Guide to MATLAB Simulation Software

Paul D. Groves

This document presents a guide to the MATLAB[1] simulation software that accompanies the book *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*, 2nd edition. It is intended as an overview rather than a detailed description. The functions and scripts themselves contain headers summarizing their functions and defining all inputs and outputs. The code is fully commented and refers to the equations in the book that it implements. Variable names are either descriptive or based on the notation used in the book. Greek letters are referred to by name and subscripts and superscripts are both preceded by underscores. For symbols in the book that have both subscripts and superscripts, the subscripts are always given first.

Section M.1 introduces the software, describes its main capabilities and explains its structure. Section M.2 describes the demonstration scripts that configure and run the software. Section M.3 describes the master navigation functions that simulate stand-alone inertial navigation, stand-alone GNSS, or integrated INS/GNSS navigation. Section M.4 summarizes the general navigation functions that form part of the simulation suite. Section M.5 summarizes a number of tool functions that read, write, and manipulate data files and plot results. Section M.6 describes the motion profiles that provide the true position, velocity, and attitude data for the simulation. Finally, Section M.7 lists the output error file format.

M.1 Introduction

The software simulates inertial navigation, GNSS, and their integration. Its comprises:

- A simple IMU model;
- A simple GNSS constellation and error model;
- Inertial navigation equations implemented in ECI, ECEF, and local-navigation frames;
- A single-epoch iterative least-squares GNSS positioning algorithm (ECEF frame);
- An EKF-based GNSS positioning algorithm (ECEF frame);
- A loosely coupled INS/GNSS integration algorithm;
- A tightly coupled INS/GNSS integration algorithm;
- Associated functions and utilities;
- Demonstration scripts and motion profiles.

It may be configured to simulate stand-alone inertial navigation, stand-alone GNSS, or integrated INS/GNSS navigation.

[1] MATLAB is a registered trademark of The Math Works, Inc.

The purpose of this software is educational. It is intended to illustrate some of the main points of the book it accompanies. It is not designed to provide a rigorous simulation of real-world navigation technology. Simplifications made include the following:

- All IMU and GNSS error sources are treating as constants or white noise sequences.
- The Kalman filter-based estimation algorithms used for GNSS positioning and INS/GNSS integration are relatively basic: the measurement noise models are very simple and there is no innovation filtering.
- The GNSS constellation model assumes circular orbits with satellites equally distributed amongst six orbital planes.
- There is no GNSS receiver simulation; receiver errors are modeled simply as a white noise sequence. This does not account for dynamics response lags in the tracking loops.
- There is no simulation of GNSS signal blockage, attenuation, interference, jamming, reflection, or diffraction. Multipath interference can only be simulated by increasing the receiver noise standard deviation.

The software comprises a series of files with the extension '.m' in the directory *MATLAB Software* of the CD, which may be copied onto a local drive. These are known as M-files and each one contains a function or script with the same name as the file. Once the directory containing the files has been added to the MATLAB path, the function or script may be called direct from the MATLAB command window. M-files can also be run using Octave. However, the software has not been tested in the Octave environment.

The software comprises four types of M-file: demonstration scripts, master navigation simulation functions, general navigation functions, and tool functions. These are described in Sections M.2 to M.5, respectively. The CD also contains a number of motion profile files with the extension '.csv'. These describe the true position, velocity, and attitude of the navigation system at each epoch of the simulation and may be read into MATLAB, Octave, Microsoft Excel, and many other programs. Motion profiles are described in Section M.6 and are also used by the simulation software to output the navigation solution as a function of time. Navigation error files, described in Section M.7, may also be output.

Each simulation is controlled by a script which sets the configuration parameters, reads in a truth motion profile, calls the master navigation simulation function, writes output files, and calls graph-plotting functions. A script is called simply by typing its name into the MATLAB command window and entering return. All variables directly referred to in the script remain in the MATLAB workspace after the script is called. A number of demonstration scripts are provided (see Section M.2) and you are encouraged to create your own scripts by editing them.

