APPENDIX A

MATLAB SOFTWARE

The accompanying CD contains MATLAB functions and scripts for implementing the Kalman filter and demonstrating its use. The ASCII file `READMEFIRST.TXT` in the root directory should be read before starting to use any of the software. It describes the current contents and directory structure of the files on the CD.

A.1 NOTICE

This software is intended for demonstration and instructional purposes only. The authors and publishers make no warranty of any kind, expressed or implied, that these routines meet any standards of mercantibility for commercial purposes. These routines should not be used as-is for any purpose or application that may result in loss or injury, and the publishers and authors shall not be liable in any event for incidental or consequential damages in connection with or arising out of the furnishing, performance, or use of these programs.

A.2 GENERAL SYSTEM REQUIREMENTS

The MATLAB scripts on the CD are designed for MATLAB environments. Information on MATLAB can be obtained from

Kalman Filtering: Theory and Practice using MATLAB®, Third Edition. By Mohinder S. Grewal and Angus P. Andrews

The Math Works, Inc.
3 Apple Hill Drive
Natick, MA 01760 USA

Tel: 508-647-7000
Fax: 508-647-7101
E-mail: info@mathworks.com
Web: www.mathworks.com

You may also use the MATLAB editor to modify these scripts as needed. Comments in the listings contain additional information on the calling sequences and array dimensions.

A.3 CD DIRECTORY STRUCTURE

The CD directories are organized by the chapters in which the supporting concepts are presented. The ASCII file `WHATSUP.DOC` in the root directory describes any changes made in the directory structure or software after printing. It should also be read before starting to use any of the software.

There are some common functions that are not repeated in every directory containing a calling script, so users are encouraged to copy all MATLAB scripts to a common subdirectory of the MATLAB "work" directory on their own computers.

A.4 MATLAB SOFTWARE FOR CHAPTER 2

The directory CHAPTER2 contains `expm1.m`, referred to in Chapter 2 and Appendix B Section B.4.4.2.

A.5 MATLAB SOFTWARE FOR CHAPTER 3

`VanLoan.m` implements Van Loan's method for computing Kalman filter model parameter matrices (i.e., in discrete time) from Kalman–Bucy filter model parameter matrices (i.e., in continuous time).

A.6 MATLAB SOFTWARE FOR CHAPTER 4

The directory CHAPTER4 contains software implementing the algorithms defined in Chapter 4. See the file `WHATSUP.DOC` in the root directory for descriptions of any changes.

`demo1.m` is a MATLAB script for demonstrating the effects of process noise and observations of the probability distribution of a single state variable as a function of time. The plots show the evolution of the probability density function while observations are made at discrete times.

`exam43.m` is a MATLAB script for demonstrating Example 4.4 in MATLAB.

`exam44.m` is a MATLAB script for demonstrating Example 4.5 in MATLAB.

`obsup.m` is a MATLAB script implementation of the Kalman filter observational update, including the state update and the covariance update (Riccati equation).

`timeup.m` is a MATLAB script implementation of the Kalman filter temporal update, including the state update and the covariance update (Riccati equation).

A.7 MATLAB SOFTWARE FOR CHAPTER 5

`sim3pass.m` generates a trajectory for a scalar linear time-invariant model driven by simulated noise for measurement noise and dynamic disturbance noise, applies a three-pass fixed-interval smoother to the resulting measurement sequence, and plots the resulting estimates along with the true (simulated) values and computed variances of estimation uncertainty.

`FiniteFIS.m` computes and plots the "end effects" of a finite-interval smoother.

`FLSmovies.m` creates two short video clips of the smoothing improvement ratio $P[s]/P[f]$ as a function of QR/H^2 and FR/H^2 as the lag time parameter Δt_{lag} H^2/R varies from frame to frame from 10^{-6} to about 0.06. (It runs a long time.)

`RTSvsKF.m` is a MATLAB script for demonstrating the solutions to an estimation problem using

1. Kalman filtering, which uses only data up to the time of the estimate, and
2. Rauch–Tung–Striebel smoothing, which uses all the data.

A MATLAB implementation of a Rauch–Tung–Striebel smoother is included in the script, along with the corresponding Kalman filter implementation.

`BMFLS.m` implements the complete Biswas-Mahalanabis fixed-lag smoother (BMFLS) through the transient phase from the first measurement and through the transition to steady-state state augmentation and beyond.

`FPSperformance.m` computes the expected performance of fixed-point smoothing and plots the results.

A.8 MATLAB SOFTWARE FOR CHAPTER 6

`shootout.m` provides a demonstration of the relative fidelity of nine different ways to perform the covariance correction on Example 6.2.

To test how different solution methods perform as conditioning worsens, the observational update is performed for $10^{-9}\ \varepsilon^{2/3} \leq \delta \leq 10^{9}\ \varepsilon^{2/3}$ using nine different implementation methods:

1. The conventional Kalman filter, as published by R. E. Kalman;
2. Swerling inverse implementation, published by P. Swerling before the Kalman filter;

3. Joseph-stabilized implementation as given by P. D. Joseph;
4. Joseph-stabilized implementation as modified by G. J. Bierman;
5. Joseph-stabilized implementation as modified by T. W. DeVries;
6. The Potter algorithm (due to J. E. Potter);
7. The Carlson "triangular" algorithm (N. A. Carlson);
8. The Bierman *UD* algorithm (G. J. Bierman); and
9. The closed-form solution for this particular problem.

The first, second, and last methods are implemented within the m-file `shootout.m`. The others are implemented in m-files listed below.

The results are plotted as the RMS error in the computed value of P relative to the closed-form solution. In order that all results, including failed results, can be plotted, the value NaN (not a number) is interpreted as an underflow and set to zero, and the value Inf is interpreted as the result of a divide-by-zero and set to 10^4. This demonstration should show that, for this particular problem, the accuracies of the Carlson and Bierman implementations degrade more gracefully than the others as $\delta \rightarrow \varepsilon$. This might encourage the use of the Carlson and Bierman methods for applications with suspected roundoff problems, although it does not necessarily demonstrate the superiority of these methods for all applications:

`bierman.m` performs the Bierman *UD* implementation of the Kalman filter measurement update and

`carlson.m` performs the Carlson *fast triangular* implementation of the Kalman filter measurement update.

`joseph.m` performs the Joseph-stabilized implementation of the Kalman filter measurement update, as proposed by Bucy and Joseph [56];

`josephb.m` performs the Joseph-stabilized implementation of the Kalman filter measurement update, as modified by G. J. Bierman;

`josephdv.m` performs the Joseph-stabilized implementation of the Kalman filter measurement update, as modified by T. W. DeVries;

`potter.m` performs the Potter *square-root* implementation of the Kalman filter measurement update; and

`utchol.m` performs upper triangular Cholesky factorization for initializing the Carlson fast triangular implementation of the Kalman filter measurement update.

A.9 MATLAB SOFTWARE FOR CHAPTER 7

`ExamIEKF.m` performs Monte Carlo analysis of iterated extended Kalman filtering (EKF) on a nonlinear sensor problem.

`exam7.5.m` implements a discrete linearized Kalman filter.

`UTsimplex` implements the lowest-complexity unscented transform (UT) using "simplex" sampling. `UTsimplexDemo.m` is a MATLAB demonstration of its application.

`UTscaled.m` implements the scaled UT. `UTscaledDemo.m` demonstrates its use.

`UKFtempUD.m` implements the nonlinear Kalman filter temporal update using the UT.

A.10 MATLAB SOFTWARE FOR CHAPTER 8

`KFvsSKF.m` includes MATLAB implementations of the Schmidt–Kalman filter and Kalman filter for a common problem, implements both, and plots the results for comparison.

`thornton.m` implements the Thornton temporal update compatible with the Bierman observational update using modified Cholesky factors of P.

`Schmidt.m` performs the schmidt temporal update cumpatible with the Bierman observational update using modified Cholesky factors of P.

`InnovAnalysis.m` demonstrates innovations analysis on a linear stochastic process with nine different Kalman filters: one with the same model parameters as the simulated process and eight with the four model parameters Φ, H, Q, and R scaled up and down by a factor of 2.

A.11 MATLAB SOFTWARE FOR CHAPTER 9

`SchulerDemo.m` simulates and plots Schuler oscillations due to initial inertial navigation system (INS) errors.

`FNNmat.m` computes the dynamic coefficient matrix F_{NN} for INS navigation errors.

`Example9pt1.m` generates the plot used in Example 9.1.

`FSSmat.m` computes the dynamic coefficient matrix for inertial sensor errors.

`Example9pt3.m` simulates the nine-state INS navigation error dynamics.

`FSSQSmat.m` computes the dynamic coefficient submatrix F_{SS} and the covariance matrix QS for the INS sensor error model.

`FNSmat.m` computes the dynamic coefficient submatrix F_{NS} linking sensor errors to navigation errors.

`FreeInertial.m` calculates the propagation of INS navigation error covariances over a 10-h period, starting with zero navigation errors and the sensor error parameters shown in Table 9.2.

`Fig8TrackSim` generates values of vehicle dynamic state as it traverses a 1.5-km figure-8 test track at 90 kph.

`Fig8TrackSim2` computes additional values of vehicle dynamic state variables as it traverses a 1.5-km figure-8 test track at 90 kph.

`Example0901.m` demonstrates the effects of GNSS satellite motion on navigation performance.

`Example0906.m` demonstrates the influence of satellite geometry on GNSS navigation performance.

`YUMAdata.m` generates the actual GPS ephemeris information for Wednesday, March 8, 2006, at 10:48 A.M.

`HSatSim.m` uses the YUMA data to generate pseudorange measurement sensitivity matrices for GNSS receivers at a given latitude, longitude, and time.

`Example0906.m` requests four directions to GNSS satellites, then computes and plots navigation performance in terms of RMS uncertainties in position (three components) and clock errors (two parameters) versus time for 1 min.

`GNSSshootoutNCE.m` compares side-by-side the simulated performance of three different host vehicle dynamic models for GNSS navigation on a figure-8 racetrack and plots the results.

`AltStab.m` compares inertial navigation altitude estimates with and without auxiliary sensor damping.

`CDModelParams.m` computes dynamic model parameters Φ_θ and Q_θ for a critically damped system driven by white noise, given the system steady-state variances.

`HINS67` computes the 67-state measurement sensitivity matrix for an INS navigation sensor with 15 measurement variables for three components each of position, velocity, acceleration, attitude, and attitude rate.

`Fig8FreeInert.m` simulates navigation performance on a figure-8 track for a free-inertial INS without aiding (except for vertical channel stabilization).

`Fig8HVINS.m` calculates navigation performance on a figure-8 track for an INS without sensor aiding (except for vertical channel stabilization), but using a host vehicle dynamic model to statistically constrain navigation errors. This host vehicle dynamic model is "tuned" to the statistical dynamic conditions on the figure-8 test track. The same host vehicle model is used in the following performance simulators as well.

`Fig8GNSSonly.m` calculates expected performance using only GNSS navigation on a figure-8 track.

`Fig8GNSSINS.m` calculates and plots RMS performance statistics of a tightly coupled strapdown INS/GNSS integrated system which estimates and compensates for INS navigation errors and sensor errors.

A.12 OTHER SOURCES OF SOFTWARE

A.12.1 MATLAB Controls Toolbox

Available from The Mathworks, Controls Toolbox includes MATLAB routines for numerical solution of the algebraic Riccati equation for the Kalman filtering

problem for linear time-invariant systems. These essentially provide the steady-state Kalman gain (Wiener gain).

A.12.2 Software for Kalman Filter Implementation

There are several sources of good up-to-date software specifically designed to address the numerical stability issues in Kalman filtering. Scientific software libraries and workstation environments for the design of control and signal processing systems typically use the more robust implementation methods available. In addition, as a noncommercial source of algorithms for Kalman filtering, the documentation and source codes of the collected algorithms from the Transactions on Mathematical Software (TOMS) of the Association for Computing Machinery are available at moderate cost on electronic media. The TOMS collection contains several routines designed to address the numerical stability issues related to Kalman filter implementation, and these are often revised to correct deficiencies discovered by users.

Many sources for the unscented Kalman filter (UKF) can be found by searching the World Wide Web.

A.12.3 Utilities for Monte Carlo Simulation

The TOMS collection also contains several routines designed for pseudorandom number generation with good statistical properties. In addition, most reputable libraries contain good pseudorandom number generators, and many compilers include them as built-in functions. There are also several books (e.g., [221] or [128]) that come with the appropriate code on machine-readable media.

A.12.4 GNSS and Inertial Navigation Software

There are several sources of software for the design and analysis of inertial and GNSS navigation systems. These include:

1. Inertial Navigation System (INS) Toolbox for MATLAB
2. Navigation System Integration and Kalman Filter Toolbox for MATLAB
3. Satellite Navigation (SatNav) ToolBox for MATLAB

produced by GPSoft (www.gpsoftnav.com) and marketed through

Navtech Seminars & GPS Supply
www.navtechgps.com
Tel: 1-703-256-8900
Fax: 1-703-256-8988

Other commercial sources can be located by searching the World Wide Web.