THE KALMAN FILTER AND SURVEYING APPLICATIONS

R.E. Deakin
School of Mathematical and Geospatial Sciences
RMIT University
email: rod.deakin@rmit.edu.au
June 2006

ABSTRACT

The *Kalman filter* is a set of equations, applied recursively, that can be used in surveying applications to obtain position, acceleration and velocity of a moving object from traditional surveying measurements of distance and direction. The aim of this paper is to introduce the surveyor to the Kalman filter by examination of two simple applications, (i) Electromagnetic Distance Measurement (EDM) and (ii) the position and velocity of a ship in a navigation channel.

INTRODUCTION

A Kalman filter is a set of equations that are applied recursively to estimate the *state* of a *system* from a sequence of *noisy* measurements at times $t_1, t_2, t_3, \ldots$ etc. The "state" of the system is its value or values at times $t_1, t_2, t_3, \ldots$ etc and a "system" may have a single value or multiple values. Say, for instance, the system is a ship steaming on a particular heading in a shipping channel and the state of the system (the ship) is its east and north coordinates $\left(E_k, N_k\right)$ and its velocity $\left(\dot{E}_k, \dot{N}_k\right)$. We say that this system (the ship) has a *state vector* $\mathbf{x}_k = \left[E_k, N_k, \dot{E}_k, \dot{N}_k\right]^T$ containing four elements and the subscript k indicates a value at time t_k.

[Note that in this paper, vectors are taken to mean *column-vectors*. A *row-vector* $\begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix}$ containing 1 row and n columns is the transpose of the column vector containing n rows and 1 column. To save space, column-vectors are often shown as $\begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix}^T$ where the notation $[\ ]^T$ indicates the transpose. A vector can also contain a single element].

On the other hand, a system may be a process such as electronic distance measurement (EDM) by phase comparison of emitted and reflected light beams. The state of this system is a single value, the distance (D_k), determined at times $t_1, t_2, t_3, \ldots$ etc, and this system (the EDM) has a state vector $\mathbf{x}_k = [D_k]$ containing a single element and the subscript k indicates a value at time t_k.

"Noisy" measurements are measurements that contain small *random errors* assumed to be *normally distributed,* i.e., the aggregation of errors in size groupings would form the familiar symmetric bell-shaped histogram with positive and negative errors equally likely and small errors more frequent than large errors. Surveyors usually talk of residuals (or corrections) rather than errors, where a residual is the same magnitude as an error but of opposite sign.

A Kalman filter gives the best estimates of the state of a *dynamic system* at a particular instant of time. And a dynamic system can be one whose values are changing with time, due to the motion of the system and measurement errors, or one whose values are measured at various instants of time and appear to change due to measurement errors. Dynamic systems do not have a single state (consisting of one or many values) that can be determined from a finite set of measurements but instead have a continuously changing state that has values sampled at different instants of time.

This paper aims to provide some insight into the Kalman filter and its implementation by studying two examples (i) the determination of a theoretical distance by an EDM; a dynamic system with a state vector containing a single value, and (ii) the determination of the position and velocity of a ship in a navigation channel; a dynamic system having a state vector containing four parameters.

THE KALMAN FILTER

The Kalman filter equations were published in 1960 by Dr. R.E. Kalman in his famous paper describing a new approach to the solution of linear filtering and prediction (Kalman 1960). Since that time, papers on the application of the technique have been filling numerous scientific journals and it is regarded as one of the most important algorithmic techniques ever devised. It has been used in applications ranging from navigating the Ranger and Apollo spacecraft in their lunar missions to predicting short-term fluctuations in the stock market. Sorenson (1970) shows Kalman's technique to be an extension of C.F. Gauss' original method of least squares developed in 1795 and provides an historical commentary on its practical solution of linear filtering problems studied by 20th century mathematicians.

The derivation of the Kalman filter equations can be found in many texts related to signal processing that is the usual domain of Electrical Engineers, e.g., Brown and Hwang, 1992. These derivations often use terminology that is unfamiliar to surveyors, but two authors, Krakiwsky (1975) and Cross (1992) both with geodesy\surveying backgrounds, have derivations, explanations, terminology and examples that would be familiar to any surveyor. This paper uses terminology similar to Cross and Krakiwsky. The Kalman filter equations and the associated measurement and dynamic models are given below with a brief explanation of the terms. It is hoped that the study of the two examples will make help to make the Kalman filter a relatively easily understood process.

The *primary models* (or *measurement models*) at times t_{k-1} and t_k, and the *secondary model* (or *dynamic model*) linking the states at t_{k-1} and t_k are given by the matrix equations

$$\begin{aligned} \mathbf{v}_{k-1} + \mathbf{B}_{k-1}\mathbf{x}_{k-1} &= \mathbf{f}_{k-1} && \text{primary model at } t_{k-1} \\ \mathbf{v}_k + \mathbf{B}_k\mathbf{x}_k &= \mathbf{f}_k && \text{primary model at } t_k \\ \mathbf{x}_k &= \mathbf{T}\mathbf{x}_{k-1} + \mathbf{v}_m && \text{dynamic model} \end{aligned} \qquad (1)$$

where **x** is the state vector

v is the vector of residuals associated with the measurements

B is a coefficient matrix

f is a vector of numeric terms derived from the measurements

$\mathbf{T}$ is the *transition* matrix

$\mathbf{v}_m$ is a vector of residuals associated with the dynamic model.

Enforcing the least squares condition that the sum of the squares of the residuals, (multiplied by coefficients reflecting their precisions) be a minimum, gives rise to the set of recursive equations (the Kalman filter) that are applied as follows

With initial estimates of the state vector $\mathbf{x}_{k-1}$ and the state cofactor matrix $\mathbf{Q}_{x_{k-1}}$ a Kalman filter has the following five general steps

(1) Compute the predicted state vector at t_k

$$\mathbf{x}'_k = \mathbf{T}\hat{\mathbf{x}}_{k-1} \tag{2}$$

(2) Compute the predicted state cofactor matrix at t_k

$$\mathbf{Q}'_{x_k} = \mathbf{T}\mathbf{Q}_{x_{k-1}}\mathbf{T}^T + \mathbf{Q}_m \tag{3}$$

(3) Compute the Kalman *Gain matrix*

$$\mathbf{K} = \mathbf{Q}'_{x_k}\mathbf{B}_k^T\left(\mathbf{Q} + \mathbf{B}_k\mathbf{Q}'_{x_k}\mathbf{B}_k^T\right)^{-1} \tag{4}$$

(4) Compute the filtered state vector by updating the predicted state with the measurements at t_k

$$\hat{\mathbf{x}}_k = \mathbf{x}'_k + \mathbf{K}\left(\mathbf{f}_k - \mathbf{B}_k\mathbf{x}'_k\right) \tag{5}$$

(5) Compute the filtered state cofactor matrix

$$\mathbf{Q}_{x_k} = \left(\mathbf{I} - \mathbf{K}_k\mathbf{B}_k\right)\mathbf{Q}'_{x_k} \tag{6}$$

Go to step **(1)** and repeat the process for the next measurement epoch t_{k+1}.

The "hat" symbol (^) above the vector $\mathbf{x}$ indicates that it is an estimate of the true (but unknown) state of the system derived from the Kalman filter. This is also known as the filtered state. The "prime" symbol ($'$) indicates a predicted quantity. The superscript T, e.g., $\mathbf{B}^T$ denotes the matrix transpose where the rows of $\mathbf{B}$ become the columns of $\mathbf{B}^T$. If $\mathbf{B}$ contains a single element, say $\mathbf{B} = [x]$ then $\mathbf{B}^T = \mathbf{B} = [x]$. The superscript -1, e.g., $\mathbf{A}^{-1}$ denotes the inverse of a matrix and matrix inversion is defined by $\mathbf{A}\mathbf{A}^{-1} = \mathbf{I}$ where $\mathbf{I}$ is the diagonal Identity matrix (ones on the leading-diagonal). This relationship gives rise to matrix inversion routines that some

computer languages offer as standard matrix functions. It would be assumed that any computer implementation of a Kalman filter would use a language (C++, Visual Basic, etc.) that had some standard matrix routines attached that offered matrix transposition, multiplication and inversion. If a matrix has only a single element, say $\mathbf{A} = [x]$ then $\mathbf{A}^{-1} = \left[1/x\right]$.

Cofactor matrices, designated $\mathbf{Q}$, contain estimates of variances and covariances associated with measurements $(\mathbf{Q})$, the elements of the state vector $(\mathbf{Q}_x)$ and the model residuals $(\mathbf{Q}_m)$. These $\mathbf{Q}$ matrices are often called covariance matrices. It should be noted that the cofactor matrix $\mathbf{Q}_m$ is derived in the following manner.