M.2 Demonstration Scripts

Table M.1 summarizes the demonstration scripts supplied with the software. Each script begins with a short header describing its function. This is followed by the configuration parameters in the following order:

- Input and output file names;
- Initialization errors (where applicable);
- IMU model parameters (where applicable);
- GNSS model parameters (where applicable);
- Kalman filter parameters (where applicable);
- Seeding of the random number generator.

In your own scripts, configuration parameters may be entered in any order. The following functions are then called (in order);

- Read_profile – reads in the truth motion profile;
- The master navigation simulation function (see Section M.3);
- Plot_profile – plots the truth motion profile on screen;

- Plot_errors – plots the position, velocity, and attitude errors on screen;
- Write_profile – writes the navigation solution at each epoch to a file;
- Write_errors – writes a file containing the position, velocity, and attitude errors.

Table M.1 List of Demonstration Scripts

Name	*Navigation System*	*Truth Motion Profile*	*Notes*
GNSS_Demo_1	Stand-alone GNSS with least-squares positioning	Profile_1 (car motion with two 90° turns)	
GNSS_Demo_2	Stand-alone GNSS with EKF positioning	Profile_1 (car motion with two 90° turns)	
GNSS_Demo_3	Stand-alone GNSS with EKF positioning	Profile_0 (stationary)	
GNSS_Demo_4	Stand-alone GNSS with EKF positioning	Profile_2 (car motion: two 90° turns, curve & halt)	
GNSS_Demo_5	Stand-alone GNSS with EKF positioning	Profile_3 (aircraft motion: two 45° turns & 500m climb)	
GNSS_Demo_6	Stand-alone GNSS with EKF positioning	Profile_4 (boat motion with two 45° turns; sea state 3)	
Inertial_Demo_1ECEF	Stand-alone inertial (ECEF-frame equations)	Profile_1 (car motion with two 90° turns)	Tactical-grade IMU
Inertial_Demo_1ECI	Stand-alone inertial (ECI-frame equations)	Profile_1 (car motion with two 90° turns)	Tactical-grade IMU
Inertial_Demo_1NED	Stand-alone inertial (local-navigation-frame equations)	Profile_1 (car motion with two 90° turns)	Tactical-grade IMU
Inertial_Demo_2	Stand-alone inertial (local-navigation-frame equations)	Profile_1 (car motion with two 90° turns)	Aviation-grade IMU
Inertial_Demo_3	Stand-alone inertial (local-navigation-frame equations)	Profile_1 (car motion with two 90° turns)	Consumer-grade IMU
Inertial_Demo_4	Stand-alone inertial (local-navigation-frame equations)	Profile_0 (stationary)	Tactical-grade IMU
Inertial_Demo_5	Stand-alone inertial (local-navigation-frame equations)	Profile_2 (car motion: two 90° turns, curve & halt)	Tactical-grade IMU
Inertial_Demo_6	Stand-alone inertial (local-navigation-frame equations)	Profile_3 (aircraft motion: two 45° turns & 500m climb)	Aviation-grade IMU
Inertial_Demo_7	Stand-alone inertial (local-navigation-frame equations)	Profile_4 (boat motion with two 45° turns; sea state 3)	Tactical-grade IMU
INS_GNSS_Demo_1	Tightly coupled INS/GNSS	Profile_1 (car motion with two 90° turns)	Tactical-grade IMU
INS_GNSS_Demo_2	Tightly coupled INS/GNSS	Profile_0 (stationary)	Tactical-grade IMU
INS_GNSS_Demo_3	Loosely coupled INS/GNSS	Profile_1 (car motion with two 90° turns)	Tactical-grade IMU
INS_GNSS_Demo_4	Loosely coupled INS/GNSS	Profile_0 (stationary)	Tactical-grade IMU
INS_GNSS_Demo_5	Tightly coupled INS/GNSS	Profile_1 (car motion with two 90° turns)	Aviation-grade IMU
INS_GNSS_Demo_6	Tightly coupled INS/GNSS	Profile_0 (stationary)	Aviation-grade IMU
INS_GNSS_Demo_7	Tightly coupled INS/GNSS	Profile_1 (car motion with two 90° turns)	Consumer-grade IMU
INS_GNSS_Demo_8	Tightly coupled INS/GNSS	Profile_0 (stationary)	Consumer-grade IMU
INS_GNSS_Demo_9	Tightly coupled INS/GNSS	Profile_2 (car motion: two 90° turns, curve & halt)	Tactical-grade IMU

