Project assignment in
TTK4115 Linear System Theory

Discrete Kalman Filter Applied to a Ship Autopilot

Norges teknisk-naturvitenskapelige universitet
Institutt for teknisk kybernetikk
Trondheim

Version 1.0, Nov 2003 JS and TAJ
Version 1.1, Oct 2004, JS
Version 1.2, Oct 2005, JH
Version 1.3, Nov 2007, JM
Version 1.4, Oct 2008, AAE
Version 1.5, Oct 2009, JR
Version 1.6, Sept 2010, DB
Version 1.7, Sept 2011, DB
Version 1.8, Oct 2016, KG

1. Practical information

- The assignment is mandatory and will account for 20% of the total grade.
- The assignment shall be carried out in groups of two (or three) students.
- The project is to be carried out *independently* by each group.
- The amount of time to complete the assignment is estimated at 30 hours. Each group will get 12 hours of guidance.
- The time schedule can be found on the group registration page (the webpage of the course).
- The file `Boat_files.zip` can be downloaded from It's Learning.

2. Report

- The final report is due in week 47, the exact time will be posted on It's Learning. The report should include
  - The deductions, reasoning and calculations for each exercise. Make sure that it is easy to follow how you obtained your result. It is not enough to list the MATLAB functions you used; give a short description of what the MATLAB function do and explain why you used them.
  - Simulation results for each exercise. Make sure that your plots clearly illustrate your obtained results. Moreover, describe (in words) what can be observed in the plots.
  - A short discussion of the results. Discuss what implications your results have. In addition, suggest alternatives that may improve the obtained results.
  - Attachments, copies of the m-files and Simulink diagrams. The code and diagrams should be easy to understand. Add comments if necessary.
- It should be easy to look up where your answer to each part of the assignment can be found, so clearly mark out to which part of the assignment each section in the report belongs.
- Make sure that you give an answer to each question in the assignment. Your answer should be concise. However, if your answer is not in the report, you do not get points for it.

3. Purpose of the assignment

The objectives of the assignment can be summarized as:

- To partly model and simulate a continuous system influenced by stochastic signals.
- To use basic identification techniques on parameters that are not explicitly given.
- To use basic control theory to design a simple autopilot.
- To implement a discrete Kalman filter for wave filtering and estimation of disturbances using MATLAB and Simulink.

4. Background material

4.1. Coordinate systems

4.1.1. Reference frames

In navigation several reference frames are used. By a reference frame we mean a coordinate system, or frame, a vector is described relatively to. We only consider two coordinate systems in this assignment, "NED" and "BODY".

- NED is a coordinate system in which the x-axis point to the north, the y-axis points east and the z-axis points towards the center of the earth (down).
- BODY is a coordinate system where the x-axis is along the longitudinal axis of the ship (from aft to fore). The y-axis is along the transversal axis (to starboard) and the z-axis along the normal axis (from top to bottom).

Figure 1 illustrates the BODY and NED reference frames. In Figure 1 the heading ψ is depicted. ψ is the compass measurement.

4.1.2. Transforming vectors between different reference frames

This section is not intended as a complete description of the subject, but a short introduction to how transformations are carried out. Given a vector $\mathbf{v}^b$, where the superscript b denotes that $\mathbf{v}$ is described in the BODY reference frame. We can