The dynamic model in equation (1)

$$\mathbf{x}_k = \mathbf{T}\mathbf{x}_{k-1} + \mathbf{v}_m \tag{7}$$

is an estimation of the true (but unknown) changes in the elements of the state vector from time t_{k-1} to time t_k and as such we assume that there are corrections to these estimations that are contained in the vector $\mathbf{v}_m$, the dynamic model residuals; and the elements of $\mathbf{v}_m$ are assumed to be small, random and normally distributed with a mean of zero. Also, we assume that the vector $\mathbf{v}_m$ is the product of two matrices, a coefficient matrix $\mathbf{H}$ and a vector $\mathbf{w}$ known as the *system driving noise*

$$\mathbf{v}_m = \mathbf{H}\mathbf{w} \tag{8}$$

The system driving noise $\mathbf{w}$ is a vector of random variables having variances and covariances contained in the cofactor matrix $\mathbf{Q}_w$ and applying the general law of propagation of variances to equation (8) gives

$$\mathbf{Q}_m = \mathbf{H}\mathbf{Q}_w\mathbf{H}^T \tag{9}$$

Determining $\mathbf{T}$, $\mathbf{H}$, $\mathbf{w}$, $\mathbf{Q}_w$, and $\mathbf{Q}_m$ by equation (9), will be discussed in the examples below.

The Kalman filter equations [(2) to (6)] are relatively easy to implement on modern computers (a reason for its popularity) and the examples studied below will be supplemented by MATLAB[1] computer code in the appendix.

---

[1] MATLAB, a registered trademark of The MathWorks, Inc., is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation.

DETERMINATION OF A THEORETICAL DISTANCE BY AN EDM

The EDM component of a Total Station measures distances by phase comparison of an emitted and reflected modulated light beam. The measurement is an electro/optical process and the distance we see displayed after pressing the measure button on the Total Station is the "filtered" value of many hundreds of individual measurements, since a measurement takes only a number of milliseconds. This value is the result of a Kalman filter process.

Consider the following sequence of measurements at times $t_1, t_2, t_3 \ldots$ etc,

355.416, 355.430, 355.412, 355.402, 355.419, ...

The variation in the measurements is assumed to be due to normally distributed random errors arising from the internal measurement process; often called the process noise. [The measurement sequence above, was generated by adding normally distributed random errors with mean zero and standard deviation 0.010 m to a constant value of 355.420 m.]

How will a Kalman filter produce the "filtered" value from this sequence?

First, let us assume that the measurement model is

$$\mathbf{l}_k + \mathbf{v}_k = \hat{\mathbf{l}}_k \tag{10}$$

where $\mathbf{l}_k$ is the n,1 vector of measurements, $\mathbf{v}_k$ is the n,1 vector of residuals (small unknown corrections to the measurements) and $\hat{\mathbf{l}}_k$ are estimates of the true (but unknown) value of the measurements. n is the number of measurements, that in this case is one. The primary measurement model can be expressed in terms of the filtered state vector $\hat{\mathbf{x}}_k$ at time t_k as

$$\mathbf{v}_k + \mathbf{B}_k\hat{\mathbf{x}}_k = \mathbf{f}_k \tag{11}$$

In this case $\hat{\mathbf{x}}_k$ contains the elements of $\hat{\mathbf{l}}_k$, both vectors containing single quantities and $\mathbf{f}_k = -\mathbf{l}_k$ also both containing single quantities (the measured distance at t_k). The matrix $\mathbf{B}$ will contain a single quantity, $\mathbf{B} = [-1]$.

Secondly, the dynamic model linking the elements of the state vector at times t_{k-1} and t_k is

$$\mathbf{x}_k = \mathbf{T}\mathbf{x}_{k-1} + \mathbf{v}_m \tag{12}$$

The state vector contains a single element that should remain unchanged between t_{k-1} and t_k (any change is simply due to measurement errors) then the transition matrix $\mathbf{T}$ will contain a single element $\mathbf{T} = [1]$ and there are no assumed corrections to this model; hence $\mathbf{v}_m = \mathbf{0}$, $\mathbf{H} = \mathbf{0}$, $\mathbf{w} = \mathbf{0}$ and from equation (9) $\mathbf{Q}_m = \mathbf{0}$.

Lastly, an estimate of the cofactor matrix of the measurements $\mathbf{Q}$ and the elements of the state vector $\mathbf{Q}_{x_k}$ must be made. Let us assume (guess) that the measurements have a standard deviation of 10 mm (0.010 m) and hence their estimated variance is $(0.010)^2$ and $\mathbf{Q} = \left[(0.010)^2\right]$. Since our primary measurement model has a state vector containing a single value (the measurement), then $\mathbf{Q}_x$ will only contain a single value, and we have as a starting estimate $\mathbf{Q}_{x_1} = \left[(0.010)^2\right]$, the same as $\mathbf{Q}$.

Now we can now start the Kalman filter at epoch t_2 using the values at t_1 as filtered estimates.

---

(1) Compute the predicted state vector at epoch t_2 using the measurement 355.416 at t_1 as the filtered estimate $\hat{\mathbf{x}}_1$

$$\begin{aligned}\mathbf{x}'_2 &= \mathbf{T}\hat{\mathbf{x}}_1 \\ &= [1][355.416] \\ &= 355.416\end{aligned}$$

(2) Compute the predicted state cofactor matrix at t_2 using $\mathbf{Q}_{x_1} = \left[(0.010)^2\right]$ as the filtered estimate

(2) continued

$$\begin{aligned}\mathbf{Q}'_{x_2} &= \mathbf{T}\mathbf{Q}_{x_1}\mathbf{T}^T + \mathbf{Q}_m \\ &= [1]\left[(0.010)^2\right][1] + 0 \\ &= (0.010)^2\end{aligned}$$

(3) Compute the Kalman *Gain matrix* noting that $\mathbf{Q} = \left[(0.010)^2\right]$

$$\begin{aligned}\mathbf{K} &= \mathbf{Q}'_{x_2}\mathbf{B}_2^T\left(\mathbf{Q} + \mathbf{B}_2\mathbf{Q}'_{x_2}\mathbf{B}_2^T\right)^{-1} \\ &= [(0.010)^2][-1] \\ &\quad \times ([(0.010)^2] + [-1][(0.010)^2][-1])^{-1} \\ &= \left[-(0.010)^2\right]\left[2(0.010)^2\right]^{-1} \\ &= -0.500\end{aligned}$$

(4) Compute the filtered state vector $\hat{\mathbf{x}}_2$ by updating the predicted state with the measurements at t_2

$$\begin{aligned}\hat{\mathbf{x}}_2 &= \mathbf{x}'_2 + \mathbf{K}\left(\mathbf{f}_2 - \mathbf{B}_2\mathbf{x}'_2\right)\\ &= [355.416] + \{[-0.500]\\ &\quad \times([-355.430] - [-1][355.416])\}\\ &= 355.423\end{aligned}$$

(5) Compute the filtered state cofactor matrix at t_2

$$\begin{aligned}\mathbf{Q}_{x_2} &= \left(\mathbf{I} - \mathbf{K}_2\mathbf{B}_2\right)\mathbf{Q}'_{x_2}\\ &= \left([1] - [-0.500][-1]\right)\left[(0.010)^2\right]\\ &= 0.000050\end{aligned}$$

Go to step **(1)** and repeat the process for the next measurement epoch t_3.

The values from the Kalman filter for epochs t_3, t_4 and t_5 are

---

epoch t_3	epoch t_4	epoch t_5
$\mathbf{x}'_3 = 355.423000$	$\mathbf{x}'_4 = 355.419333\ldots$	$\mathbf{x}'_5 = 355.415000$
$\mathbf{Q}'_{x_3} = 0.000050$	$\mathbf{Q}'_{x_4} = 0.0000333\ldots$	$\mathbf{Q}'_{x_5} = 0.000025$
$\mathbf{K}_3 = -0.333333\ldots$	$\mathbf{K}_4 = -0.250000$	$\mathbf{K}_5 = -0.200000$
$\hat{\mathbf{x}}_3 = 355.419333\ldots$	$\hat{\mathbf{x}}_4 = 355.415000$	$\hat{\mathbf{x}}_5 = 355.415800$
$\mathbf{Q}_{x_3} = 0.000033\ldots$	$\mathbf{Q}_{x_4} = 0.000025$	$\mathbf{Q}_{x_5} = 0.000020$

So, the sequence of measurements is

355.416, 355.403, 355.421, 355.423, 355.408, ...

and the Kalman filter estimates $\hat{\mathbf{x}}$ (the filtered values) are

355.416, 355.423, 355.419, 355.415, 355.416, ...

Something that should be noted is that the filtered state cofactor matrix $\mathbf{Q}_x$ contains the estimate of the variance of the filtered value (variance is standard deviation squared). We started with an estimated value $\mathbf{Q}_{x_1} = \mathbf{Q} = (0.010)^2 = 0.000100$ and after five epochs the estimated value had reduced to $\mathbf{Q}_{x_5} = 0.000020$ equivalent to an estimated standard deviation of 0.0045 m. So the Kalman filter gives estimates with a better precision than the assumed precision of the measurement sequence; as we

should expect from a least squares process. The Kalman filter for this simple case is very easily programmed and Appendix 1 contains a Kalman filter program (*edm.m*) written in the MATLAB language that processes 250 EDM measurements. These measurements are obtained by adding normally distributed random errors, with mean zero and standard deviation 0.010 m, to a constant value 355.420 m. Figure 1 below shows two plots (i) the filtered estimate of the distance (the filtered state) as a black solid line and the 250 measurements as black dots and (ii) a plot of the standard deviation of the filtered distance. After processing the 250 measurements the filtered distance was 355.420 m with a standard deviation of 0.000632 m.