Name	*Navigation System*	*Truth Motion Profile*	*Notes*
INS_GNSS_Demo_10	Tightly coupled INS/GNSS	Profile_3 (aircraft motion: two 45° turns & 500m climb)	Tactical-grade IMU
INS_GNSS_Demo_11	Tightly coupled INS/GNSS	Profile_4 (boat motion with two 45° turns; sea state 3)	Tactical-grade IMU
INS_GNSS_Demo_12	Tightly coupled INS/GNSS	Profile_3 (aircraft motion: two 45° turns & 500m climb)	Aviation-grade IMU

M.3 Master Navigation Simulation Functions

There are seven master navigation simulation functions:

- GNSS_Kalman_Filter – Stand-alone GNSS with EKF positioning using an ECEF frame;
- GNSS_Least_Squares – Stand-alone GNSS with single-epoch least-squares positioning using an ECEF frame;
- Inertial_navigation_ECEF – Stand-alone inertial navigation with ECEF-frame navigation equations;
- Inertial_navigation_ECI – Stand-alone inertial navigation with ECI-frame navigation equations;
- Inertial_navigation_NED – Stand-alone inertial navigation with local-navigation-frame navigation equations;
- Loosely_coupled_INS_GNSS – Integrated INS/GNSS with ECEF-frame inertial navigation equations, single-epoch least-squares GNSS positioning, and loosely coupled INS/GNSS integration using an ECEF frame;
- Tightly_coupled_INS_GNSS – Integrated INS/GNSS with ECEF-frame inertial navigation equations and tightly coupled INS/GNSS integration using an ECEF frame.

Each function initializes the navigation system, simulates it on subsequent epochs, and calculates the navigation errors. The INS simulation is run at the same interval as the truth motion profile, while the GNSS simulation interval is specified in the GNSS model configuration. The integration algorithm is run on the same epochs as the GNSS model, feeding back corrections to the inertial navigation solution every time and updating estimates of the accelerometer and gyro biases, which are used to correct the IMU measurements entering the inertial navigation equations.

Some of the following information is input, depending on which simulation is running:

- in_profile – array containing truth motion profile;
- no_epochs – number of epochs of the truth motion profile;
- initialization errors – structure used for truth plus error initialization of the inertial navigation solution (GNSS position and velocity is used where available);
- IMU_errors – structure containing IMU model configuration parameters;
- GNSS_config – structure containing GNSS model configuration parameters
- GNSS_KF_config – structure containing GNSS EKF configuration parameters;
- LC_KF_config – structure containing loosely coupled Kalman filter configuration parameters;
- TC_KF_config – structure containing tightly coupled EKF configuration parameters.

Some of the following information is output, depending on which simulation is running:

- out_profile – array containing the INS, GNSS, or integrated navigation solution at each epoch;
- out_errors – array containing the navigation solution errors at each epoch;
- out_IMU_bias_est – array containing accelerometer and gyro bias estimates from the integration Kalman filter at each epoch;
- out_clock – array containing estimated GNSS receiver clock offset and drift estimates from the least-squares positioning algorithm, GNSS Kalman filter, or tightly coupled Kalman filter at each epoch;

- out_KF_SD – array containing Kalman filter (or EKF) estimation error standard deviations at each epoch.