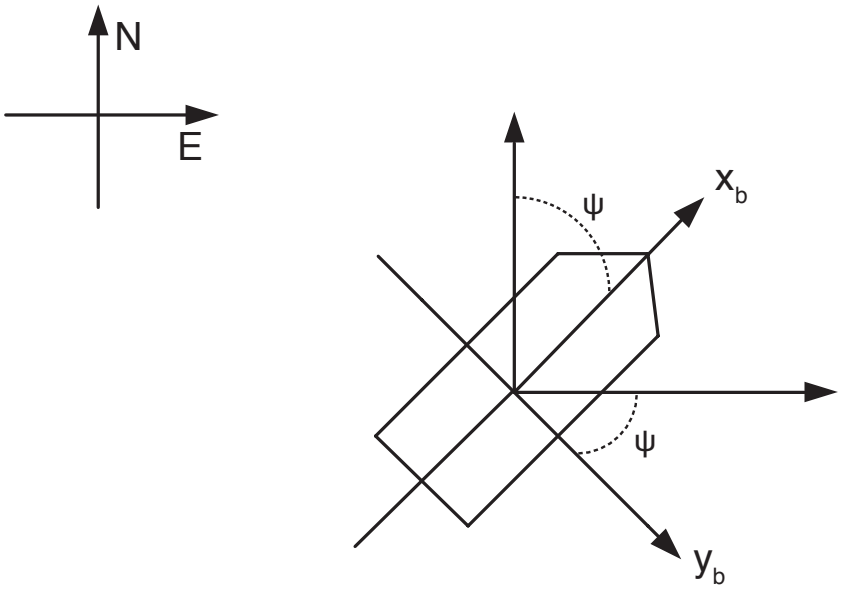

Figure 1: BODY and NED reference frames.

transform this vector such that it is described in the NED reference frame. In the horizontal plane this becomes:

$$\mathbf{v}^n = \mathbf{R}_b^n(\psi)\mathbf{v}^b \tag{1}$$

$$\mathbf{R}_b^n = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix} \tag{2}$$

The transformation can also be reversed:

$$\mathbf{v}^b = \mathbf{R}_n^b(\psi)\mathbf{v}^n = \mathbf{R}_b^n(\psi)^T\mathbf{v}^n \tag{3}$$

since $\mathbf{R}^T = \mathbf{R}^{-1}$ when $\mathbf{R}$ is a rotation matrix. To clarify we will give an example:

Example 1 (Rotation of a vector) *Let $\psi = \frac{\pi}{4}$ and $\mathbf{v}^b = [0\ 1]^T$. Carrying out the calculation as described in (1) yields a vector $\mathbf{v}^n = [-\frac{1}{\sqrt{2}}\ \frac{1}{\sqrt{2}}]^T$. Figure 2 depicts the two vectors. The transformation from NED to BODY can be carried out to see that $\mathbf{v}^b = [0\ 1]^T$.*

4.2. System description

We shall look at a model of a ship, influenced by both current and waves. The three following sections present the necessary background material to complete the project. We will model the system as if the waves only affect the heading of the ship and try to find an estimate of the course angle without considering the wave disturbance.

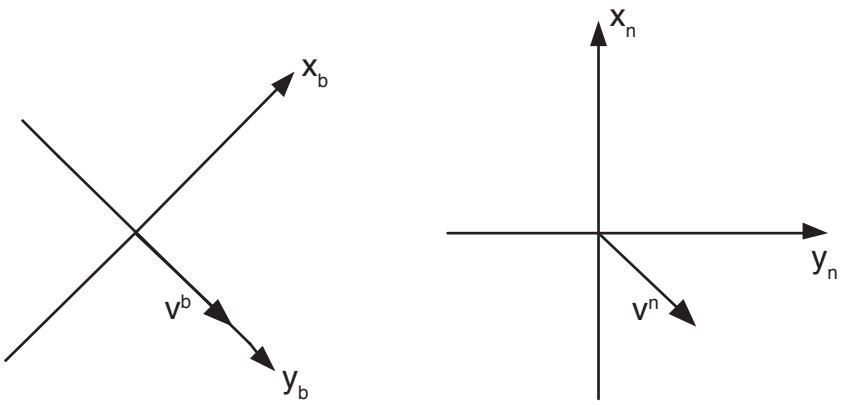

Figure 2: The same vector in different frames.

4.2.1. The ship

A nonlinear dynamical model of a ship can be represented as:

$$\dot{\boldsymbol{\eta}} = \mathbf{R}(\psi)\boldsymbol{\nu} \tag{4}$$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{D}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau} + \mathbf{w} \tag{5}$$

where

- $\mathbf{M}$ - System inertia matrix.
- $\mathbf{C}$ - Coriolis-centripetal matrix.
- $\mathbf{D}$ - Damping matrix.
- $\boldsymbol{\tau}$ - Vector of control inputs.
- $\mathbf{w}$ - Vector of environmental disturbances.
- $\boldsymbol{\eta}$ - NED positions $[x, y, \psi]$. Where x is the position in the north-direction, y is the position in the east-direction, and ψ is the angle between the north direction and the x_b axis. ψ is positive clockwise.
- $\boldsymbol{\nu}$ - BODY velocities $[u, v, r]$. Where u is the velocity in the x-direction, v the velocity in the y-direction and r is rotation velocity about the z-axis.

Assuming that the speed is low such that some of the nonlinear terms are negligible, the equations are reduced to:

$$\dot{\boldsymbol{\eta}} = \mathbf{R}(\psi)\boldsymbol{\nu} \tag{6}$$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \mathbf{C}\boldsymbol{\nu} + \mathbf{D}\boldsymbol{\nu} = \boldsymbol{\tau} + \mathbf{w} \tag{7}$$

Note that $\mathbf{w}$ is actually given in the NED reference frame, so we assume that we only have small changes in the heading of the ship in this model. We now have a linear equation for the BODY velocities. But equation (6) is still non-linear. This will be simplified later. Assume that the forward speed u is constant, i.e. $u = u_0$. We then simplify the model by only considering the sway(v)-yaw(r) dynamics. Also letting $\boldsymbol{\tau} = \mathbf{B}\delta$, where δ is the rudder angle relative to the BODY frame, we get the equation:

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \mathbf{N}(u_0)\boldsymbol{\nu} = \mathbf{B}\delta + \mathbf{w}_{waves} + \mathbf{w}_{current} \tag{8}$$

where $\boldsymbol{\nu} = [v\ r]^T$ and $\mathbf{N}(u_0) = \mathbf{C}(u_0) + \mathbf{D}(u_0)$.

If we are only interested in ψ, we get the following equation for $\boldsymbol{\eta}$:

$$\dot{\boldsymbol{\eta}} = \dot{\psi} = r \tag{9}$$

4.2.2. Waves

The waves are considered to be high-frequency disturbances. The response to the waves can be modelled as a damped harmonic oscillator:

$$\begin{bmatrix} \dot{x}_{w1} \\ \dot{x}_{w2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & -2\lambda\omega_0 \end{bmatrix} \begin{bmatrix} x_{w1} \\ x_{w2} \end{bmatrix} + \begin{bmatrix} 0 \\ K_w \end{bmatrix} w_w \tag{10}$$

$$y_w = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_{w1} \\ x_{w2} \end{bmatrix} \tag{11}$$

where w_w is a zero mean white noise process with unity variance. This representation of the waves corresponds to a spectral factorization of the wave spectrum. How to add this to the model is shown in Section 4.2.4.

4.2.3. Current

The current is a slowly varying disturbance. We will assume that the only effect of the current is a rudder angle bias b. This bias is modelled as:

$$\dot{b} = w_b \tag{12}$$

where w_b is Gaussian white noise. Note also here that this assumes only small deviation from the reference heading of the ship.

4.2.4. The complete system

In the model of the system (not the actual process) that will be used in the rest of the assignment the state vector is given by: $[\xi_w\ \psi_w\ \psi\ r\ b]^T$, where

- ψ - Is the average heading, i.e. without wave disturbance.
- ψ_w - Is a high-frequency component due to the wave disturbance.
- $\dot{\xi}_w = \psi_w$
- r - As described above.
- b - Bias to the rudder angle.

See Figure 3 for an illustration of ψ and ψ_w.

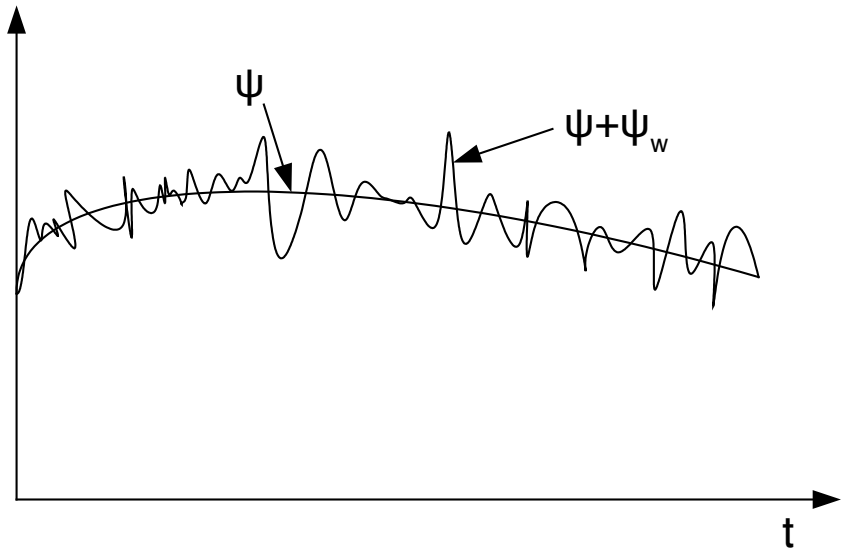

Figure 3: Average heading and high frequency wave disturbance.

The model which will be used can be stated as:

$$\begin{aligned}
\dot{\xi}_w &= \psi_w && \text{(13a)}\\
\dot{\psi}_w &= -\omega_0^2 \xi_w - 2\lambda\omega_0\psi_w + K_w w_w && \text{(13b)}\\
\dot{\psi} &= r && \text{(13c)}\\
\dot{r} &= -\frac{1}{T}r + \frac{K}{T}(\delta - b) && \text{(13d)}\\
\dot{b} &= w_b && \text{(13e)}\\
y &= \psi + \psi_w + v && \text{(13f)}
\end{aligned}$$

where y is the measured heading (compass measurement). w_b, w_w, and v are white noise processes. Also note that the model in Section 4.2.1 is simplified to a 1st

order Nomoto-approximation in equations (13c)-(13d), with Nomoto time and gain constants, T and K. The Nomoto model is common for modelling yaw in marine systems.

Clearly, the system can be written as:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u + \mathbf{E}\mathbf{w}, \quad y = \mathbf{C}\mathbf{x} + v \tag{14}$$

with $\mathbf{x} = [\xi_w \ \psi_w \ \psi \ r \ b]^T$, $u = \delta$ and $\mathbf{w} = [w_w \ w_b]^T$. The purpose of this model is to estimate the course angle without the wave disturbance. Thus, we model the ship as a system not affected by waves and include the disturbance only in the measurement. Further, the current only affects the rudder angle in the model. This is of course not the case for an actual ship, but it simplifies the Kalman filter design.

5. Assignment

Unzip the files from `Boat_files.zip` to a desired directory. The files in `Boat_files.zip` are:

- `ship.mdl`: Simulink file representing a cargo ship.
- `wave.mat`: The wave disturbance, ψ_w.
- `Sfunctionshell.m`: A partially complete discrete Kalman filter.

Open the mdl file `ship.mdl`. You will see a block which represents the actual ship (a cargo ship). The output of the model is *compass*, x, and y. The ship is assumed to have a constant forward speed $u = u_0$. The input to the ship is the rudder set-point, which is given in degrees. Clicking on the ship model opens a dialog box in which currents, waves and measurement noise can be turned on and off. Note that the rudder angle is constrained to ± 45 degrees.

5.1. Identification of the boat parameters

a) Assume that there are no disturbances. Calculate the transfer function from δ to ψ, $H(s)$, parameterized by T and K.

b) Turn off all disturbances in the model. This corresponds to identifying the parameters in smooth weather conditions. We want to identify the boat parameters T and K. Apply a sine input with amplitude 1 and frequency $\omega_1 = 0.005$ (rad/s). Then apply a sine input with amplitude 1 and frequency $\omega_2 = 0.05$ (rad/s). The amplitude of the sine waves on the output equals $|H(j\omega_1)|$ and $|H(j\omega_2)|$, respectively. This gives us two equations with two unknowns.

c) Repeat part b) with waves and measurement noise turned on. This corresponds to identifying the parameters in rough weather conditions. Is it possible to get good estimates of the boat parameters in this case?

d) Apply a step input of 1 degree to the rudder at $t = 0$ and compare the step response of the model with the response of the ship. Is the model a good approximation?

Include plots for the results for the above problems in your report. *In the rest of the problems the measurement noise shall be turned on.*

5.2. Identification of wave spectrum model

a) Load the `wave.mat` file. The second row in the resulting matrix `psi_w` contains the influence the waves have on the compass measurement, that is ψ_w. NB! ψ_w is given in degrees in the file. The first row is the time instants the elements of ψ_w are applied to the system. Find an estimate of the Power Spectral Density (PSD) function of ψ_w, $S_{\psi_w}(\omega)$. The sampling frequency is 10 Hz. Use the MATLAB function `[pxx,f] = pwelch(x,window,noverlap,nfft,fs)`. (Hint: type `doc pwelch` for help.) Use a window size of 4096. You do not need to specify the other parameters (`noverlap`, `nfft`). Note: if the input `fs` is given in Hz, then the units of the outputs `pxx` and `f` are power per Hz and Hz, respectively. The scaling factors $\frac{1}{2\pi}$ and 2π need to be applied to convert the outputs to the required units power s/rad and rad/s, respectively.

b) Find an analytical expression for the transfer function of the wave response model (from w_w to ψ_w). Also find an analytical expression for the Power Spectral Density function of ψ_w, that is $P_{\psi_w}(\omega)$.

c) Find ω_0 from the estimated $S_{\psi_w}(\omega)$ in part a).

d) To have a complete model for the wave response we need to identify the damping factor λ. Define $K_w = 2\lambda\omega_0\sigma$ where σ^2 is the peak value of $P_{\psi_w}(\omega)$. Find λ by fitting the $P_{\psi_w}(\omega)$ to the estimate of the PSD, $S_{\psi_w}(\omega)$. Use trial and error. Alternatively, you may use curve-fitting methods in MATLAB (Hint: `doc lsqcurvefit`). Include plot for comparison manner.

5.3. Control system design

In this section we want to design an autopilot for the ship. That is, we want to be able to give a desired course angle ψ_r, and get the ship to follow this course. (Note that the ship model in the simulation only holds for small deviations in

compass value ψ, so leave ψ inside a boundary of ± 35 deg. Use $\psi_r = 30$ in all the simulations done in this part of the report. Make sure to address this constraint in your simulations in part b), c), and d).)

a) Design a PD controller, $H_{pd}(s) = K_{pd}\frac{1+T_d s}{1+T_f s}$, based on the transfer function from δ to ψ without disturbances. Let the ω_c and the phase margin of the open loop system, $H_{pd}(s) \cdot H_{ship}(s)$, be approximately 0.10 (rad/s) and 50 degrees, respectively. Choose the derivative time constant, T_d, such that it cancels the transfer function time constant.

b) Simulate the system without disturbances (only measurement noise). Does the autopilot work? Include plots of compass course and rudder input.

c) Simulate the system with a current disturbance (and without wave disturbance). Does the autopilot work in this case? Include plots of compass course and rudder input.

d) Simulate the system with wave disturbance (and without the current disturbance), does the system work satisfactory? Include plots of compass course and rudder input.

5.4. Observability

In this assignment, include all relevant information in order to justify your answer. Yes or no is not an appropriate answer.

a) Find the matrices A, B, C and E in equation (14).

b) Is the system observable without disturbances?

c) Is the system observable with the current disturbance?

d) Is the system observable with the wave disturbance?

e) Is the system observable with both current and wave disturbance?

5.5. Discrete Kalman filter

In this section we shall implement a discrete Kalman filter to estimate the bias b, the heading ψ and the high-frequency wave induced motion on the heading ψ_w. ψ_w must be removed from the control loop to avoid wear and tear on the actuator system. That is, we do not want the rudder to compensate for ψ_w. Hence, we use only the estimated ψ in the control law. This is referred to as *wave filtering*.

(Note that the ship model in the simulation only holds for small deviations in compass value ψ, so leave ψ inside a boundary of ± 35 deg. Make sure to address this constraint in your simulations in parts d) and e).)

a) Discretize the model found in problem 5.4 part a) using exact discretization. Use a sample frequency of 10 Hz.

b) Find an estimate of the variance of the measurement noise. (Hint: MATLAB-function `var`.)

c) Now let

$$\begin{aligned} \mathbf{w} &= [w_w \; w_b]^T, \; E\{\mathbf{w}\mathbf{w}^T\} = \mathbf{Q} = \begin{bmatrix} 30 & 0 \\ 0 & 10^{-6} \end{bmatrix}, \qquad (15) \\ \mathbf{P}_0^- &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0.013 & 0 & 0 & 0 \\ 0 & 0 & \pi^2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2.5 \cdot 10^{-4} \end{bmatrix}, \; \hat{\mathbf{x}}_0^- = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{aligned}$$

where $\mathbf{w}$ is the process noise, $\mathbf{Q}$ is the process noise covariance, $\mathbf{P}_0^-$ is the initial a priori estimate error covariance and $\hat{\mathbf{x}}_0^-$ is the initial a priori state estimate. Since the process is sampled, $E\{v^2\} = R$ equals the measurement noise variance found in part b) divided by the sample interval. There are many ways to implement the discrete Kalman filter. Two recommended ways are:

- Writing an s-function, which can be called from Simulink. You can implement the s-function either as an m-file (using the MATLAB programming language) or write it in the C programming language. See Appendix A.
- Write a normal Matlab function within the `Matlab function` Simulink block. See Appendix B

Traditionally, s-functions are better suited for dynamic systems since they facilitate initialization, termination and updates at a given frequency. The `Matlab function` block is designed for static systems, so it does not have the same overhead as s-functions. This makes `Matlab function` seem easier, but it also require some additional work to set up the initialization of the Kalman filter. Hint: Let the compass measurement and the rudder input be input to the Kalman filter function and the output be the a posteriori estimate of ψ and b. Also put a zero order hold on the compass measurement and the rudder command, since the process and controller are continuous. Use the same sampling frequency as above. Use the update expression (4.2.11) in Brown&Hwang.

d) Make a feed forward from the estimated bias such that the bias is cancelled. Use ψ_r=30. Simulate the system with the current disturbance. Does the autopilot have a better performance than the equivalent simulation in problem 5.3 part c)? Include plots of measured compass course, rudder input and estimated bias.

e) Use the wave filtered ψ instead of the measured heading in the autopilot. Simulate the system with wave and current disturbance. Does the autopilot have a better performance than the equivalent simulation in problem 5.3 part d)? Include plots of both measured and filtered compass course, rudder input and estimated bias. Also include plots of actual wave influence and estimated wave influence.

A. S-function hints

We start by looking at the structure of an s-function, and we will only consider the elements which are of relevance to this assignment. The s-function is called by Simulink with parameters x, u, t and $flag$, where x is the current state, u is the input, t is the time and $flag$ decides which part of the s-function that shall be executed. The execution flow for the s-function in this assignment is depicted in figure 4.

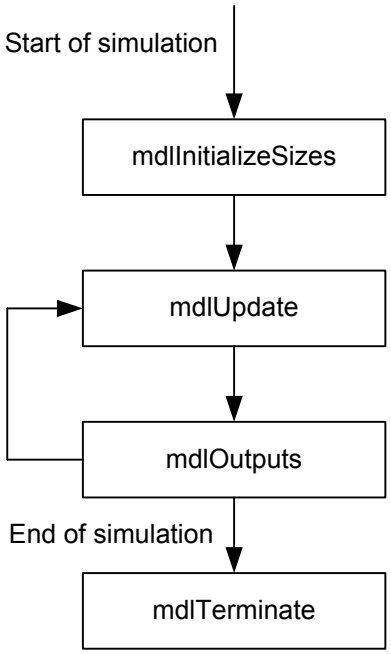

Figure 4: Execution flow of an S-function.

- `mdlInitializeSizes`: Simulink calls this function only at the start of the simulation. Here the system `struct` is initialized. That is, the number of discrete/continuous states, inputs, outputs are defined. Sample times and initial conditions are also specified.

- `mdlUpdate`: The discrete states are updated when Simulink calls this function. We can look at the update function as:

$$\mathbf{x}(t + \Delta t) = f(\mathbf{x}(t), \mathbf{u}(t), t) \tag{16}$$

  where Δt is the sampling interval. Example: For a linear time invariant system we have $\mathbf{x}(t+\Delta t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$. In the s-function syntax we get: `sys=A*x+B*u;`

- `mdlOutputs`: When Simulink calls this function, the output of the s-function is calculated. We can look at the output function as:

$$\mathbf{y}(t) = h(\mathbf{x}(t), \mathbf{u}(t), t) \tag{17}$$

  Example: For a linear time invariant system this the function reduces to: $\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)$. In the s-function syntax we get: `sys=C*x+D*u;`

- `mdlTerminate`: Simulink only calls this function when the simulation terminates.

It also possible to define additional parameters as input to the s-function.

Clearly, we can write the discrete Kalman filter with use of the update and output function. There are many ways of implementing the filter. We will describe two approaches below.

1. One can define all the matrices, that is, $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{E}$, $\mathbf{Q}$, $\mathbf{R}$, and $\mathbf{P}$ as global variables. Then let the a priori and the a posteriori estimates be state variables. Define the output to be the a posteriori estimates and the input to be the compass measurement and the rudder input. However, usually we want to avoid use of global variables.

2. Let a `struct` which contains the matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{E}$, $\mathbf{Q}$, $\mathbf{R}$, $\mathbf{P}_\mathbf{0}^-$, and $\hat{\mathbf{x}}_0^-$ be a parameter sent to the s-function. The matrices can then be used in all the sub-routines of the s-function. Let the state vector be given by the a priori estimates, the a posteriori estimates and the elements of the error covariance matrix, that is:

$$\mathbf{x} = [\hat{\mathbf{x}}^-,\ \hat{\mathbf{x}},\ P_{11}, \ldots P_{15},\ P_{21}, \ldots, P_{55}]^T \tag{18}$$

   See Appendix D for convertions of P^- from a matrix to a vector and vice versa.

B. `Matlab function` hints

Like any other function, the `Matlab function` block does not remember its variables from one call to the next. This is unfortunate, since we would like to keep some of the variables in the Kalman filter for the next iteration. An elegant solution to this is to declare the variables *persistent*. Persistent variables are retained in memory from call to call, and are local to the function they are declared in, much like static variables in a C function. Listing 1 shows how to declare the variables `init_flag` and `acc_sum` persistent.

Since the `Matlab function` block has no notion of whether it is called for the first time, or for the n'th time, we have to program this notion. One way to do this is to have a flag that is set when the function is called, and then check the status of this flag to determine if the function has been called before. If the function has not been called before, we should do an initialization. This concept is illustrated below, where a function takes an input and adds it to the accumulating sum. Calling this function with input 5, 1 and 3 would return 5, 6 and 9.

Listing 1: Illustration of `persistent` and initialization

```
function output = dummy_func( in )
persistent acc_sum init_flag

if isempty(init_flag)
    init_flag = 1;
    acc_sum = 0;
end
acc_sum = A + in;
output = acc_sum;
end
```

It is important to use the `isempty` function, since this allows to check a variable that potentially has not yet been initialized.

The final modification that is needed for the `Matlab function` block to work in this setting is to avoid an *algebraic loop*. When Simulink starts up, it tries to determine the states from the initial values. In this case, the Kalman filter depends on the control input, but due to the feedback the control input also depends on the output from the Kalman filter. Therefore, Simulink is unable to determine some of the initial values and blames an *algebraic loop*. To solve this, we have to add a `Memory` block to the outputs of the Kalman filter, which will allow initialization of the variables used in the feedback.

C. PSD function hints

In newer versions of MATLAB, i.e. MATLAB 7.0, the command `help psd` does not work. This is because `psd` has been replaced by `spectrum.welch`. But the command `psd` still works. Below we list the help-text for using this command in Power Spectral Density estimates.

```
>> help psd

 PSD Power Spectral Density estimate.
    Pxx = PSD(X,NFFT,Fs,WINDOW) estimates the Power Spectral Density of
    a discrete-time signal vector X using Welch's averaged, modified
    periodogram method.

    X is divided into overlapping sections, each of which is detrended
    (according to the detrending flag, if specified), then windowed by
    the WINDOW parameter, then zero-padded to length NFFT.  The magnitude
    squared of the length NFFT DFTs of the sections are averaged to form
    Pxx.  Pxx is length NFFT/2+1 for NFFT even, (NFFT+1)/2 for NFFT odd,
    or NFFT if the signal X is complex.  If you specify a scalar for
    WINDOW, a Hanning window of that length is used.  Fs is the sampling
    frequency which doesn't affect the spectrum estimate but is used
    for scaling the X-axis of the plots.

    [Pxx,F] = PSD(X,NFFT,Fs,WINDOW,NOVERLAP) returns a vector of frequen-
    cies the same size as Pxx at which the PSD is estimated, and overlaps
    the sections of X by NOVERLAP samples.

    [Pxx, Pxxc, F] = PSD(X,NFFT,Fs,WINDOW,NOVERLAP,P) where P is a scalar
    between 0 and 1, returns the P*100% confidence interval for Pxx.

    PSD(X,...,DFLAG), where DFLAG can be 'linear', 'mean' or 'none',
    specifies a detrending mode for the prewindowed sections of X.
    DFLAG can take the place of any parameter in the parameter list
    (besides X) as long as it is last, e.g. PSD(X,'mean');

    PSD with no output arguments plots the PSD in the current figure window,
    with confidence intervals if you provide the P parameter.

    The default values for the parameters are NFFT = 256 (or LENGTH(X),
    whichever is smaller), NOVERLAP = 0, WINDOW = HANNING(NFFT), Fs = 2,
```

```
    P = .95, and DFLAG = 'none'.  You can obtain a default parameter by
    leaving it off or inserting an empty matrix [], e.g. PSD(X,[],10000).

    NOTE: For Welch's method implementation which scales by the sampling
          frequency, 1/Fs, see PWELCH.

    See also PWELCH, CSD, COHERE, TFE.
    ETFE, SPA, and ARX in the System Identification Toolbox.
```

Use the default values for the parameters in the exercise, i.e

```
[Sx,Fx]=psd(psi_wsig,[],samp_freq);
```

where `psi_wsig` is the signals of wave influence on the compass measurement, in rad.

D. Vector/Matrix conversions

You may find it useful in your S-Function to convert between a matrix and a vector to represent the covariances. Take a look at the following example:

```
>> v = 0:8
v =
     0     1     2     3     4     5     6     7     8
>> vmat = reshape(v,sqrt(length(v)),sqrt(length(v)))
vmat =
     0     3     6
     1     4     7
     2     5     8
>> vvec = vmat(:)'
vvec =
     0     1     2     3     4     5     6     7     8
```