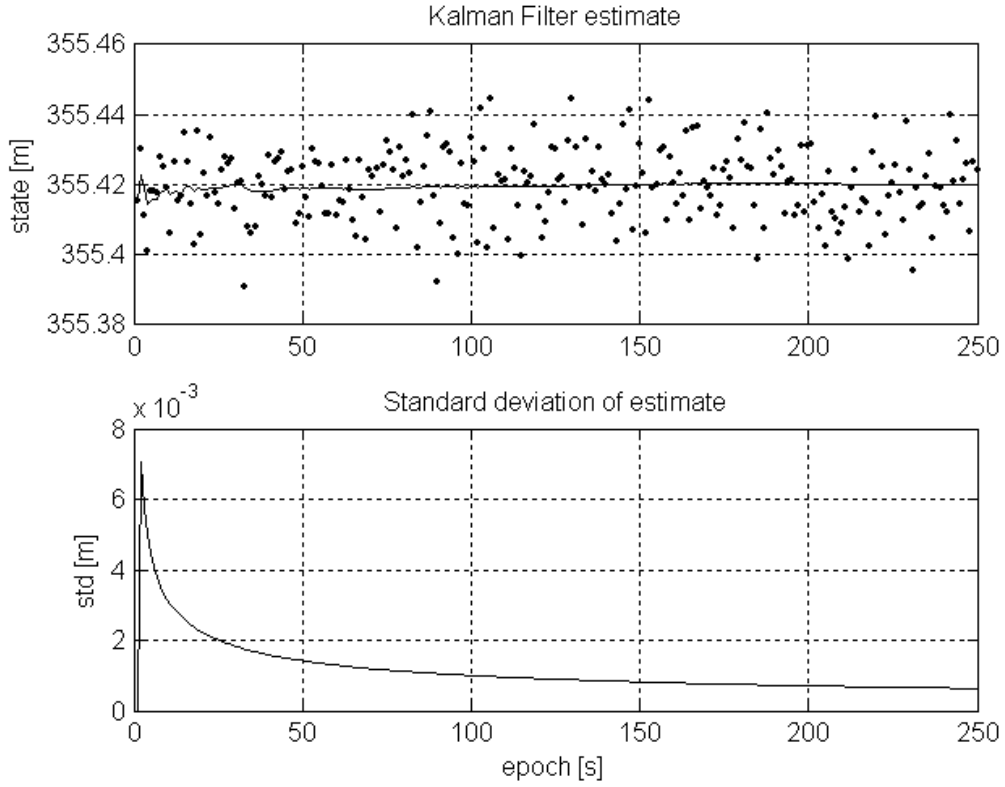
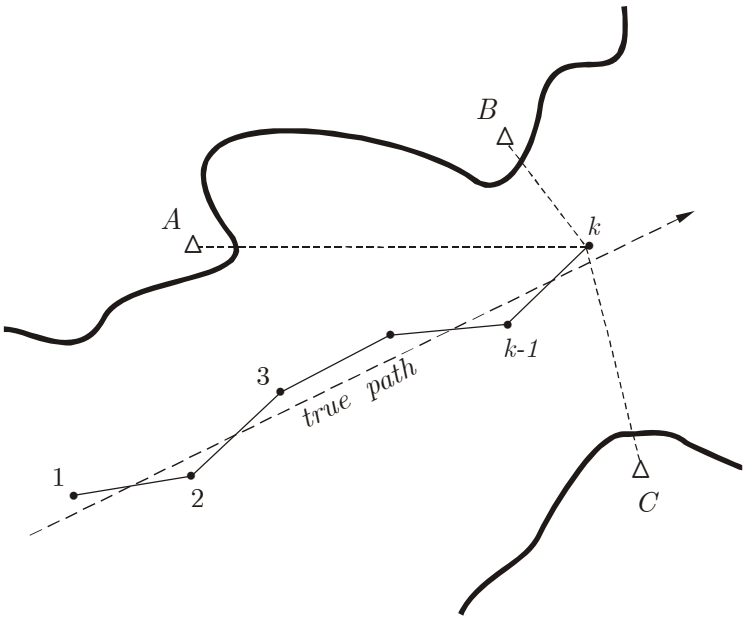


Figure 1. MATLAB plots of filtered state standard deviation of edm distance.

It is interesting to note that if all 250 measurements $x_1, x_2, x_3, \ldots, x_{250}$ (each with standard deviation $s_x = 0.010$ m) had been recorded and the mean $\overline{x} = \dfrac{x_1 + x_2 + \cdots + x_{250}}{250}$ computed, then propagation of variances gives the standard deviation of the mean as $s_{\overline{x}} = \dfrac{s_x}{\sqrt{250}} = 0.000632$ m which is the same as the Kalman filter result.

DETERMINATION OF POSITION AND VELOCITY OF A SHIP IN A NAVIGATION CHANNEL

Figure 2 shows the path of a ship in a navigation channel as it moves down the shipping channel at a constant heading and speed. Navigation equipment on board automatically measures distances to transponders at three navigation beacons *A, B* and *C* at 60-second intervals. The measured distances are known to have a standard deviation of 1 metre and the solid line in Figure 2 represents solutions of the ship's position for each set of measurements at the 60-second time intervals. The true path of the ship is shown as the dotted line.



The coordinates of the three navigation beacons are:

A : 10000.000 E
10000.000 N

B : 13880.000 E
11250.000 N

C : 15550.000 E
7160.000 N

Figure 2. Path of a ship in a navigation channel.

The transponder measurements, from the ship to the navigation beacons *A, B* and *C* at 60-second time intervals are shown in Table 1 below. The data were generated by assuming the starting coordinates of the ship were 7875.000 m East and 6319.392 m North and the ship was travelling at 15 knots on a heading of 064° (1 knot = 1 nautical mile per hour and 1 nautical mile = 1852 metres). At 60-second intervals, the true ship position and distances to the beacons were computed. These distances were then "disturbed" by the addition of normally distributed random errors (with zero mean and standard deviation 1 metre) and then rounded to the nearest 0.1 m.

Measurement Epoch	Transponder measurements to navigation beacons		
	A	B	C
1	4249.7	7768.6	7721.1
2	3876.1	7321.4	7288.5
3	3518.4	6872.2	6857.6
4	3193.3	6426.0	6429.1
5	2903.6	5982.6	6009.7
6	2664.0	5543.2	5596.6
7	2490.9	5107.7	5191.5
8	2392.9	4678.9	4797.1
9	2383.2	4253.4	4417.8
10	2463.0	3841.7	4050.9
11	2623.2	3435.6	3709.9
12	2849.0	3054.2	3395.8
13	3126.7	2692.9	3119.4
14	3446.9	2366.6	2891.1
15	3793.4	2096.4	2724.4
16	4166.0	1900.6	2630.9
17	4552.2	1804.7	2610.2
18	4956.2	1824.8	2677.4
19	5366.4	1959.6	2819.7
20	5785.0	2182.8	3023.5

Table 1. Transponder measurements at 60-second time intervals

How will a Kalman filter produce an estimated position, speed and heading of the ship from the transponder measurements?

Note that in our Kalman filter, the state vector will be $\mathbf{x}_k = \left[E_k, N_k, \dot{E}_k, \dot{N}_k\right]^T$ containing $n = 4$ elements (or parameters) where $\left(E_k, N_k\right)$ are the ship's position and $\left(\dot{E}_k, \dot{N}_k\right)$ the ship's velocity components. The speed of the ship at time t_k is

$$speed = \sqrt{\left(\dot{E}_k\right)^2 + \left(\dot{N}_k\right)^2} \tag{13}$$

and the heading of the ship (bearing from North) at time t_k is

$$\tan\left(heading\right) = \frac{\dot{E}_k}{\dot{N}_k} \tag{14}$$

The measurement model (primary model)

Let us assume that the primary or measurement model is

$$\mathbf{l}_k + \mathbf{v}_k = \hat{\mathbf{l}}_k \tag{15}$$

where $\mathbf{l}_k$ is the m,1 vector of measurements (the transponder distances), $\mathbf{v}_k$ is the m,1 vector of residuals (small unknown corrections to the measurements) and $\hat{\mathbf{l}}_k$ are estimates of the true (but unknown) value of the measurements. m is the number of measurements, that in this case is three at each measurement epoch. The estimates $\hat{\mathbf{l}}_k$ are non-linear functions of the coordinates E,N of the beacons A, B and C and the filtered state coordinates $\hat{E}_k, \hat{N}_k$ of the ship at time t_k

$$\hat{l}_j = \hat{l}\left(\hat{E}_k, \hat{N}_k, E_j, N_j\right) = \sqrt{\left(\hat{E}_k - E_j\right)^2 + \left(\hat{N}_k - N_j\right)^2} \qquad \text{for } j = A, B, C \tag{16}$$

Expanding equation (16) into a series using Taylor's theorem gives

$$\hat{l} = l' + \frac{\partial \hat{l}}{\partial \hat{E}_k}\left(\hat{E}_k - E'_k\right) + \frac{\partial \hat{l}}{\partial \hat{N}_k}\left(\hat{N}_k - N'_k\right) + \text{higher order terms}$$

where E'_k, N'_k are approximate coordinates of the ship at t_k, l' is an approximate distance computed using E'_k, N'_k and the coordinates of the beacon, and the partial derivatives are

$$\begin{aligned} \frac{\partial \hat{l}}{\partial \hat{E}_k} &= \frac{E'_k - E_j}{l'_j} = d_j \\ \frac{\partial \hat{l}}{\partial \hat{N}_k} &= \frac{N'_k - N_j}{l'_j} = c_j \qquad \text{for } j = A, B, C \end{aligned} \tag{17}$$

Re-arranging equation (15) for a single distance gives

$$v - \hat{l} = -l$$

and substituting the Taylor series approximation for $\hat{l}$ (ignoring higher-order terms) and re-arranging gives the linearized form of the primary measurement model as

$$v_j - d_j\hat{E}_k - c_j\hat{N}_k = l'_j - l_j + \left(-d_jE'_k - c_jN'_k\right) \qquad \text{for } j = A, B, C \tag{18}$$

This primary measurement model can be expressed in terms of the filtered state vector $\hat{\mathbf{x}}_k$ at time t_k in the matrix form as

$$\begin{bmatrix} v_A \\ v_B \\ v_C \end{bmatrix}_k + \begin{bmatrix} -d_A & -c_A & 0 & 0 \\ -d_B & -c_B & 0 & 0 \\ -d_C & -c_C & 0 & 0 \end{bmatrix}_k \begin{bmatrix} \hat{E} \\ \hat{N} \\ \dot{E} \\ \dot{N} \end{bmatrix}_k = \begin{bmatrix} l'_A - l_A \\ l'_B - l_B \\ l'_C - l_C \end{bmatrix}_k + \begin{bmatrix} -d_A & -c_A & 0 & 0 \\ -d_B & -c_B & 0 & 0 \\ -d_C & -c_C & 0 & 0 \end{bmatrix}_k \begin{bmatrix} E' \\ N' \\ \dot{E}' \\ \dot{N}' \end{bmatrix}_k$$

or
$$\mathbf{v}_k + \mathbf{B}_k \hat{\mathbf{x}}_k = \mathbf{l}'_k - \mathbf{l}_k + \mathbf{B}_k \mathbf{x}'_k = \mathbf{f}_k \tag{19}$$

Now in step (4) of the Kalman filter algorithm [see equation (5)], the filtered state vector $\hat{\mathbf{x}}_k$ is obtained from

$$\hat{\mathbf{x}}_k = \mathbf{x}'_k + \mathbf{K}\left(\mathbf{f}_k - \mathbf{B}_k \mathbf{x}'_k\right) \tag{20}$$

and substituting for $\mathbf{f}_k$ from equation (19) gives

$$\begin{aligned} \hat{\mathbf{x}}_k &= \mathbf{x}'_k + \mathbf{K}\left(\mathbf{l}'_k - \mathbf{l}_k + \mathbf{B}_k \mathbf{x}'_k - \mathbf{B}_k \mathbf{x}'_k\right) \\ &= \mathbf{x}'_k + \mathbf{K}\left(\mathbf{l}'_k - \mathbf{l}_k\right) \end{aligned} \tag{21}$$

Note: (i) the term $\left(\mathbf{f}_k - \mathbf{B}_k \mathbf{x}'_k\right)$ in equation (20) is often called the *predicted residuals* $\mathbf{v}'_k$ where, in our case

$$\mathbf{v}'_k = \mathbf{f}_k - \mathbf{B}_k \mathbf{x}'_k = \mathbf{l}'_k - \mathbf{l}_k \tag{22}$$

(ii) The term $\mathbf{K}\left(\mathbf{l}'_k - \mathbf{l}_k\right)$ in equation (21) is often called the *corrections to the predicted state* $\Delta\mathbf{x}$ where, in our case

$$\Delta\mathbf{x}_k = \mathbf{K}_k\left(\mathbf{l}'_k - \mathbf{l}_k\right) \tag{23}$$

The dynamic model (secondary model)

A dynamic model that is extremely simple and often used in navigation problems can be developed by considering a continuous function of time, say $y = y(t)$. Following the development by Cross (1987), we can use Taylor's theorem to expand the function $y(t)$ about the point $t = t_k$ into the series

$$y(t) = y(t_k) + (t - t_k)\dot{y}(t_k) + \frac{(t - t_k)^2}{2!}\ddot{y}(t_k) + \frac{(t - t_k)^3}{3!}\dddot{y}(t_k) + \cdots$$

where $\dot{y}(t_k), \ddot{y}(t_k), \dddot{y}(t_k)$, etc are derivatives of y with respect to t evaluated at $t = t_k$. Letting $t = t_k + \delta t$ and then $\delta t = t - t_k$ we may write

$$y(t_k + \delta t) = y(t_k) + \dot{y}(t_k)\delta t + \frac{\ddot{y}(t_k)}{2!}(\delta t)^2 + \frac{\dddot{y}(t_k)}{3!}(\delta t)^3 + \frac{\ddddot{y}(t_k)}{4!}(\delta t)^4 + \cdots \tag{24}$$

We now have power series expression for the continuous function $y(t)$ at the point $t = t_k + \delta t$ involving the function y and its derivatives $\dot{y}$, $\ddot{y}$, etc , (all evaluated at t_k) and the time difference $\delta t = t - t_k$.

In a similar manner, if we assume $\dot{y}(t)$, $\ddot{y}(t)$, etc to be continuous functions of t then

$$\begin{aligned}
\dot{y}(t_k + \delta t) &= \dot{y}(t_k) + \ddot{y}(t_k)\delta t + \frac{\dddot{y}(t_k)}{2!}(\delta t)^2 + \frac{\ddddot{y}(t_k)}{3!}(\delta t)^3 + \cdots \\
\ddot{y}(t_k + \delta t) &= \ddot{y}(t_k) + \dddot{y}(t_k)\delta t + \frac{\ddddot{y}(t_k)}{2!}(\delta t)^2 + \cdots \\
\dddot{y}(t_k + \delta t) &= \dddot{y}(t_k) + \ddddot{y}(t_k)\delta t + \cdots \\
&\text{etc}
\end{aligned} \tag{25}$$

Now consider two time epochs t_k and t_{k-1} separated by a time interval δt , we can combine equations (24) and (25) (with a change of subscripts for t) into the general matrix forms:

(i) involving terms up to $\ddot{y}$

$$\begin{bmatrix} y \\ \dot{y} \end{bmatrix}_k = \begin{bmatrix} 1 & \delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} y \\ \dot{y} \end{bmatrix}_{k-1} + \begin{bmatrix} \frac{1}{2}(\delta t)^2 \\ \delta t \end{bmatrix} [\ddot{y}]_{k-1} \tag{26}$$

(ii) involving terms up to $\dddot{y}$

$$\begin{bmatrix} y \\ \dot{y} \\ \ddot{y} \end{bmatrix}_k = \begin{bmatrix} 1 & \delta t & \frac{1}{2}(\delta t)^2 \\ 0 & 1 & \delta t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} y \\ \dot{y} \\ \ddot{y} \end{bmatrix}_{k-1} + \begin{bmatrix} \frac{1}{6}(\delta t)^3 \\ \frac{1}{2}(\delta t)^2 \\ \delta t \end{bmatrix} [\dddot{y}]_{k-1} \tag{27}$$

(iii) involving terms up to $\ddddot{y}$

$$\begin{bmatrix} y \\ \dot{y} \\ \ddot{y} \\ \dddot{y} \end{bmatrix}_k = \begin{bmatrix} 1 & \delta t & \frac{1}{2}(\delta t)^2 & \frac{1}{6}(\delta t)^3 \\ 0 & 1 & \delta t & \frac{1}{2}(\delta t)^2 \\ 0 & 0 & 1 & \delta t \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} y \\ \dot{y} \\ \ddot{y} \\ \dddot{y} \end{bmatrix}_{k-1} + \begin{bmatrix} \frac{1}{24}(\delta t)^4 \\ \frac{1}{6}(\delta t)^3 \\ \frac{1}{2}(\delta t)^2 \\ \delta t \end{bmatrix} [\ddddot{y}]_{k-1} \tag{28}$$

In Kalman filtering and navigation problems the continuous function of time $y = y(t)$ is simply *position* so that $y(t) = \{E, N\}(t)$ where *E,N* are east and north coordinates. The derivatives are *velocity,* $\dot{y}(t) = \left\{\dot{E}, \dot{N}\right\}(t)$ (rate of change of distance), *acceleration,* $\ddot{y}(t) = \left\{\ddot{E}, \ddot{N}\right\}(t)$ (rate of change of velocity), *jerk,* $\dddot{y}(t) = \left\{\dddot{E}, \dddot{N}\right\}(t)$ (rate of change of acceleration) and rate of change of jerk, $\ddddot{y}(t) = \left\{\ddddot{E}, \ddddot{N}\right\}(t)$. In each of the cases above [equations (26), (27) and (28)], we can consider the vector on the left-hand-side of the equals sign to be the vector $\mathbf{x}_k$, the state vector, or the state the system at time t_k. The matrix on the right-hand-side is the *transition matrix* $\mathbf{T}$ and the elements of this matrix contain the links between the state vector at times t_k and t_{k-1}, i.e., $\mathbf{x}_k = \mathbf{T}\mathbf{x}_{k-1}$. The second term in the equations above is the product of two matrices (in these cases two vectors) and the result will be the vector of model residuals $\mathbf{v}_m$ (containing the same number of elements as the state vector). $\mathbf{v}_m$ is a reflection of the fact that the transition matrix does not fully describe the exact physical links between the states at times t_k and t_{k-1} and $\mathbf{v}_m = \mathbf{H}\mathbf{w}$ where $\mathbf{H}$ is a coefficient matrix and $\mathbf{w}$ is the *system driving noise.* In the equations above the system driving noise is acceleration, jerk and rate of change of jerk respectively.

We can now use these general forms to define a suitable dynamic model.

In our simple case (the ship in the channel) the state vector $\mathbf{x}$ contains four elements $\mathbf{x}_k = \left[E_k, N_k, \dot{E}_k, \dot{N}_k\right]^T$ and the appropriate dynamic model in the form of equation (26) is

$$\begin{bmatrix} E \\ N \\ \dot{E} \\ \dot{N} \end{bmatrix}_k = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} E \\ N \\ \dot{E} \\ \dot{N} \end{bmatrix}_{k-1} + \begin{bmatrix} \frac{1}{2}(\Delta t)^2 & 0 \\ 0 & \frac{1}{2}(\Delta t)^2 \\ \Delta t & 0 \\ 0 & \Delta t \end{bmatrix} \begin{bmatrix} \ddot{E} \\ \ddot{N} \end{bmatrix}_{k-1} \tag{29}$$

or

$$\mathbf{x}_k = \mathbf{T}\mathbf{x}_{k-1} + \mathbf{v}_m \tag{30}$$

where $\mathbf{T}$ is the *n,n* transition matrix and $\mathbf{v}_m$ is the *n,*1 vector of model residuals.

If we expand equation (29) we see that it is really just the matrix form of the two equations of rectilinear motion; (i) $v = u + at$ and (ii) $s = ut + \frac{1}{2}at^2$ where s is distance, u is initial velocity, v is final velocity, a is acceleration and t is time. In our notation they are:

(i)
$$\begin{aligned} \dot{E}_k &= \dot{E}_{k-1} + \ddot{E}\,\Delta t \\ \dot{N}_k &= \dot{N}_{k-1} + \ddot{N}\,\Delta t \end{aligned}$$

and (ii)
$$\begin{aligned} E_k &= E_{k-1} + \dot{E}_{k-1}\,\Delta t + \tfrac{1}{2}\ddot{E}\,(\Delta t)^2 \\ N_k &= N_{k-1} + \dot{N}_{k-1}\,\Delta t + \tfrac{1}{2}\ddot{N}\,(\Delta t)^2 \end{aligned}$$

The model residuals $\mathbf{v}_m = \mathbf{H}\mathbf{w}$ are

$$\begin{bmatrix} v_E \\ v_N \\ v_{\dot{E}} \\ v_{\dot{N}} \end{bmatrix} = \begin{bmatrix} \frac{1}{2}(\Delta t)^2 & 0 \\ 0 & \frac{1}{2}(\Delta t)^2 \\ \Delta t & 0 \\ 0 & \Delta t \end{bmatrix} \begin{bmatrix} \ddot{E} \\ \ddot{N} \end{bmatrix} \tag{31}$$

where the coefficient matrix $\mathbf{H}$ and the system driving noise $\mathbf{w}$ are

$$\mathbf{H} = \begin{bmatrix} \frac{1}{2}(\Delta t)^2 & 0 \\ 0 & \frac{1}{2}(\Delta t)^2 \\ \Delta t & 0 \\ 0 & \Delta t \end{bmatrix} \quad \text{and} \quad \mathbf{w} = \begin{bmatrix} \ddot{E} \\ \ddot{N} \end{bmatrix} \tag{32}$$

In this simple navigation problem it is assumed that the system driving noise $\mathbf{w}$ contains small random accelerations caused by the sea and wind conditions, the steering of the ship, the engine speed variation, etc.

The cofactor matrix of the dynamic model $\mathbf{Q}_m$ is given by

$$\mathbf{Q}_m = \mathbf{H}\mathbf{Q}_w\mathbf{H}^T \tag{33}$$

where $\mathbf{Q}_w$, the cofactor matrix of the system driving noise, is

$$\mathbf{Q}_w = \begin{bmatrix} s_{\ddot{E}}^2 & 0 \\ 0 & s_{\ddot{N}}^2 \end{bmatrix}$$

and $s^2_{\ddot{E}}$, $s^2_{\ddot{N}}$ are the estimates of the variances of the accelerations in the east and north directions and the covariance is assumed to be zero. Using the coefficient matrix **H** in equation (32) we have

$$\mathbf{Q}_m = \begin{bmatrix} \frac{1}{2}(\Delta t)^2 & 0 \\ 0 & \frac{1}{2}(\Delta t)^2 \\ \Delta t & 0 \\ 0 & \Delta t \end{bmatrix} \begin{bmatrix} s^2_{\ddot{E}} & 0 \\ 0 & s^2_{\ddot{N}} \end{bmatrix} \begin{bmatrix} \frac{1}{2}(\Delta t)^2 & 0 & \Delta t & 0 \\ 0 & \frac{1}{2}(\Delta t)^2 & 0 & \Delta t \end{bmatrix} \tag{34}$$

Now we can now start the Kalman filter, but some initial values must be set beforehand. These will be designated (a), (b), (c) etc and then the Kalman filter steps (1), (2), (3) etc.

(a) Set the elements of the transition matrix

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

In this exercise $\Delta t = 60\,\text{sec}$

(b) Set the cofactor matrix of the measurements

$$\mathbf{Q} = \begin{bmatrix} s^2_{l_A} & 0 & 0 \\ 0 & s^2_{l_B} & 0 \\ 0 & 0 & s^2_{l_C} \end{bmatrix}$$

In this exercise

$s^2_{l_A} = s^2_{l_B} = s^2_{l_C} = 1.0\ \text{m}^2$

(c) Set the cofactor matrix of the system driving noise

$$\mathbf{Q}_w = \begin{bmatrix} s^2_{\ddot{E}} & 0 \\ 0 & s^2_{\ddot{N}} \end{bmatrix}$$

In this exercise

$s^2_{\ddot{E}} = s^2_{\ddot{N}} = 0.017\ \text{m}^2/\text{s}^4$

(d) Set the coefficient matrix of the system driving noise

$$\mathbf{H} = \begin{bmatrix} \frac{1}{2}(\Delta t)^2 & 0 \\ 0 & \frac{1}{2}(\Delta t)^2 \\ \Delta t & 0 \\ 0 & \Delta t \end{bmatrix}$$

(e) Compute the cofactor matrix of the dynamic model

$$\mathbf{Q}_m = \mathbf{H}\mathbf{Q}_w\mathbf{H}^T$$

(f) Set the starting estimates of the state vector. This will be the filtered state vector for epoch t_2

$$\mathbf{x}_1 = \begin{bmatrix} E \\ N \\ \dot{E} \\ \dot{N} \end{bmatrix}_1 = \begin{bmatrix} 7875.000 \text{ m} \\ 6319.392 \text{ m} \\ 7 \text{ m/s} \\ 3 \text{ m/s} \end{bmatrix}$$

(g) Set the starting estimates of the state cofactor matrix. This will be the filtered state cofactor matrix for epoch t_2

$$\mathbf{Q}_{x_k} = \begin{bmatrix} s_E^2 & 0 & 0 & 0 \\ 0 & s_N^2 & 0 & 0 \\ 0 & 0 & s_{\dot{E}}^2 & 0 \\ 0 & 0 & 0 & s_{\dot{N}}^2 \end{bmatrix}$$

In this exercise $s_E^2 = s_N^2 = 20\,\text{m}^2$ and $s_{\dot{E}}^2 = s_{\dot{N}}^2 = 0.5\ \text{m}^2/\text{s}^2$

Now start the Kalman filter at epoch t_2

(1) Compute the predicted state vector at epoch t_2 using the filtered estimate $\hat{\mathbf{x}}_1$

$$\mathbf{x}_2' = \mathbf{T}\hat{\mathbf{x}}_1$$

(2) Compute the predicted state cofactor matrix at t_2 using $\mathbf{Q}_m$ from step (e)

$$\mathbf{Q}_{x_2}' = \mathbf{T}\mathbf{Q}_{x_1}\mathbf{T}^T + \mathbf{Q}_m$$

(3) Compute the Kalman *Gain matrix*

$$\mathbf{K} = \mathbf{Q}_{x_2}'\mathbf{B}_2^T\left(\mathbf{Q} + \mathbf{B}_2\mathbf{Q}_{x_2}'\mathbf{B}_2^T\right)^{-1}$$

Using $\mathbf{Q}$ from step (b) and $\mathbf{B}$ whose elements have been determined using equations (17). The form of $\mathbf{B}$ is given in equation (19).

(4.1) Compute the numeric terms "computed – observed" distances [see equation (21)]

(4.2) Compute the filtered state vector $\hat{\mathbf{x}}_2$ by updating the predicted state [see equation (21)]

$$\hat{\mathbf{x}}_k = \mathbf{x}_k' + \mathbf{K}\left(\mathbf{l}_k' - \mathbf{l}_k\right)$$

(5) Compute the filtered state cofactor matrix at t_2

$$\mathbf{Q}_{x_2} = \left(\mathbf{I} - \mathbf{K}_2\mathbf{B}_2\right)\mathbf{Q}_{x_2}'$$

Go to step **(1)** and repeat the process for the next measurement epoch t_3.

The following output from a MATLAB program *kalship3.m* (see Appendix 2) processes the data in Table 1 in a Kalman filter beginning at epoch 2.

```
>> help kalship3

  KALSHIP   This function implements a Kalman Filter to estimate the POSITION and
    VELOCITY of a ship in a channel.
    The observations are distances to the ship at regular time intervals
    from three known locations.

  eg, kalship3('c:\projects\kalman\exercise\Kalshipdata3.dat');


>> kalship3('c:\projects\kalman\exercise\Kalshipdata3.dat');

epoch =   2
Filtered State  Corrns        Filtered State cofactor matrix Qxx
   8289.594     -5.406        1.009225 -0.797965  0.033097 -0.026169
   6521.882     22.490       -0.797965  1.439797 -0.026169  0.047217
      6.823     -0.177        0.033097 -0.026169  0.506780 -0.000858
      3.738      0.738       -0.026169  0.047217 -0.000858  0.507243

epoch =   3
Filtered State  Corrns        Filtered State cofactor matrix Qxx
   8705.780      6.823        0.926924 -0.643621  0.030398 -0.021105
   6727.944    -18.189       -0.643621  1.218371 -0.021105  0.039955
      7.046      0.224        0.030398 -0.021105  0.494715 -0.004015
      3.141     -0.596       -0.021105  0.039955 -0.004015  0.496822

epoch =   4
Filtered State  Corrns        Filtered State cofactor matrix Qxx
   9124.759     -3.808        0.863463 -0.498398  0.028327 -0.016346
   6928.604     12.198       -0.498398  1.011477 -0.016346  0.033180
      6.922     -0.125        0.028327 -0.016346  0.483077 -0.006336
      3.541      0.400       -0.016346  0.033180 -0.006336  0.486133

epoch =   5
Filtered State  Corrns        Filtered State cofactor matrix Qxx
   9540.095      0.042        0.798512 -0.363666  0.026205 -0.011929
   7132.756     -8.319       -0.363666  0.856483 -0.011929  0.028105
      6.923      0.001        0.026205 -0.011929  0.471881 -0.007919
      3.268     -0.273       -0.011929  0.028105 -0.007919  0.475282

epoch =   6
Filtered State  Corrns        Filtered State cofactor matrix Qxx
   9955.721      0.252        0.745133 -0.241643  0.024463 -0.007926
   7335.908      7.063       -0.241643  0.744933 -0.007927  0.024453
      6.931      0.008        0.024463 -0.007927  0.461092 -0.008861
      3.500      0.232       -0.007926  0.024453 -0.008861  0.464473

epoch =   7
Filtered State  Corrns        Filtered State cofactor matrix Qxx
  10372.275      0.680        0.694954 -0.127524  0.022824 -0.004181
   7537.141     -8.768       -0.127524  0.685548 -0.004182  0.022513
      6.953      0.022        0.022824 -0.004182  0.450710 -0.009250
      3.212     -0.288       -0.004181  0.022513 -0.009250  0.453885

epoch =   8
Filtered State  Corrns        Filtered State cofactor matrix Qxx
  10787.414     -2.069        0.655321 -0.030086  0.021530 -0.000982
   7739.732      9.867       -0.030086  0.681207 -0.000982  0.022378
      6.886     -0.068        0.021530 -0.000982  0.440722 -0.009161
      3.536      0.324       -0.000982  0.022378 -0.009161  0.443714
```

```
epoch =   9
Filtered State  Corrns        Filtered State cofactor matrix Qxx
  11203.122      2.572        0.621011  0.055445  0.020410  0.001828
   7943.701     -8.202        0.055445  0.730235  0.001829  0.023998
      6.970      0.084        0.020410  0.001829  0.431131 -0.008687
      3.267     -0.270        0.001828  0.023998 -0.008687  0.434157

epoch =  10
Filtered State  Corrns        Filtered State cofactor matrix Qxx
  11619.834     -1.491        0.599578  0.119122  0.019712  0.003921
   8143.904      4.203        0.119122  0.803937  0.003923  0.026428
      6.921     -0.049        0.019712  0.003923  0.421936 -0.007908
      3.405      0.138        0.003921  0.026428 -0.007908  0.425365

epoch =  11
Filtered State  Corrns        Filtered State cofactor matrix Qxx
  12037.433      2.334        0.590729  0.165695  0.019427  0.005453
   8349.749      1.557        0.165695  0.871649  0.005456  0.028662
      6.998      0.077        0.019427  0.005456  0.413158 -0.006927
      3.456      0.051        0.005453  0.028662 -0.006927  0.417358

epoch =  12
Filtered State  Corrns        Filtered State cofactor matrix Qxx
  12452.105     -5.198        0.602166  0.198841  0.019809  0.006544
   8550.099     -7.011        0.198841  0.893693  0.006547  0.029394
      6.827     -0.171        0.019809  0.006547  0.404827 -0.005820
      3.225     -0.231        0.006544  0.029394 -0.005820  0.410018

epoch =  13
Filtered State  Corrns        Filtered State cofactor matrix Qxx
  12868.305      6.591        0.641345  0.229261  0.021103  0.007545
   8754.315     10.692        0.229261  0.864696  0.007548  0.028447
      7.044      0.217        0.021103  0.007548  0.397006 -0.004638
      3.577      0.352        0.007545  0.028447 -0.004638  0.403101

epoch =  14
Filtered State  Corrns        Filtered State cofactor matrix Qxx
  13286.287     -4.642        0.722813  0.270807  0.023789  0.008914
   8958.950     -9.994        0.270807  0.806893  0.008916  0.026551
      6.891     -0.153        0.023789  0.008916  0.389795 -0.003378
      3.248     -0.329        0.008914  0.026551 -0.003378  0.396386

epoch =  15
Filtered State  Corrns        Filtered State cofactor matrix Qxx
  13699.993      0.251        0.855700  0.341580  0.028169  0.011244
   9160.935      7.088        0.341580  0.769390  0.011246  0.025322
      6.899      0.008        0.028169  0.011246  0.383349 -0.001975
      3.482      0.233        0.011244  0.025322 -0.001975  0.389787

epoch =  16
Filtered State  Corrns        Filtered State cofactor matrix Qxx
  14116.846      2.900        1.021943  0.435529  0.033647  0.014338
   9365.035     -4.795        0.435529  0.770140  0.014340  0.025352
      6.995      0.095        0.033647  0.014340  0.377823 -0.000314
      3.324     -0.158        0.014338  0.025352 -0.000314  0.383397

epoch =  17
Filtered State  Corrns        Filtered State cofactor matrix Qxx
  14531.436     -5.091        1.097853  0.503116  0.036151  0.016564
   9565.143      0.684        0.503116  0.809750  0.016567  0.026660
      6.827     -0.168        0.036151  0.016567  0.373179  0.001640
      3.346      0.023        0.016564  0.026660  0.001640  0.377365
```

```
epoch =  18
Filtered State  Corrns        Filtered State cofactor matrix Qxx
  14950.377      9.319        0.955699  0.439824  0.031474  0.014481
   9770.483      4.566        0.439824  0.822675  0.014483  0.027090
      7.134      0.307        0.031474  0.014483  0.368845  0.003650
      3.497      0.150        0.014481  0.027090  0.003650  0.371765

epoch =  19
Filtered State  Corrns        Filtered State cofactor matrix Qxx
  15366.544    -11.869        0.732074  0.288579  0.024112  0.009501
   9973.570     -6.708        0.288579  0.820826  0.009502  0.027033
      6.743     -0.391        0.024112  0.009502  0.364017  0.005200
      3.276     -0.221        0.009501  0.027033  0.005200  0.366470

epoch =  20
Filtered State  Corrns        Filtered State cofactor matrix Qxx
  15781.273     10.148        0.598119  0.158080  0.019704  0.005204
  10175.278      5.167        0.158080  0.847313  0.005203  0.027911
      7.077      0.334        0.019704  0.005203  0.358551  0.006055
      3.446      0.170        0.005204  0.027911  0.006055  0.361445

Filtered Values
Epoch  Distance  Velocity  Heading
  1       0.000    7.616   66.801
  2     461.400    7.779   61.286
  3     925.806    7.715   65.975
  4    1390.357    7.775   62.905
  5    1853.155    7.656   64.729
  6    2315.773    7.765   63.208
  7    2778.387    7.660   65.206
  8    3240.322    7.741   62.817
  9    3703.373    7.698   64.889
 10    4165.683    7.713   63.805
 11    4631.259    7.805   63.717
 12    5091.795    7.550   64.711
 13    5555.396    7.900   63.076
 14    6020.782    7.618   64.761
 15    6481.164    7.728   63.223
 16    6945.300    7.744   64.584
 17    7405.657    7.603   63.888
 18    7872.215    7.945   63.889
 19    8335.291    7.497   64.090
 20    8796.471    7.872   64.039

>>
```

APPENDIX 1

MATLAB function *edm.m*

```
function edm
%
% function to simulate the processing of EDM measurements through
% a Kalman filter.

%=========================================================================
% Function:  edm
%
% Author:
%  Rod Deakin,
%  School of Mathematical and Geospatial Sciences, RMIT University,
%  GPO Box 2476V, MELBOURNE VIC 3001
%  AUSTRALIA
%  email: rod.deakin@rmit.edu.au
%
% Date:
%  Version 1.0  15 April 2006
%
% Remarks:
%  This function simulates the processing of EDM measurements
%  through a Kalman filter.
%
% References:
%  [1] Deakin, R.E., 2006, The Kalman Filter and Surveying Applications,
%        A paper for presentation at the Regional Surveying Conference,
%        Mildura, 23-25 June 2006.
%
% Arrays:
%  B         - Design matrix, dimensions (m,n)
%  corrn     - array of corrections to State vector, dimensions (n,epochs)
%  H         - coefficient matrix of System Driving Noise, dimensions (n,u)
%  I         - Identity matrix, dimensions (n,n)
%  K         - Gain matrix, dimensions (n,m)
%  meas      - vector of measurements, dimensions (epochs,1)
%  noise     - vector of normally distributed measurement "noise" with
%              a mean of zero and a standard deviation of sigma (epochs,1)
%  Q         - cofactor matrix of measurements, dimensions (n,n)
%  Qmm       - cofactor matrix of Secondary Model, dimensions (n,n)
%  Qww       - cofactor matrix of System Driving Noise, dimensions (n,n)
%  Qxx       - cofactor matrix of State vector, dimensions (n,n)
%  std_xhat  - vector of standard deviations of the filtered State
%  T         - Transition matrix, dimensions (n,n)
%  U         - cofactor update matrix, dimensions (n,n)
%  xhat      - array containing the State vector at each epoch,
%              dimensions (n,epochs)
%
% Variables:
%  const     - a constant value
%  epochs    - number of epochs
%  i,j,k     - integer counters
%  m         - number of measurements at particular epoch
%  n         - number of parameters in the State vector 'xhat'
%  u         - number of parameters in the System Driving Noise vector
%  std_noise - st dev of measurement noise
%=========================================================================

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% define the Kalman filter matrices: %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
n = 1;  % number of elements in State vector
m = 1;  % number of measurements at each epoch
```

```
u = 1;  % number of elements in System Driving Noise vector
% Set the total number of measurement epochs
epochs = 250;
% Initialize the State vector.
xhat = zeros(n,epochs);
% Initialize the corrections to State vector.
corrn = zeros(n,epochs);
% Set the State Transition matrix T
T = eye(n);
T(1,1) = 1.0;
% Set the cofactor matrix of the System Driving Noise Qww
Qww = zeros(u,u);
Qww(1,1) = 0.0;
% Set the coefficient matrix of System Driving Noise H
H = zeros(n,u);
H(1,1) = 1.0;
% Compute cofactor matrix of Dynamic Model by propagation of variances
Qmm = H*Qww*H';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate some measurements with noise %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Set the state of the random number generator.  This will ensure that
% you get the same result as Ref [1].  For repeated operations the
% following two lines should be commented out or removed.
S = [2.0493e9; 1.2454e9];
randn('state',S);
% Set the st. dev. of measurement noise, initialise the measurement vector
% then generate the noise
std_noise = 0.010;
meas      = zeros(epochs,1);
noise     = std_noise .* randn(epochs,1);
% Generate the measurement vector
const = 355.42;
for k = 1:epochs
  meas(k) = const + noise(k);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Set Cofactor matrices and State vector to initial values %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Set the cofactor matrix of the measurements Q
Q = zeros(m,m);
Q(1,1)  = std_noise^2;
% Set the State cofactor matrix Qxx.  This will be the filtered
% State cofactor matrix at epoch t2.
Qxx = zeros(n,n);
Qxx(1,1) = std_noise^2;
% Set the starting estimate of the State vector for Epoch 1
% This will be the filtered Sate at epoch t2
xhat(1,1) = meas(1,1);
% Initialize the Gain matrix
K = zeros(n,m);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Start the Kalman Filter at epoch t2 %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for k = 2:epochs
  % Compute the predicted State vector.
  xhat(:,k) = T*xhat(:,k-1);
  % Compute the predicted State cofactor matrix
  Qxx = T*Qxx*T' + Qmm;
  % Set the elements of the design matrix B
  B = zeros(m,n);
  B(1,1) = -1;
  % Compute the Gain matrix K
  K = Qxx*B'/(Q + B*Qxx*B');
  % Compute corrections to predicted State
```

```
  corrn(:,k) = K*(-meas(k) - B*xhat(:,k));
  % Compute the filtered State vector
  xhat(:,k) = xhat(:,k) + corrn(:,k);
  % Compute the cofactor update matrix
  I = eye(n);
  U = I - K*B;
  % Compute the filtered State cofactor matrix
  Qxx = U*Qxx;
  std_xhat(k) = sqrt(Qxx(1,1));
  % Print the State vector, corrections and filtered State cofactor matrix
  fprintf('\n\nEpoch = %3d, measurement = %8.4f',k,meas(k));
  fprintf('\nFiltered State  Corrn         Filtered State cofactor matrix Qxx');
  for i=1:u
      fprintf('\n  %9.4f %10.4f      ',xhat(i,k),corrn(i,k));
      for j=1:u
          fprintf('%12.9f',Qxx(i,j));
      end
  end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%
%% End of Kalman filter %%
%%%%%%%%%%%%%%%%%%%%%%%%%%

% Figure 1: create a figure window with two plots
figure(1);
clf;
subplot(2,1,1);
hold on;
grid on;
box on;
plot(meas,'k.');
plot(xhat,'k');
title('Kalman Filter estimate');
ylabel('state [m]');

subplot(2,1,2);
hold on;
grid on;
box on;
plot(std_xhat,'k');
title('Standard deviation of estimate');
xlabel('epoch [s]');
ylabel('std [m]');
```

APPENDIX 2

MATLAB function *kalship3.m*

```
function kalship3(filename)
%
% KALSHIP   This function implements a Kalman Filter to estimate the POSITION and
%   VELOCITY of a ship in a channel.
%   The observations are distances to the ship at regular time intervals
%   from three known locations.
%
% e.g., kalship3('c:\projects\kalman\exercise\Kalshipdata3.dat');

%=========================================================================
% Function:  kalship3
%
% Author:
%  Rod Deakin,
%  School of Mathematical & Geospatial Sciences, RMIT University,
%  GPO Box 2476V, MELBOURNE VIC 3001
%  AUSTRALIA
%  email: rod.deakin@rmit.edu.au
%
% Date:
%  Version 1.0  17 April 2006
%
% Remarks:
%  This function implements a Kalman Filter to estimate the POSITION and
%  VELOCITY of a ship moving at a near constant velocity in a channel.
%  The observations are horizontal distances to three fixed stations
%  taken at regular time intervals.
%  The observations are contained in an ASCII data file, an example of which
%  is shown below (c:\projects\kalman\Kalshipdata3.dat)
%
%--------start of data file c:\projects\kalman\exercise\Kalshipdata3.dat---------
%
% Kalman Filter Navigation problem
%
% A ship is moving at a constant velocity with distance measurements
% every 60 seconds to three shore-based beacons A, B and C.
%
% Data
%       Distances to beacons
% Epoch   A       B       C
%   1   4249.7  7768.6  7721.1
%   2   3876.1  7321.4  7288.5
%   3   3518.4  6872.2  6857.6
%   :       :       :       :
%   :       :       :       :
%   :       :       :       :
%   19  5366.4  1959.6  2819.7
%   20  5785.0  2182.8  3023.5
%----------------------end of data file--------------------------------
%
% References:
%  [1] Deakin, R.E., 2006, The Kalman Filter and Surveying Applications,
%        A paper for presentation at the Regional Surveying Conference,
%        Mildura, 23-25 June 2006.
%
% Arrays:
%  B         - Design matrix, dimensions (m,n)
%  Distance  - vector of distances from start at each epoch
%  Ef        - vector of east coords of fixed stations (beacons)
%  epoch     - vector of epoch numbers (integers)
%  H         - coefficient matrix of System Driving Noise, dimensions (n,u)
```

```
%  Heading   - vector of headings (bearings from north) at each epoch
%  I         - Identity matrix, dimensions (n,n)
%  K         - Gain matrix, dimension (n,m)
%  meas      - array of measured distances from the ship to the beacons (metres)
%  Nf        - vector of north coords of fixed stations (beacons)
%  Q         - cofactor matrix of measurements, dimension is (m,m)
%  Qmm       - cofactor matrix of Dynamic Model, dimensions (n,n)
%  Qww       - cofactor matrix of System Driving Noise, dimensions (u,u)
%  Qxx       - cofactor matrix of State vector, dimensions (n,n)
%  T         - Transition matrix, dimensions (n,n)
%  U         - cofactor Update matrix, dimensions (n,n)
%  Velocity  - vector of velocities at each epoch
%  xhat      - State vector, dimension is (n,epochs)
%
% Variables
%  cj,dj     - distance  coefficients for observation j
%  d         - distance
%  d2        - distance squared
%  dt        - time difference in seconds between epochs
%  dE,dN     - difference in east and north coordinates
%  j,k       - integer counters
%  m         - number of measurements at each epoch
%  n         - number of parameters in the State vector 'xhat'
%  s2_aE     - variance of east  acceleration (m^2/s^4)
%  s2_aN     - variance of north acceleration (m^2/s^4)
%  s2_Dist   - variance of observed distance(m^2)
%  u         - number of elements in System Driving Noise vector
%  vE,vN     - east and north velocities
%========================================================================

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Set the coordinates of the fixed stations, the beacons A, B and C. %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Ef = [10000 13880 15550];
Nf = [10000 11250 7160];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Read in the data from a file using function textread. %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Read the epoch number into array 'epoch' and the distances to the
% three beacons into matrix 'meas'.  Distances to beacons A, B and C
% are in matrix 'meas' in columns 1, 2 and 3
[epoch,meas(:,1),meas(:,2),meas(:,3)]=textread(filename,'%d %f %f
%f','headerlines',8);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Set estimates of variances of measurements and System Driving Noise %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Set the estimate of the variance of measurements.
s2_meas = 1.0;  % variance of distance (m2)
% Set the estimates of variances of System Driving Noise.
s2_aE = 0.017;
s2_aN = 0.017;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Set the Epoch update rate. %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
dt = 60;  % update rate in seconds of time

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Define the Kalman filter matrices. %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
n = 4;  % number of elements in State vector
m = 3;  % number of measurements at each epoch
u = 2;  % number of elements in System Driving Noise vector

% The State vector has 4 elements; the east and north cordinates E,N, and
% the east and north velocities vE,vN.
```

```
% There are three measurements at each epoch (to beacons A, B and C).
% There are two elements in the system driving noise vector (accel. East
% and accel. North).

% Determine the number of measurement epochs.
epochs = length(epoch);
% Initialize the State vector.
xhat = zeros(n,epochs);
% Initialize the corrections to State vector.
corrn = zeros(n,epochs);
% Set the State Transition matrix T
T = eye(n);
T(1,3) = dt;
T(2,4) = dt;
% Set the cofactor matrix of the measurements Q
Q = zeros(m,m);
for m = 1:m
  Q(m,m) = s2_meas;  % diagonal elements of Q = variance of distance
end
% Set the cofactor matrix of the System Driving Noise Qww
Qww = zeros(u,u);
Qww(1,1) = s2_aE;
Qww(2,2) = s2_aN;
% Set the coefficient matrix of System Driving Noise H
H = zeros(n,u);
H(1,1) = (dt^2)/2;
H(2,2) = H(1,1);
H(3,1) = dt;
H(4,2) = H(3,1);
% Compute cofactor matrix of Dynamic Model by propagation of variances
Qmm = H*Qww*H';
% Set the starting estimate of the State vector.
% This will be the filtered Sate at epoch t2.
E1 = 7875.0;
N1 = 6319.392;
vE = 7;
vN = 3;

% Epoch t1
xhat(1,1) = E1;
xhat(2,1) = N1;
xhat(3,1) = vE;
xhat(4,1) = vN;
% Set the State cofactor matrix Qxx.
% This will be the filtered State cofactor matrix at epoch t2.
Qxx = zeros(n,n);
Qxx(1,1) = 20.0;
Qxx(2,2) = 20.0;
Qxx(3,3) = 0.5;
Qxx(4,4) = 0.5;
% Initialize the Gain matrix
K  = zeros(n,m);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Start the Kalman Filter at epoch t2. %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for k = 2:epochs
  % Compute the predicted State vector.
  xhat(:,k) = T*xhat(:,k-1);
  % Compute the predicted State cofactor matrix
  Qxx = T*Qxx*T' + Qmm;
  % Compute the Design matrix B
  B  = zeros(m,n);
  f  = zeros(m,1);
  for j = 1:m
    % coordinate differences Pi to Pk
    dE = xhat(1,k)-Ef(j);
    dN = xhat(2,k)-Nf(j);
```

```
    % Computed distance Pi to Pk
    d2 = dE^2 + dN^2;
    d  = sqrt(d2);
    % Distance coefficients
    cj =  dN/d;
    dj =  dE/d;
    % Set the elements of B
    B(j,1) = -dj;
    B(j,2) = -cj;
    % Compute the numeric terms (computed - observed)
    f(j,1) = d - meas(k,j);
  end
  % Compute the Gain matrix K
  K = Qxx*B'/(Q + B*Qxx*B');
  % Compute corrections to predicted State
  corrn(:,k) = K*f;
  % Compute the filtered State vector
  xhat(:,k) = xhat(:,k) + corrn(:,k);
  % Compute the cofactor update matrix
  I = eye(n);
  U = I - K*B;
  % Compute the filtered State cofactor matrix
  Qxx = U*Qxx;
  % Print the State vector, corrections and filtered State cofactor matrix
  fprintf('\n\nepoch = %3d',k);
  fprintf('\nFiltered State  Corrns        Filtered State cofactor matrix Qxx');
  for i=1:n
    fprintf('\n  %9.3f %10.3f      ',xhat(i,k),corrn(i,k));
    for j=1:n
      fprintf('%10.6f',Qxx(i,j));
    end
  end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% End of Kalman filter. %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%

fprintf('\n\n');

% Compute distance from start and Velocity
% and get corrections to east, north, velocity east and
% velocity north from the array of corrections.

Distance = zeros(epochs,1);
Velocity = zeros(epochs,1);
Heading  = zeros(epochs,1);
corrnE   = zeros(epochs,1);
corrnN   = zeros(epochs,1);
corrnvE  = zeros(epochs,1);
corrnvN  = zeros(epochs,1);

d2r = 180/pi; % degree to radian conversion factor d2r = 57.29577951..
for k=1:epochs
    if k >1
        % coordinate differences P(k-1) to P(k)
        dE = xhat(1,k)-xhat(1,k-1);
        dN = xhat(2,k)-xhat(2,k-1);
        % Computed distance P(k-1) to P(k)
        d = sqrt(dE^2 + dN^2);
        Distance(k,1) = Distance(k-1,1) + d;
    end
    angle = atan2(xhat(3,k),xhat(4,k))*d2r;
    if angle < 0
        angle = angle + 360;
    end
    Velocity(k,1) = sqrt(xhat(3,k)^2 + xhat(4,k)^2);
    Heading(k,1)  = angle;
    corrnE(k,1)   = corrn(1,k);
```

```
    corrnN(k,1)   = corrn(2,k);
    corrnvE(k,1)  = corrn(3,k);
    corrnvN(k,1)  = corrn(4,k);
end

% Print the filtered values

fprintf('Filtered Values');
fprintf('\nEpoch  Distance  Velocity  Heading');
for k=1:epochs
  fprintf('\n%3d %11.3f %8.3f %8.3f',k,Distance(k,1),Velocity(k,1),Heading(k,1));
end

fprintf('\n\n');

%--------------------------
% create plot of velocities
%--------------------------
figure(1);
clf;
grid on;
plot(epoch,Velocity,'bo-');
title('Kalman Filter Velocities');
xlabel('Epoch');
ylabel('Velocity (m/s)');

%-----------------------------------------
% create plot of East coordinate corrections
%-----------------------------------------
figure(2);
clf;
grid on;
plot(epoch,corrnE,'bo-');
title('Kalman Filter corrections to East coords');
xlabel('Epoch');
ylabel('Corrections (m)');

%-----------------------------------------
% create plot of North coordinate corrections
%-----------------------------------------
figure(3);
clf;
grid on;
plot(epoch,corrnN,'bo-');
title('Kalman Filter corrections to North coords');
xlabel('Epoch');
ylabel('Corrections (m)');
```

REFERENCES

Brown, R.G. and Hwang, P.Y.C., 1992, *Introduction to Random Signals and Applied Kalman Filtering,* 2nd ed, John Wiley & Sons, Inc.

Cross, P.A., 1987, 'Kalman filtering and its application to offshore position-fixing', *The Hydrographic Journal,* No. 44, pp. 19-25, April 1987.

Cross, P.A., 1992, *Advanced least squares applied to position fixing,* Working Paper No. 6, Department of Land Surveying, University of East London, 205 pages, November 1992. (Originally published by North East London Polytechnic in 1983)

Kalman, R.E., 1960, 'A new approach to linear filtering and prediction problems', *Transactions of the ASME–Journal of Basic Engineering,* Series 82D, pp. 35-45, March 1960.

Krakiwsky, E.J., 1975, *A Synthesis of Recent Advances in the Method of Least Squares,* Lecture Notes No. 42, 1992 reprint, Department of Surveying Engineering, University of New Brunswick, Fredrickton, Canada.

Sorenson, H.W., 1970, 'Least squares estimation: from Gauss to Kalman', *IEEE Spectrum,* Vol. 7, pp. 63-68, July 1970.