M.4 General Navigation Functions

The following functions are called by some or all of the master navigation simulation functions, either directly or indirectly:

- Calculate_errors_NED – Calculates the position error, $\delta\mathbf{r}_{eb}^{n}$, velocity error, $\delta\mathbf{v}_{eb}^{n}$, and attitude error, $\delta\boldsymbol{\psi}_{nb}$, all resolved along/about north, east, and down, from the true and estimated curvilinear positions, velocities, and body-to-NED coordinate transformation matrices;
- CTM_to_Euler – Converts a coordinate transformation matrix to the corresponding set of Euler angles;
- ECEF_to_ECI – Converts position, velocity, and attitude from ECEF- to ECI-frame referenced and resolved, i.e. $\mathbf{r}_{eb}^{e}$, $\mathbf{v}_{eb}^{e}$, and $\mathbf{C}_{b}^{e}$ to $\mathbf{r}_{ib}^{i}$, $\mathbf{v}_{ib}^{i}$, and $\mathbf{C}_{b}^{i}$;
- ECEF_to_NED – Converts Cartesian position, $\mathbf{r}_{eb}^{e}$, to curvilinear position, L_b, λ_b, and h_b; velocity resolving axes from ECEF, $\mathbf{v}_{eb}^{e}$, to NED, $\mathbf{v}_{eb}^{n}$; and attitude from ECEF-referenced, $\mathbf{C}_{b}^{e}$, to NED-referenced, $\mathbf{C}_{b}^{n}$;
- ECI_to_ECEF – Converts position, velocity, and attitude from ECI- to ECEF-frame referenced and resolved, i.e. $\mathbf{r}_{ib}^{i}$, $\mathbf{v}_{ib}^{i}$, and $\mathbf{C}_{b}^{i}$ to $\mathbf{r}_{eb}^{e}$, $\mathbf{v}_{eb}^{e}$, and $\mathbf{C}_{b}^{e}$;
- Euler_to_CTM – Converts a set of Euler angles to the corresponding coordinate transformation matrix;
- Generate_GNSS_measurements – Generates a set of GNSS pseudo-range and pseudo-range rate measurements for all satellites above the elevation mask angle from the true user position, $\mathbf{r}_{eb}^{e}$, and velocity, $\mathbf{v}_{eb}^{e}$, and the true satellite positions, $\mathbf{r}_{es}^{e}$, and velocities, $\mathbf{v}_{es}^{e}$, adding constant range bias errors to the pseudo-ranges and tracking noise to both the pseudo-ranges and pseudo-range rates;
- GNSS_KF_Epoch – Implements a complete cycle of the stand-alone GNSS extended Kalman filter, outputting the updated state estimates, $\hat{\mathbf{x}}_{k}^{+}$, and their error covariance matrix, $\mathbf{P}_{k}^{+}$;
- GNSS_LS_position_velocity – Implements two successive iterative least squares algorithms to calculate, firstly, the user position, $\mathbf{r}_{eb}^{e}$, and receiver clock offset, from a single-epoch set of pseudo-range measurements and, secondly the user velocity, $\mathbf{v}_{eb}^{e}$, and receiver clock drift, from a single-epoch set of pseudo-range rate measurements.
- Gravitation_ECI – Calculates the acceleration due to the gravitational force resolved about ECI-frame axes, $\boldsymbol{\gamma}_{ib}^{i}$;
- Gravity_ECEF – Calculates the acceleration due to gravity resolved about ECEF-frame axes, $\mathbf{g}_{b}^{e}$;
- Gravity_NED – Calculates the acceleration due to gravity resolved about north, east, and down, $\mathbf{g}_{b}^{n}$;
- IMU_model – Simulates IMU specific force and angular rate outputs by adding biases, scale-factor and cross-coupling errors, gyro g-dependent errors, noise, and quantization errors to the true specific force, $\mathbf{f}_{ib}^{i}$, and angular rate, $\boldsymbol{\omega}_{ib}^{i}$;
- Initialize_GNSS_biases – Initializes the GNSS pseudo-range biases due to signal-in-space, ionosphere, and troposphere errors, which are assumed to be constant throughout the simulation;

