$
0	-9.62 ± 10.00	0.11 ± 10.00	3.46 ± 3.00	0.84 ± 3.00
2	5.71 ± 1.71	4.00 ± 1.71	5.09 ± 0.90	1.31 ± 2.52
4	15.74 ± 1.33	8.98 ± 1.64	4.95 ± 0.51	1.97 ± 1.05
6	23.36 ± 1.23	19.32 ± 1.53	4.49 ± 0.37	3.35 ± 0.58
8	33.40 ± 1.19	25.17 ± 1.42	4.66 ± 0.30	3.20 ± 0.40
10	44.56 ± 1.17	28.26 ± 1.31	4.93 ± 0.27	2.68 ± 0.31
12	53.82 ± 1.15	33.15 ± 1.24	4.86 ± 0.25	2.62 ± 0.27
14	65.20 ± 1.13	36.34 ± 1.19	5.06 ± 0.24	2.32 ± 0.25
16	76.79 ± 1.11	41.27 ± 1.16	5.30 ± 0.23	2.38 ± 0.24
18	87.52 ± 1.09	46.04 ± 1.14	5.29 ± 0.23	2.37 ± 0.24
20	98.70 ± 1.09	49.89 ± 1.13	5.36 ± 0.23	2.24 ± 0.24
22	107.87 ± 1.08	56.46 ± 1.13	5.14 ± 0.23	2.53 ± 0.24
24	117.40 ± 1.08	63.85 ± 1.13	5.01 ± 0.23	2.85 ± 0.24
26	124.41 ± 1.08	69.08 ± 1.13	4.58 ± 0.23	2.78 ± 0.24
28	134.62 ± 1.08	72.63 ± 1.13	4.73 ± 0.23	2.50 ± 0.24
30	145.07 ± 1.08	81.00 ± 1.13	4.86 ± 0.23	2.96 ± 0.24
32	156.58 ± 1.08	85.84 ± 1.13	5.07 ± 0.23	2.78 ± 0.24
34	166.60 ± 1.08	91.13 ± 1.12	5.04 ± 0.23	2.74 ± 0.24
36	177.15 ± 1.08	95.37 ± 1.12	5.07 ± 0.23	2.55 ± 0.24
38	188.30 ± 1.08	97.05 ± 1.12	5.23 ± 0.23	2.07 ± 0.24
40	196.25 ± 1.08	101.75 ± 1.13	4.89 ± 0.23	2.16 ± 0.24
42	206.72 ± 1.08	104.80 ± 1.13	4.96 ± 0.23	1.97 ± 0.24
44	216.74 ± 1.08	109.52 ± 1.13	4.99 ± 0.23	2.08 ± 0.24
46	225.83 ± 1.08	116.11 ± 1.13	4.88 ± 0.23	2.43 ± 0.24
48	236.10 ± 1.08	120.72 ± 1.13	4.94 ± 0.23	2.39 ± 0.24

Table 1: The easting and northing coordinates and velocities are given with their standard deviations.

Smoothed Estimations and standard deviations				
t	$x_e \pm \sigma_e$	$x_n \pm \sigma_n$	$v_e \pm \sigma_{v_e}$	$v_n \pm \sigma_{v_n}$
0	5.20 ± 1.22	1.26 ± 1.53	4.92 ± 0.81	2.60 ± 2.47
2	15.13 ± 0.42	6.48 ± 0.37	4.96 ± 0.37	2.62 ± 0.97
4	25.14 ± 0.62	11.72 ± 0.41	4.98 ± 0.19	2.61 ± 0.46
6	35.28 ± 0.62	16.90 ± 0.37	5.03 ± 0.07	2.57 ± 0.24
8	45.53 ± 0.60	21.97 ± 0.46	5.10 ± 0.07	2.50 ± 0.10
10	55.84 ± 0.59	26.94 ± 0.53	5.14 ± 0.10	2.47 ± 0.08
12	66.18 ± 0.60	31.88 ± 0.58	5.17 ± 0.12	2.48 ± 0.12
14	76.48 ± 0.61	36.87 ± 0.61	5.16 ± 0.12	2.52 ± 0.13
16	86.66 ± 0.62	41.99 ± 0.63	5.13 ± 0.13	2.60 ± 0.13
18	96.69 ± 0.63	47.24 ± 0.65	5.05 ± 0.13	2.66 ± 0.13
20	106.60 ± 0.64	52.63 ± 0.65	4.98 ± 0.13	2.72 ± 0.13
22	116.47 ± 0.64	58.11 ± 0.65	4.94 ± 0.13	2.74 ± 0.13
24	126.38 ± 0.64	63.56 ± 0.65	4.94 ± 0.13	2.70 ± 0.13
26	136.43 ± 0.65	68.90 ± 0.65	4.98 ± 0.13	2.64 ± 0.13
28	146.56 ± 0.64	74.11 ± 0.65	5.05 ± 0.13	2.57 ± 0.13
30	156.71 ± 0.65	79.17 ± 0.65	5.08 ± 0.13	2.47 ± 0.13
32	166.79 ± 0.64	83.98 ± 0.66	5.06 ± 0.13	2.34 ± 0.13
34	176.81 ± 0.65	88.56 ± 0.66	5.03 ± 0.13	2.25 ± 0.13
36	186.75 ± 0.65	92.99 ± 0.67	4.99 ± 0.13	2.19 ± 0.13
38	196.63 ± 0.65	97.39 ± 0.67	4.95 ± 0.13	2.21 ± 0.14
40	206.49 ± 0.65	101.87 ± 0.67	4.93 ± 0.14	2.26 ± 0.14
42	216.34 ± 0.67	106.45 ± 0.68	4.93 ± 0.15	2.32 ± 0.15
44	226.22 ± 0.73	111.15 ± 0.74	4.93 ± 0.16	2.38 ± 0.17
46	236.10 ± 0.85	115.93 ± 0.88	4.94 ± 0.19	2.40 ± 0.20

Table 2: The easting and northing coordinates and velocities are given with their standard deviations.

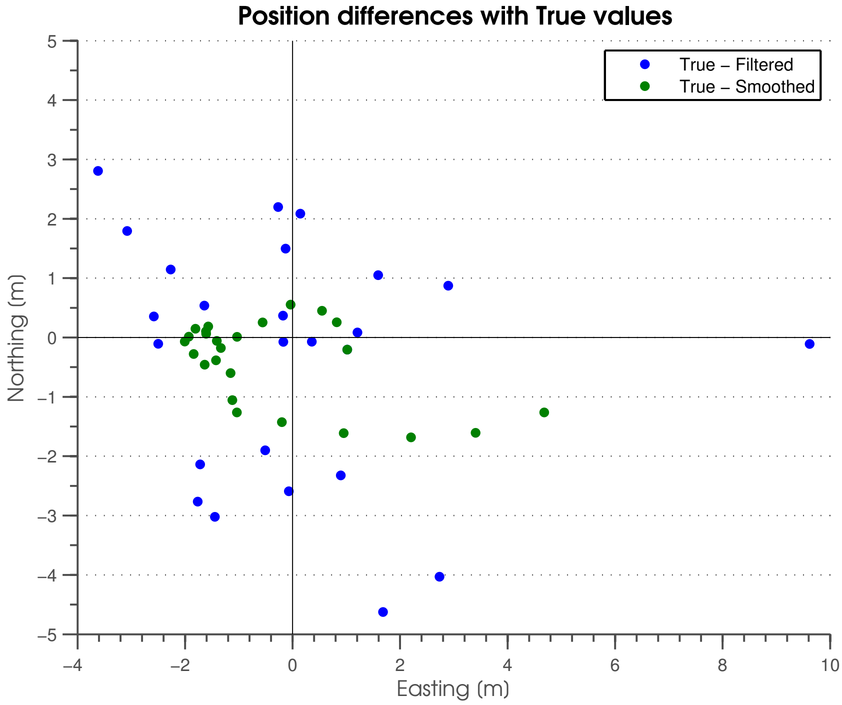

Figure 3: Plot of differences between true values and the filtered and smoothed coordinates.

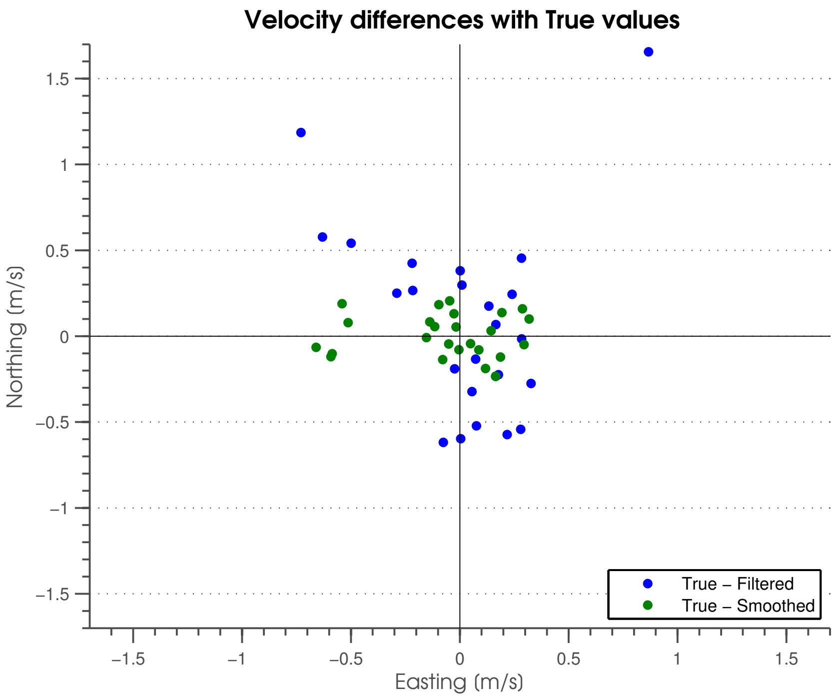

Figure 4: Plot of differences between true values and the filtered and smoothed velocities.