- Initialize_GNSS_KF – Initializes the GNSS stand-alone EKF state estimates from an input single-epoch least-squares solution and initalizes the error covariance matrix, **P**, according to the EKF configuration parameters;
- Initialize_LC_P_matrix – Initializes the loosely coupled INS/GNSS Kalman filter error covariance matrix, **P**, according to the Kalman filter configuration parameters;
- Initialize_NED – Initializes the inertial navigation curvilinear position, L_b, λ_b, and h_b, velocity, $\mathbf{v}_{eb}^n$, and attitude, $\mathbf{C}_b^n$, solution from their true counterparts, adding user-specified initialization errors;
- Initialize_NED_attitude – Initializes the inertial navigation attitude solution, $\mathbf{C}_b^n$, from its true counterpart, adding user-specified initialization errors;
- Initialize_TC_P_matrix – Initializes the tightly coupled INS/GNSS EKF error covariance matrix, **P**, according to the EKF configuration parameters;
- Kinematics_ECEF – Calculates the true specific force, $\mathbf{f}_{ib}^i$, and angular rate, $\boldsymbol{\omega}_{ib}^i$, from the current and previous ECEF-frame velocity, $\mathbf{v}_{eb}^e$, and attitude, $\mathbf{C}_b^e$, the current position, $\mathbf{r}_{eb}^e$, and the time interval, τ_i;
- Kinematics_ECI – Calculates the true specific force, $\mathbf{f}_{ib}^i$, and angular rate, $\boldsymbol{\omega}_{ib}^i$, from the current and previous ECI-frame velocity, $\mathbf{v}_{ib}^i$, and attitude, $\mathbf{C}_b^i$, the current position, $\mathbf{r}_{ib}^i$, and the time interval, τ_i;
- Kinematics_NED – Calculates the true specific force, $\mathbf{f}_{ib}^i$, and angular rate, $\boldsymbol{\omega}_{ib}^i$, from the current and previous NED-resolved velocity, $\mathbf{v}_{eb}^n$, and attitude, $\mathbf{C}_b^n$, the current latitude, L_b, and height, h_b, and the time interval, τ_i;
- LC_KF_Epoch – Implements a complete cycle of the loosely coupled INS/GNSS Kalman filter, correcting the inertial navigation solution, $\mathbf{r}_{eb}^e$, $\mathbf{v}_{eb}^e$, and $\mathbf{C}_b^e$; updating estimates of the acclerometer biases, $\mathbf{b}_a$, and gyro biases, $\mathbf{b}_g$; and outputting the updated state estimation error covariance matrix, $\mathbf{P}_k^+$.
- Nav_equations_ECEF – Implements a complete cycle of precision ECEF-frame navigation equations, updating the position, $\mathbf{r}_{eb}^e$, velocity, $\mathbf{v}_{eb}^e$, and attitude, $\mathbf{C}_b^e$, from the IMU-measured specific force $\mathbf{f}_{ib}^i$, and angular rate, $\boldsymbol{\omega}_{ib}^i$, and the time interval, τ_i;
- Nav_equations_ECI – Implements a complete cycle of precision ECI-frame navigation equations, updating the position, $\mathbf{r}_{ib}^i$, velocity, $\mathbf{v}_{ib}^i$, and attitude, $\mathbf{C}_b^i$, from the IMU-measured specific force $\mathbf{f}_{ib}^i$, and angular rate, $\boldsymbol{\omega}_{ib}^i$, and the time interval, τ_i;
- Nav_equations_NED – Implements a complete cycle of precision local-navigation-frame navigation equations, updating the curvilinear position, L_b, λ_b, and h_b, velocity, $\mathbf{v}_{eb}^n$, and attitude, $\mathbf{C}_b^n$, from the IMU-measured specific force $\mathbf{f}_{ib}^i$, and angular rate, $\boldsymbol{\omega}_{ib}^i$, and the time interval, τ_i;
- NED_to_ECEF – Converts curvilinear position, L_b, λ_b, and h_b, to Cartesian position, $\mathbf{r}_{eb}^e$; velocity resolving axes from NED, $\mathbf{v}_{eb}^n$, to ECEF, $\mathbf{v}_{eb}^e$; and attitude from NED-referenced, $\mathbf{C}_b^n$, to ECEF-referenced, $\mathbf{C}_b^e$;
- pv_ECEF_to_NED – Converts Cartesian position, $\mathbf{r}_{eb}^e$, to curvilinear position, L_b, λ_b, and h_b; and velocity resolving axes from ECEF, $\mathbf{v}_{eb}^e$, to NED, $\mathbf{v}_{eb}^n$;
- pv_NED_to_ECEF – Converts curvilinear position, L_b, λ_b, and h_b, to Cartesian position, $\mathbf{r}_{eb}^e$; and velocity resolving axes from NED, $\mathbf{v}_{eb}^n$, to ECEF, $\mathbf{v}_{eb}^e$;
- Radii_of_curvature – Calculates the meridian radius of curvature, R_N, and transverse radius of curvature, R_E, from the geodetic latitude;

- Satellite_positions_and_velocities – Calculates the true satellite positions, $\mathbf{r}_{es}^{e}$, and velocities, $\mathbf{v}_{es}^{e}$;
- Skew_symmetric – Outputs the skew symmetric matrix of the input, i.e. $[\mathbf{a}\wedge]$;
- TC_KF_Epoch – Implements a complete cycle of the tightly coupled INS/GNSS extended Kalman filter, correcting the inertial navigation solution, $\mathbf{r}_{eb}^{e}$, $\mathbf{v}_{eb}^{e}$, and $\mathbf{C}_{b}^{e}$; updating estimates of the acclerometer biases, $\mathbf{b}_{a}$, gyro biases, $\mathbf{b}_{g}$, and GNSS receiver clock offset and bias; and outputting the updated state estimation error covariance matrix, $\mathbf{P}_{k}^{+}$.

M.5 Tool Functions

The following functions are called from the demonstration scripts only:

- Plot_errors – Plots the position, velocity, and attitude errors in an on-screen window (not tested with Octave);
- Plot_profile – Plots the position displacement, velocity, and attitude in an on-screen window (not tested with Octave);
- Read_profile – Reads a motion profile from a .csv file into an array, checks the file has the correct number of columns, determines the number of epochs and converts degrees to radians;
- Write_errors – Writes the position, velocity, and attitude errors to a .csv file, converting the attitude errors from radians to degrees.
- Write_profile – Writes a motion profile (navigation solution or truth) to a .csv file, converting radians to degrees.

The following functions are used for editing motion profile files and are called directly from the MATLAB command window:

- Adjust_profile_position – Adjusts the position in a motion profile file to make it consistent with the velocity;
- Adjust_profile_velocity – Adjusts the velocity in a motion profile file to make it consistent with the position;
- Interpolate_profile – Interpolates a motion profile file to halve the time interval and adds jitter (if required);
- Smooth_profile_velocity – Smooths the velocity in a motion profile file to compensate for numerical jitter in the input profile to Adjust_profile_velocity;
- Update_curvilinear_position – Updates latitude, longitude, and height using the current and previous velocity (used by Adjust_profile_position);
- Velocity_from_curvilinear – Determines the current velocity from the current and previous curvilinear positions and the previous velocity (used by Adjust_profile_velocity).

Note that one of the adjustment functions should be run after an interpolation (or series thereof) to ensure that the position and velocity are consistent.

M.6 Motion Profiles

Motion profile files provide the position, velocity, and attitude at each epoch of the simulation with one row per epoch. They are comma-separated text files and have the extension .csv, enabling them to be read automatically into many data analysis packages, including MATLAB, Octave, and Microsoft Excel. They can also be displayed in a text editor. Table M.2 specifies the format. Motion profile files are used for both the truth motion profile and the simulated navigation solution. The following motion profiles are included with the simulation software:

- Profile_0 – 60s stationary with respect to the Earth and level, facing north;
- Profile_1 – 60s of manually generated car motion at a speed of 20 m s^{-1} with two 90° turns in opposite directions;
- Profile_2 – 175s of car motion, generated using Spirent SimGen: at an initial speed of 10 m s^{-1} with the following maneuvers: acceleration to 20 m s^{-1}, deceleration to 10 m s^{-1}, 90° turn, acceleration to 20 m s^{-1}, –30° bend; acceleration to 30 m s^{-1}, 30° bend; deceleration to 5 m s^{-1}, –90° turn, halt;
- Profile_3 – 418s of aircraft motion, generated using Spirent SimGen: at a speed of 200 m s^{-1} with two 45° turns in opposite directions and a 500m climb;
- Profile_4 – 300s of boat motion, generated using Spirent SimGen: at a speed of 10 m s^{-1} with two 45° turns in opposite directions and a sea state of 3.

Table M.2 Motion profile file format

Column	*Description*
1	Time, t (s)
2	Geodetic latitude, L_b (°)
3	Longitude, λ_b (°)
4	Geodetic height, h_b (m)
5	North velocity, $v^n_{eb,N}$ (m s^{-1})
6	East velocity, $v^n_{eb,E}$ (m s^{-1})
7	Down velocity, $v^n_{eb,D}$ (m s^{-1})
8	Bank (roll angle of body w.r.t. NED), ϕ_{nb} (°)
9	Elevation (pitch angle of body w.r.t. NED), θ_{nb} (°)
10	Heading (yaw angle of body w.r.t. NED), ψ_{nb} (°)

M.7 Navigation Error File Format

Navigation error files provide the position, velocity, and attitude errors at each epoch of the simulation with one row per epoch. They are comma-separated text files and have the extension .csv, enabling them to be read automatically into many data analysis packages, including MATLAB, Octave, and Microsoft Excel. They can also be displayed in a text editor. Table M.3 specifies the format.

Table M.3 Navigation error file format

Column	*Description*
1	Time, t (s)
2	North position error, $\delta r^n_{eb,N}$ (m)
3	East position error, $\delta r^n_{eb,E}$ (m)
4	Down position error, $\delta r^n_{eb,D}$ (m)
5	North velocity error, $\delta v^n_{eb,N}$ (m s^{-1})
6	East velocity error, $\delta v^n_{eb,E}$ (m s^{-1})
7	Down velocity error, $\delta v^n_{eb,D}$ (m s^{-1})
8	Attitude error about north, $\delta\psi^n_{nb,N}$ (°)
9	Attitude error about east, $\delta\psi^n_{nb,E}$ (°)
10	Attitude error about down, $\delta\psi^n_{nb,D}$, = heading error, $\delta\psi_{nb}$ (°)

Disclaimer of Warranty

The technical descriptions, procedures, and computer programs in this book and the associated CD have been developed with the greatest of care. However, they are provided as is, without warranty of any kind. Artech House, Inc. and the author of the book titled *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems* make no warranties, expressed or implied, that the equations, programs and procedures in this book or its associated appendices, examples, and software are free of error, or are consistent with any particular standard of merchantability, or will meet your requirements for any particular application. They should not be relied upon for solving a problem whose incorrect solution could result in injury to a person or loss of property. Any use of the programs or procedures in such a manner is at the user's own risk. The author and publisher disclaim all liability for direct, incidental, or consequent damages resulting from use of the programs in this book or its associated appendices, examples, and software.

Software License

The MATLAB simulation software accompanying *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*, 2nd edition, is distributed under a modified Berkeley Software Distribution (BSD) license as follows: