
Overview of Kalman Filter Theory and Navigation Applications

Day 4

Michael L. Carroll

Feb 25, 2003

Day 4: General Code Implementation Issues Topics

- Simulation vs. Real-Time Implementation
- Survey of Nav aids
- Filter Maintenance
- Filter Redundancy and Fault Tolerance

Day 4, Segment 1

Simulation vs. Real-Time Implementation

Topics

- Simulation
- Real-Time Implementation

Simulation

- Introduction to Matlab / Simulink
- Covariance Analysis
- Trajectory Simulation

Introduction to Simulink

- Matlab
- Simulink

Matlab

- Matrix-oriented scientific computing tool from Mathworks
 - Everything is a matrix: scalars are 1-by-1 matrices
- Lots of built-in numerical tools for matrix manipulation
- Industry standard

Matlab

```
% Row vector:  
v = [1 2 3]; % Or separate columns with comma: v = [1, 2, 3];  
% Column vector:  
u = [1;2;3]; % Note: Rows separated with semicolon  
% Natural multiplication:  
x=v*u;
```

Matlab

```
%Displaying values:
```

```
v
```

```
u
```

```
x
```

Matlab

```
% Matrices: Combinations of rows and columns:  
A = [1 2 3; 4 5 6; 7 8 9];  
w = A*u;  
% Matrix operators like inverse and transpose:  
B = inv(A);  
C = A';  
D = u*u'; % Symmetric matrix
```

Matlab

- Full-fledged programming language with control constructs and looping
- Data structures
- Many c-like functions
- Extensive plotting

Matlab

- Expensive: >\$2K / license
- Toolboxes and add-ons are extra
- Freeware alternative: Octave (go to www.gnu.org)

Simulink

- Simulation of Continuous and Discrete Systems
- Linear and Non-Linear
- Easy to Use, Drag and Drop, Block Diagram Interface
 - Large collection of block libraries
- Easy integration with external routines, Matlab functions / scripts, S-functions

Simulink

- Hierarchical Nesting of Subsystems
- Can build hybrid discrete / continuous systems
- Integrate inputs and outputs with Matlab Workspace
- Models can be executed in batch (from script or command line) or interactively in GUI

Simulink

- See demo of Model Construction

Covariance Analysis

- Overview
- Various Approaches to Covariance Simulation
- SIMLTIC: **SIM**ulation of **L**inear **T**ime-**I**nvariant **C**ovariance

Covariance Analysis: Overview

- Covariance propagation and covariance update can be accomplished without actually dealing with the state vector at all.

$$(1) \quad P_k(-) = \Phi_k P_{k-1}(+) \Phi_k^T + Q_k$$

$$(2) \quad P_k(+) = [I - K_k H_k] P_k(-)$$

- Of course, the Kalman gain still has to be computed for use in Eq. (2) above, the update equation. The gain is:

$$(3) \quad K_k = P_k(-) H_k^T \left[H_k P_k(-) H_k^T + R_k \right]^{-1}$$

Covariance Analysis: Overview

- As in real-time implementation, the Joseph form of the gain equation is generally preferred, because of its superior numerical properties:

$$(4) \quad P_k(+)= [I - K_k H_k] P_k(-) [I - K_k H_k]^T + K_k R_k K_k^T$$

Covariance Analysis: Overview

- The only items driving the covariance equations are the state dynamics F , the process noise Q , and the measurement noise R
- Thus, covariance analysis can provide good insight into how well balanced the Q and R matrices are
- More importantly, covariance analysis is crucial in exploring what-if scenarios with new measurement sources

Covariance Analysis: Overview

- As an example, suppose you want to evaluate several different GPS receivers as nav aids
- You can use a manufacturer's published 1σ accuracy specifications in your measurement noise matrix R and run several covariance simulations to see what the expected performance is

Covariance Analysis

- Of course, if GPS is your only navaid, this is a no brainer: the receiver with the least measurement noise performs best
- However, if this sensor is just one of several, you might find that some less expensive receiver when thrown into the mix with other nav aids is good enough

Covariance Analysis: Overview

- Other uses: Debugging the filter
- If you save the actually used process noise Q and measurement noise R from a real-time mission (using a maintenance logging facility), then these matrices can be fed into the covariance simulation to see what effect these have on the gains.
- Or to see how the offline gains compare with the real-time gains.

Covariance Analysis: Overview

- Covariance analysis is quite often used to determine proper error budgets
- Error budget: How error sources are allocated to various states and sensors
- E.g., in a nav application, to meet overall system position accuracy requirements, certain combinations of misalignment, instrument bias, scale factor, and inertial noise error, etc., will work when coupled with sensor errors

Covariance Analysis: Overview

- Optimal vs. Suboptimal
- Evaluating finite wordlength effects
- Bottom Line: Covariance analysis essential to filter maintenance
 - If you are planning to modify a Kalman filter at all, you need to run covariance analysis simulations
 - It is very risky not to do so!

Various Approaches to Covariance Simulation

- In the continuous formulation, covariance extrapolation can be accomplished by solving the linear variance equation

$$(5) \quad \dot{P} = FP + PF^T + Q$$

- This can be useful in exploring how a navigation system "coasts" in the absence of measurements
 - It can give you an idea of how fast the solution degrades in free-inertial

Various Approaches to Covariance Simulation

- In the continuous case, we can also eliminate the gain because

$$\lim_{\Delta t \rightarrow 0} \frac{K(t)}{\Delta t} = PH^{\top}R^{-1}$$

- This leads to a single equation, the Matrix Riccati equation, for covariance propagation and update:

$$(6) \quad \dot{P} = FP + PF^{\top} + Q - PH^{\top}R^{-1}HP$$

Various Approaches to Covariance Simulation

- Matrix Riccati Equation reveals essential characteristics of filter

$$\dot{P} = FP + PF^T + Q - PH^T R^{-1}HP$$

- Note that in this equation we increase estimation uncertainty by adding in process noise
- And we decrease estimation uncertainty by the amount of information (R^{-1}) inherent in the measurement

Various Approaches to Covariance Simulation

- See Matlab / Simulink Model

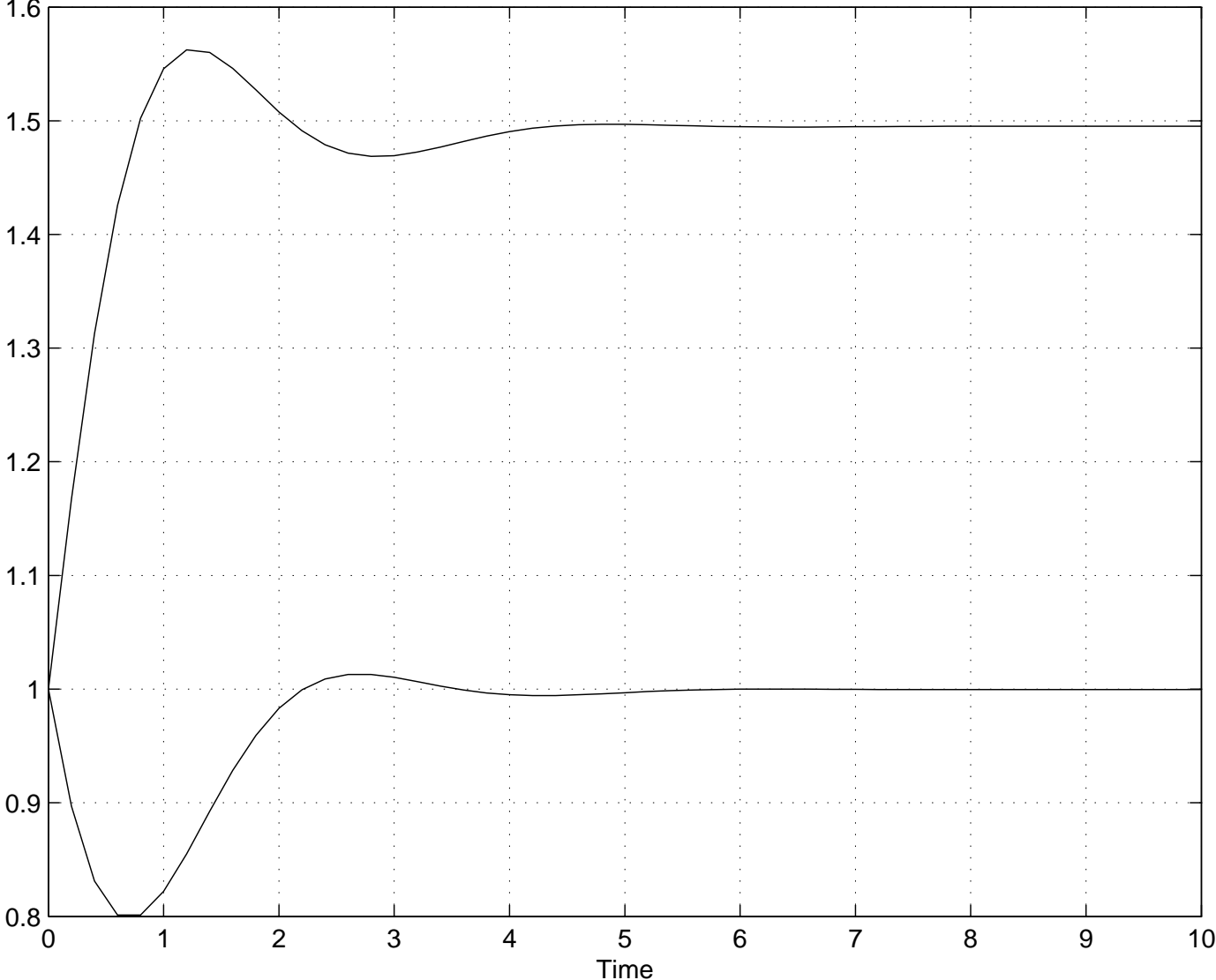
`matrix_riccati.mdl`

- Damped harmonic oscillator; parameter file

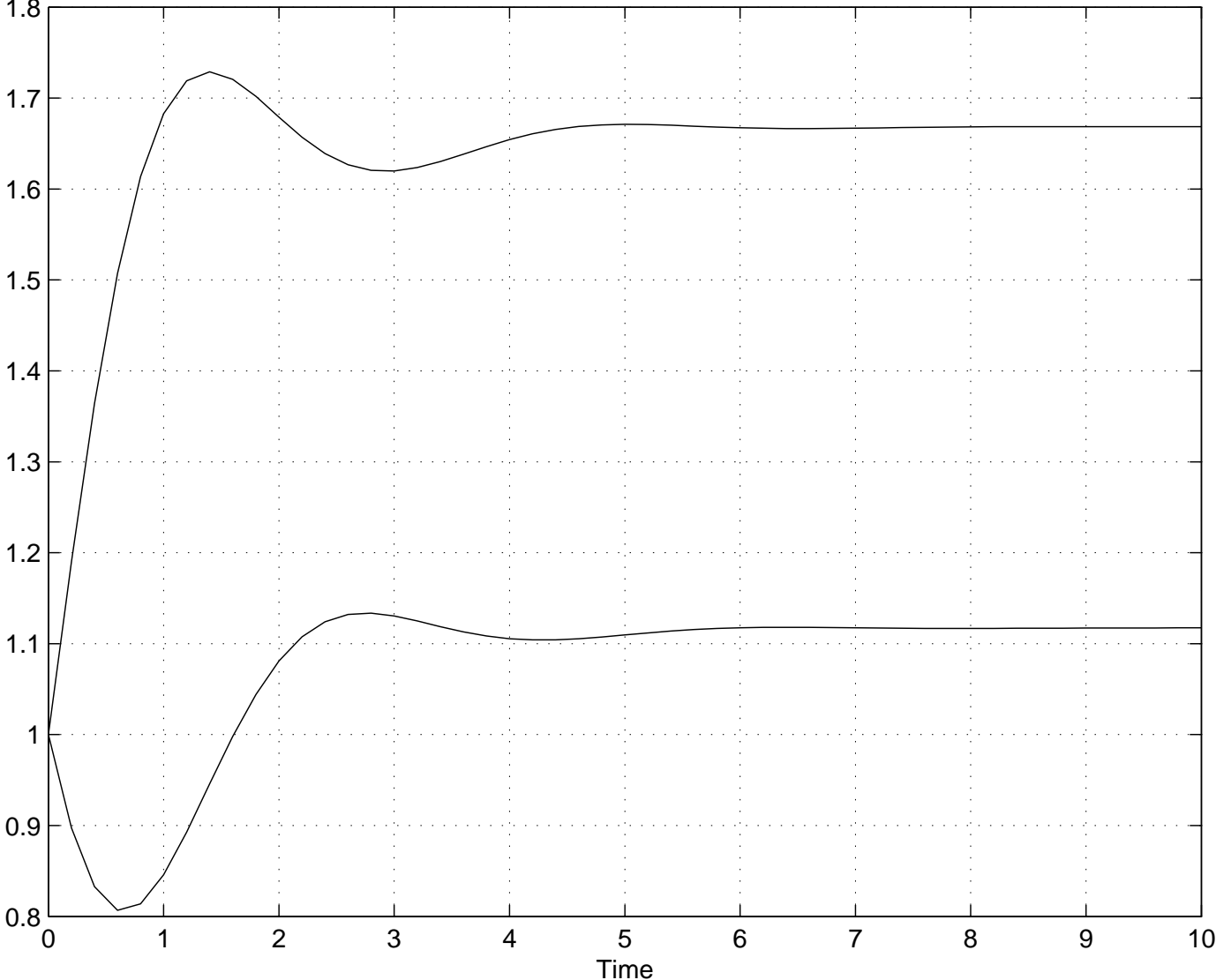
`harmosc_riccati.mat`

- 2d state vector: position and velocity
- 1 scalar measurement and scalar measurement noise R

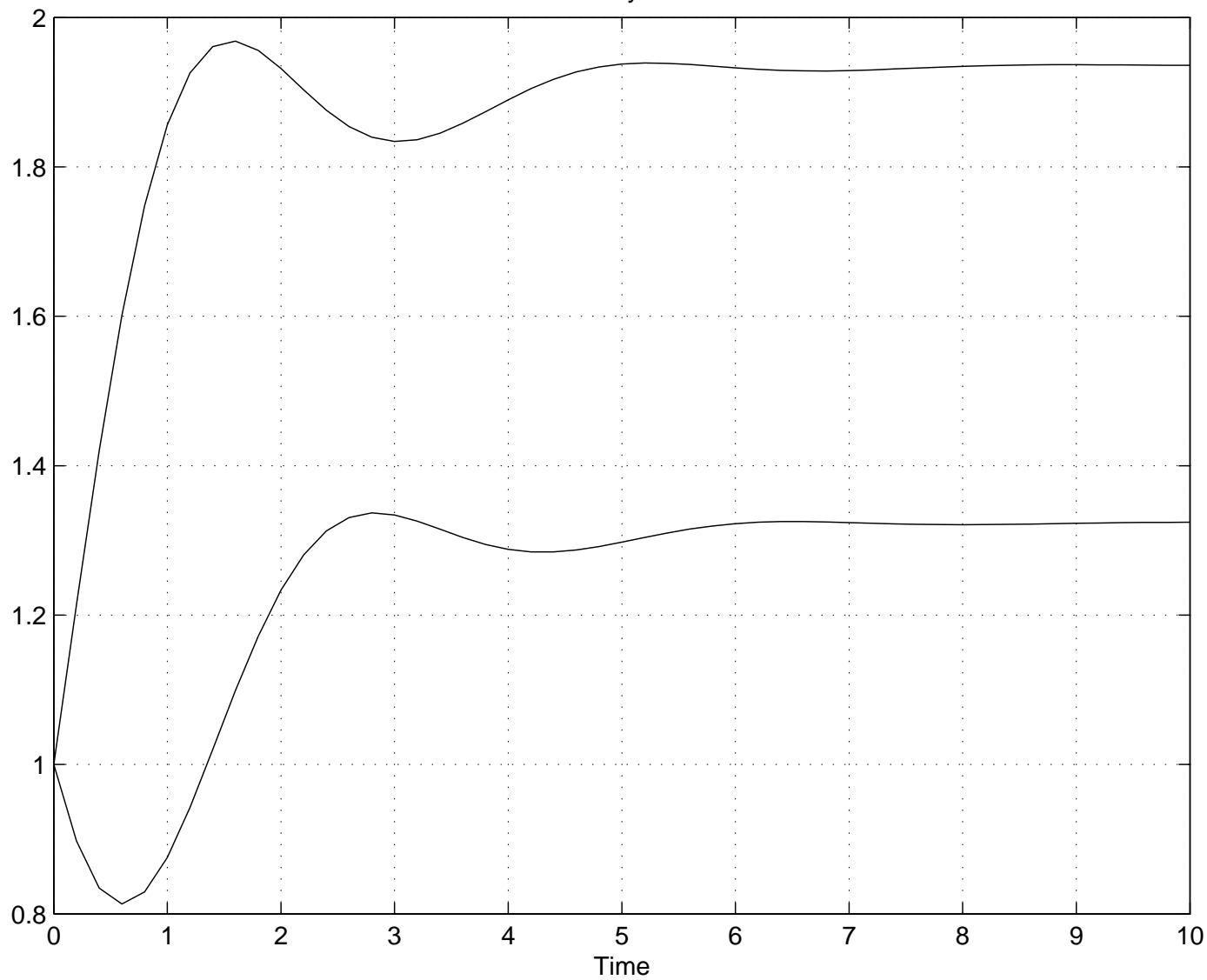
Position and Velocity Variances: R=3.0



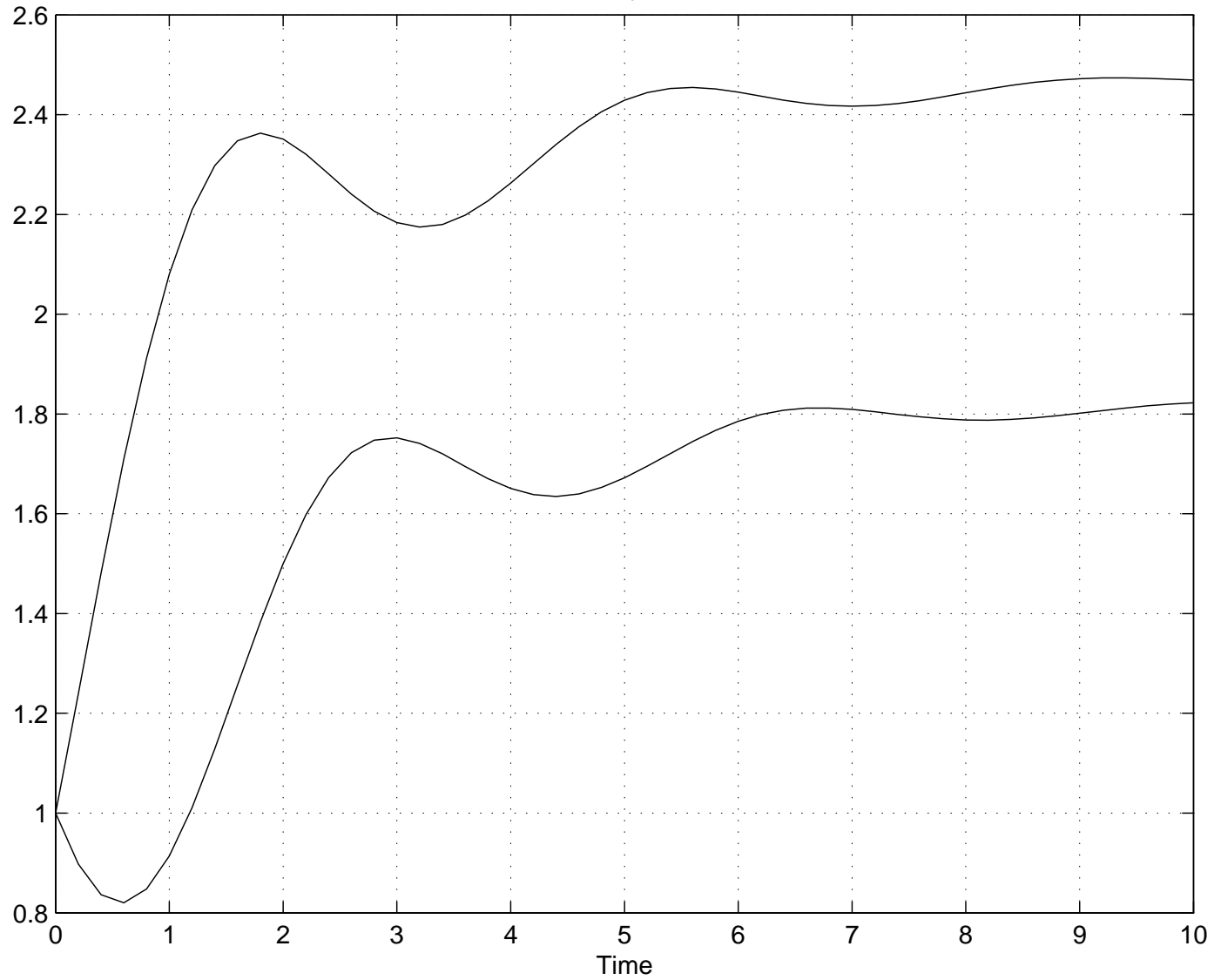
Position and Velocity Variances: R=3.1



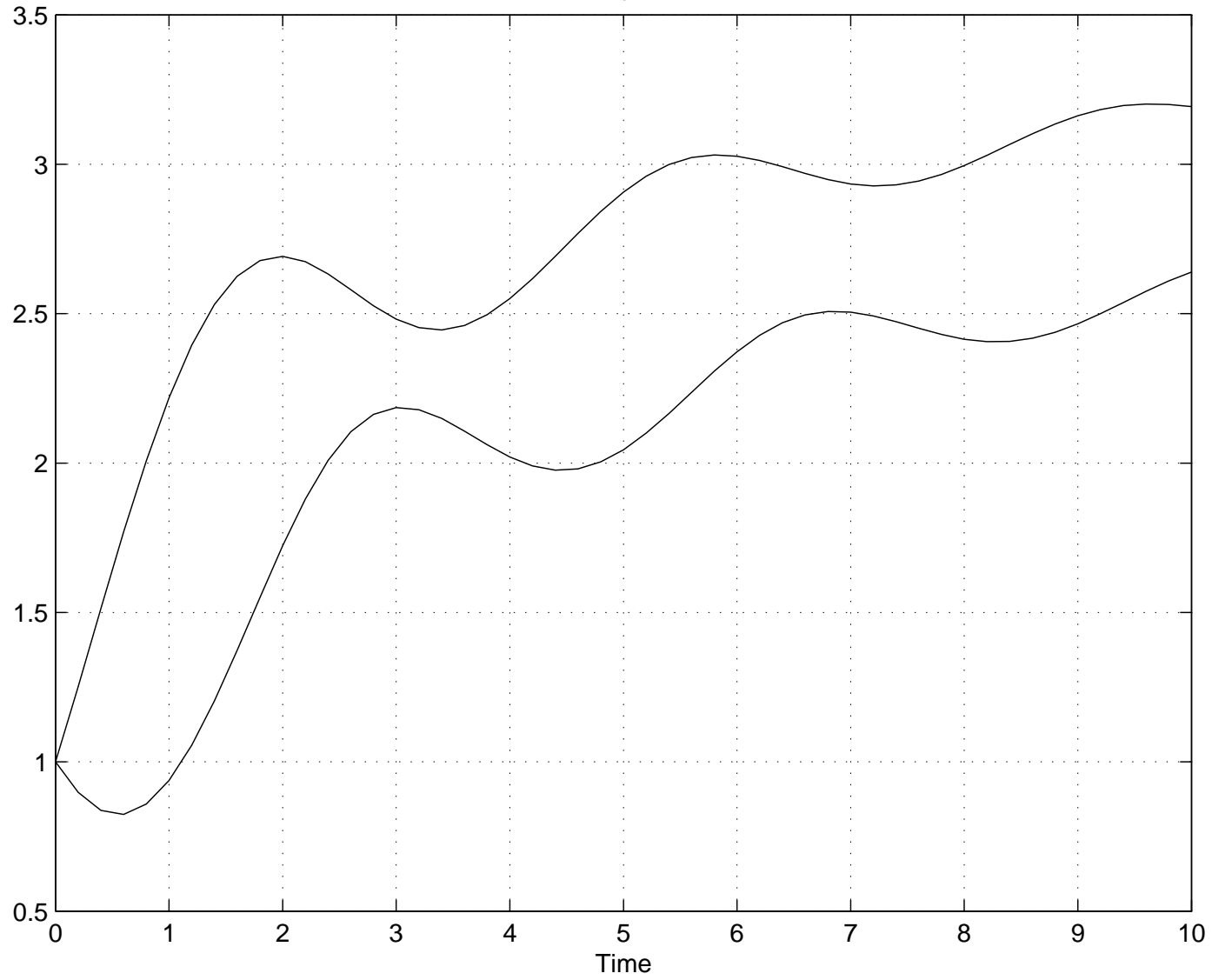
Position and Velocity Variances: R=3.2



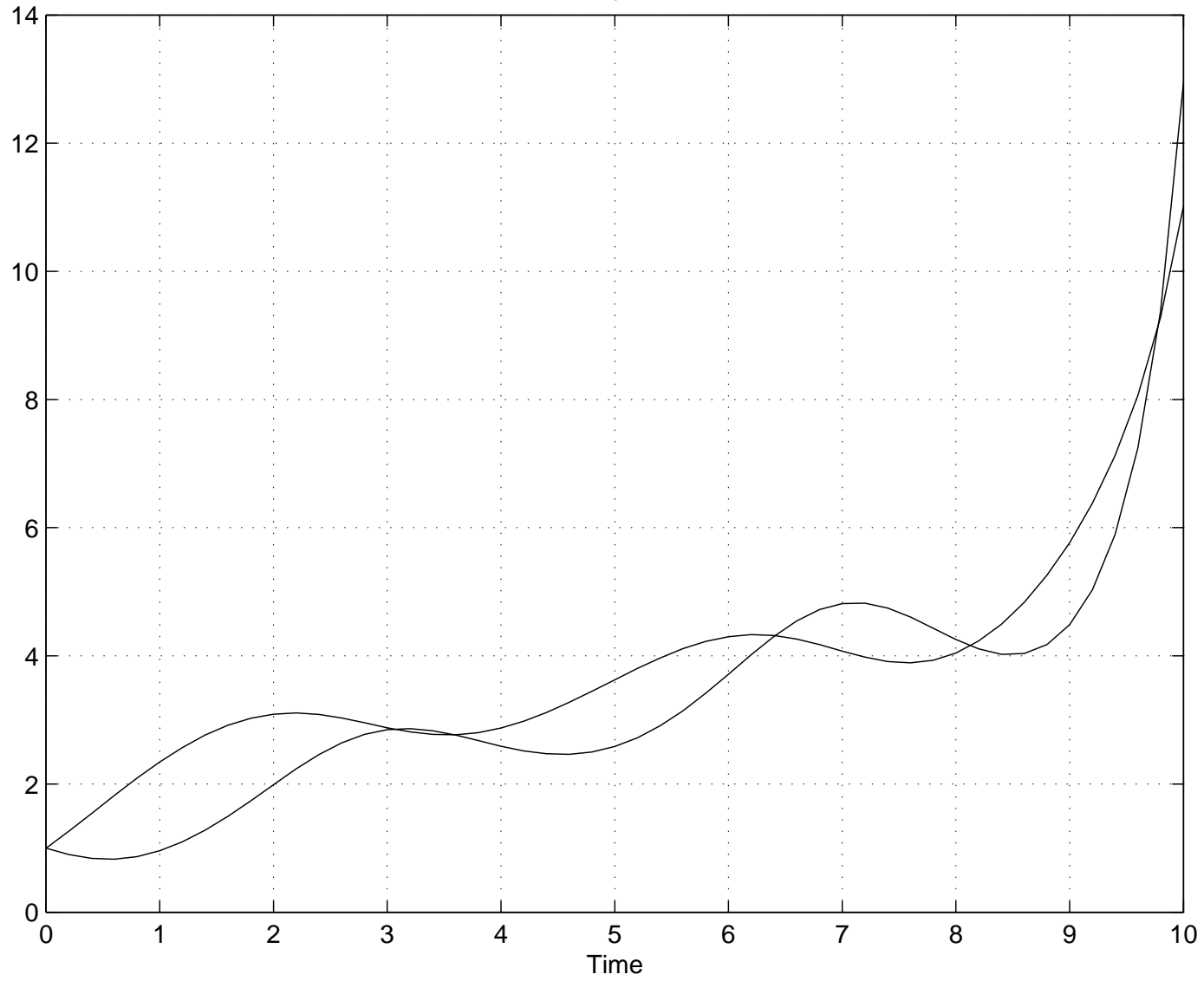
Position and Velocity Variances: R=3.3



Position and Velocity Variances: R=3.35



Position and Velocity Variances: $R=3.39$



Damped Harmonic Oscillator Covariance Analysis

- Matrix Riccati Differential Equation with constant F , H , Q , R
- Varying R demonstrates extreme sensitivity of P solution to changes in R
- $R = 3.3$ is stable; but $R = 3.4$ is unstable

Damped Harmonic Oscillator Covariance Analysis

- Example illustrates how damping coefficient (reflected in system dynamics matrix F) and process noise uncertainty Q interact with measurement noise uncertainty R
- Highly coupled, non-linear equation

$$\dot{P} = FP + PF^T + Q - PH^T R^{-1} HP$$

Alternative Forms of Matrix Riccati Equation

- Can be factored into two first-order, linear matrix
- May be better behaved numerically
 - with Matlab / Simulink it is very easy to implment even non-linear matrix differential equations
 - Simulink's ODE solvers may not always be up to the task, however

Various Approachs to Covariance Simulation

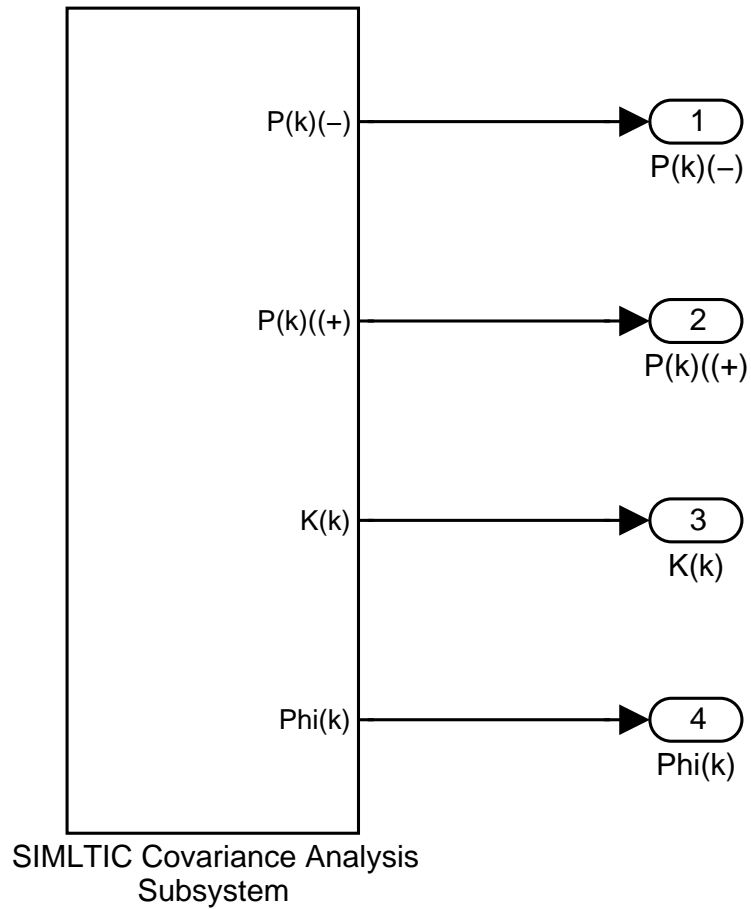
- Discrete models:
 - Implement regular filter processing less the state propagation and update

SIMLTIC: **SIM**ulation of **L**inear **T**ime-**I**nvariant **C**ovariance

- Discrete, Generic Covariance Simulation
- LTI = Constant Copefficients in System Dynamics
- Implemented as Simulink Subsystem Block
 - Can be dropped into higher level models as required
 - Output can be integrated with Matlab Workspace

SIMLTIC

SIMulation of Linear Time Invariant Covariance

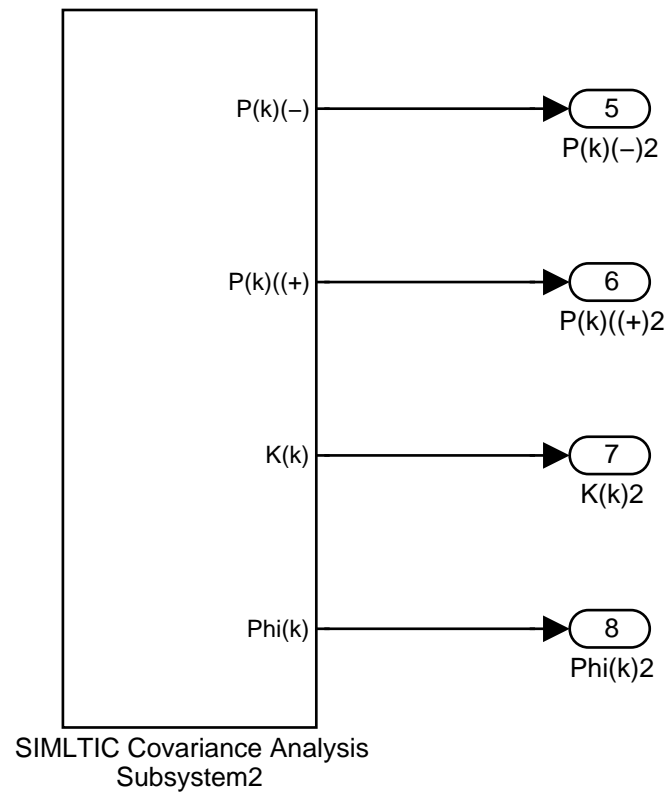
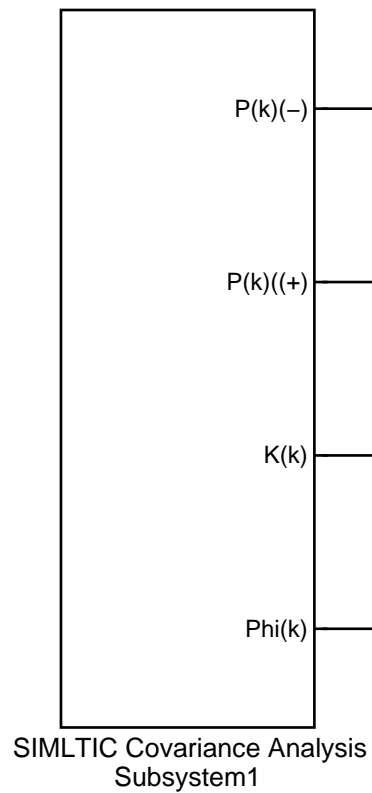


SIMLTIC: **SIM**ulation of **L**inear **T**ime-**I**nvariant **C**ovariance

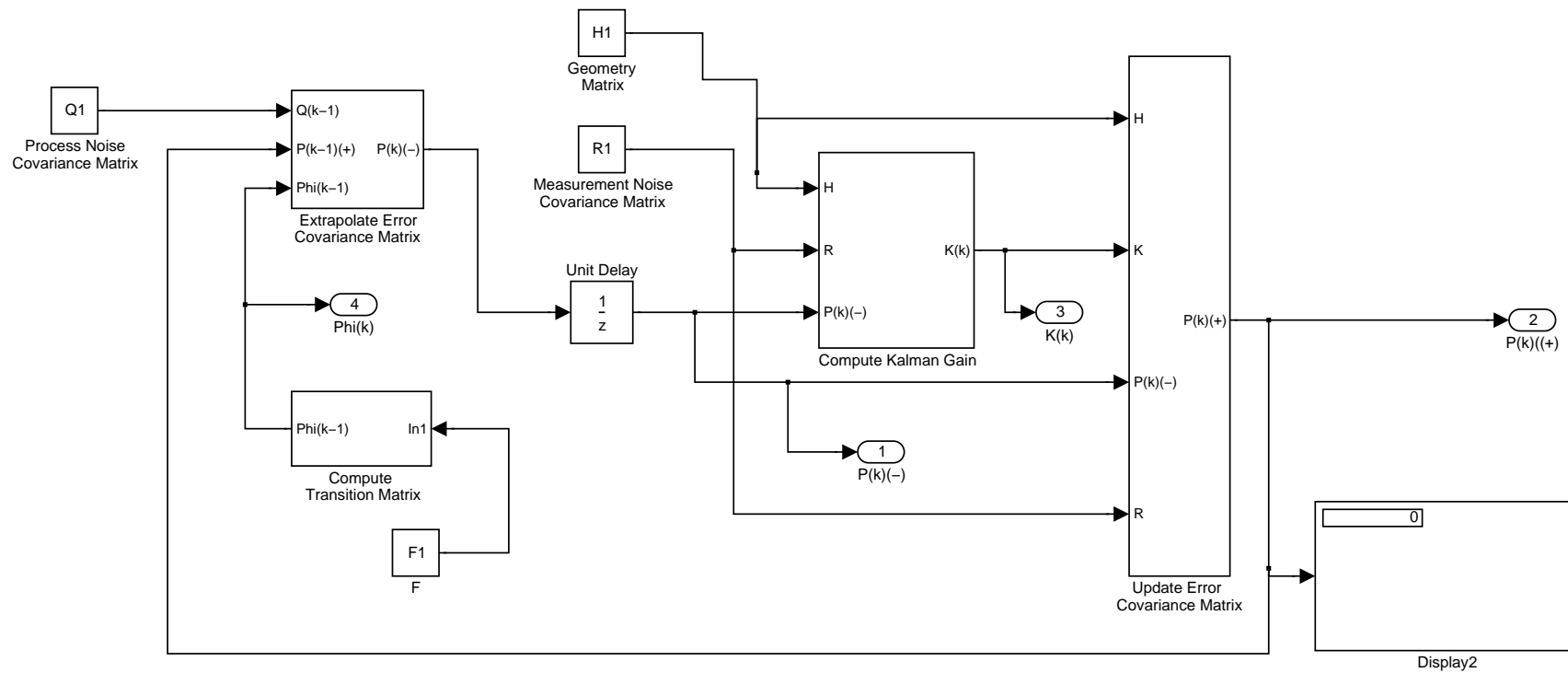
- Example: Covariance Comparison of Two Harmonic Oscillator Models

SIMLTIC

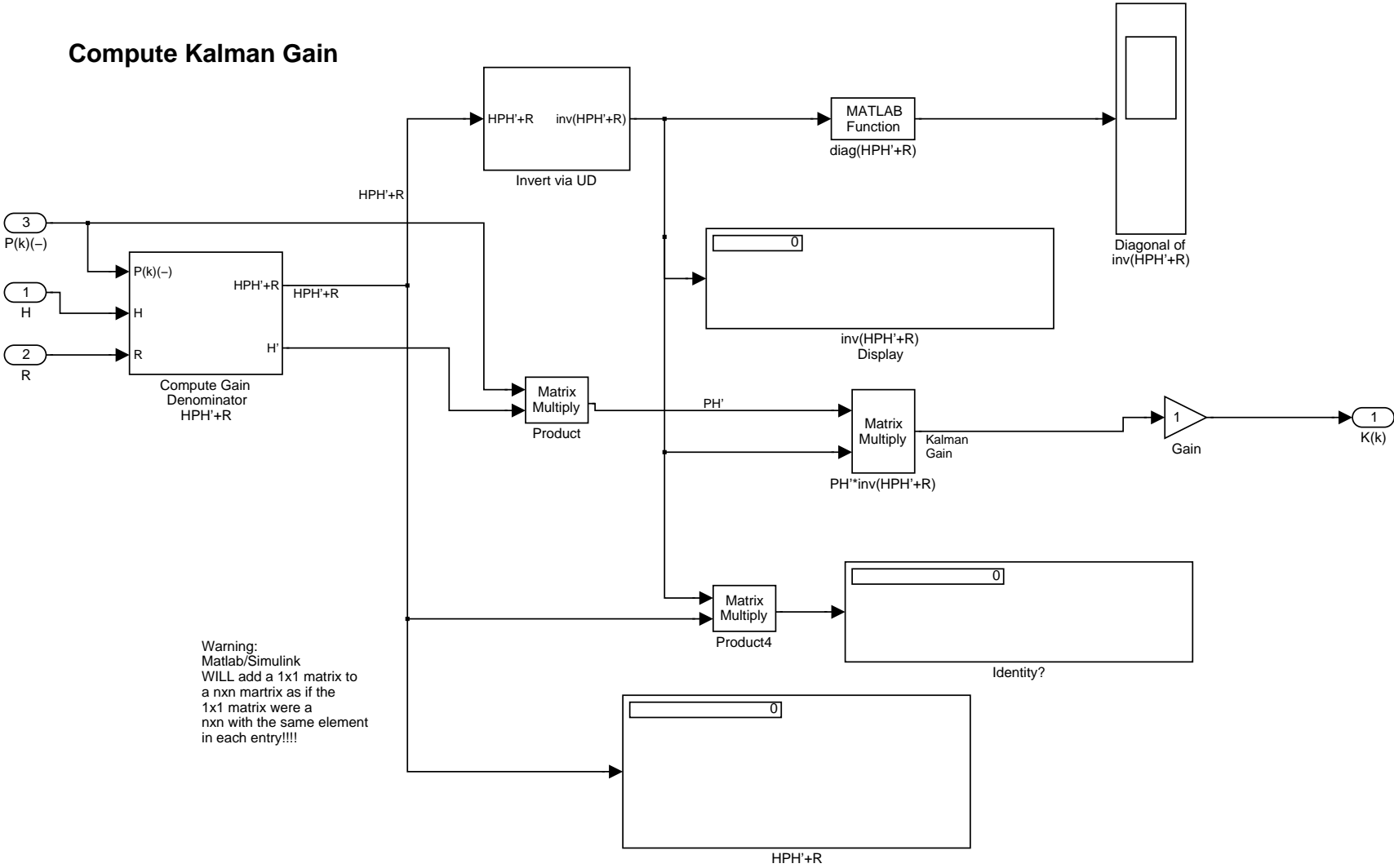
Comparing Performance of Two Models



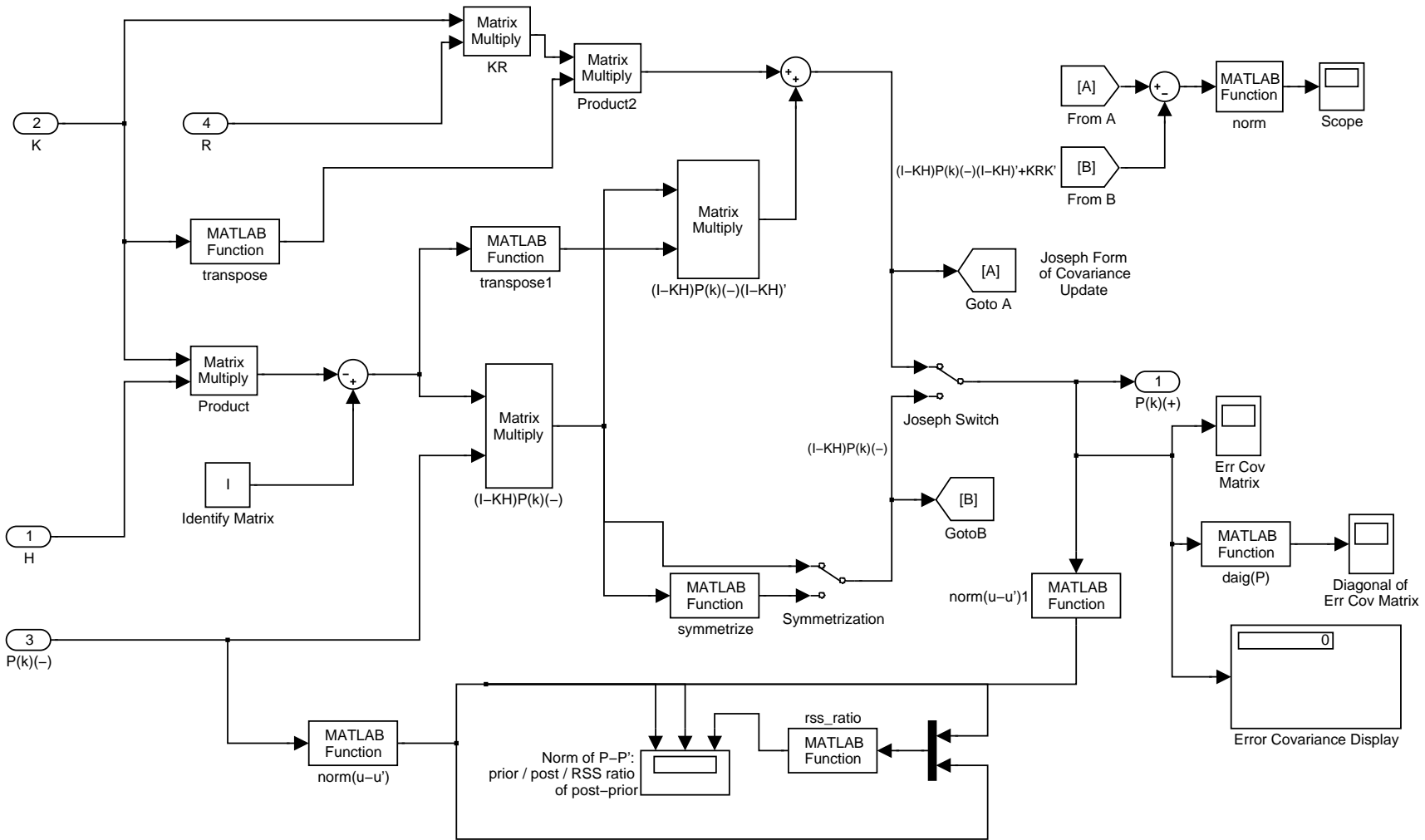
SIMLTIC Subsystem 1



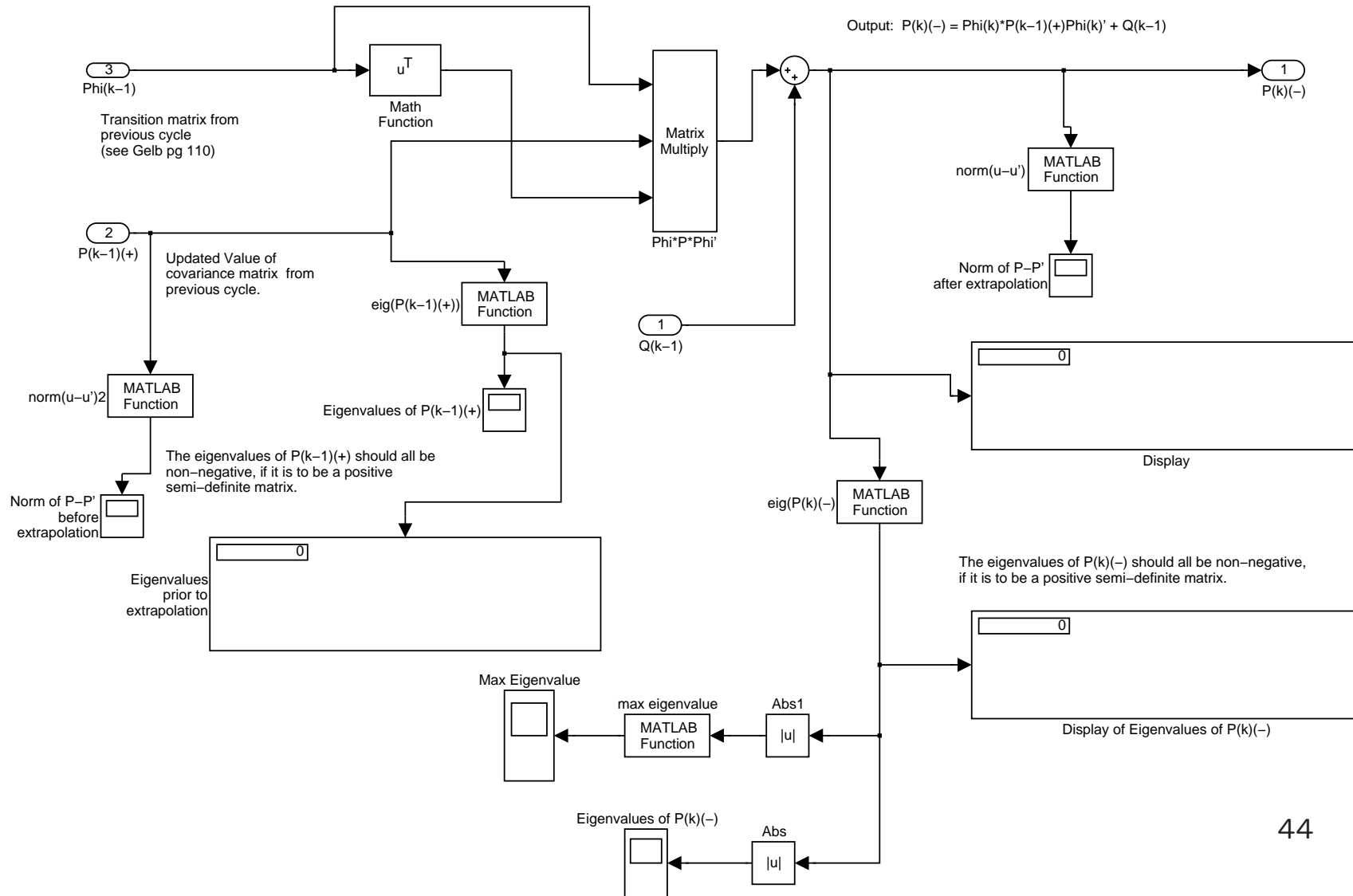
Compute Kalman Gain



Update Error Covariance Matrix

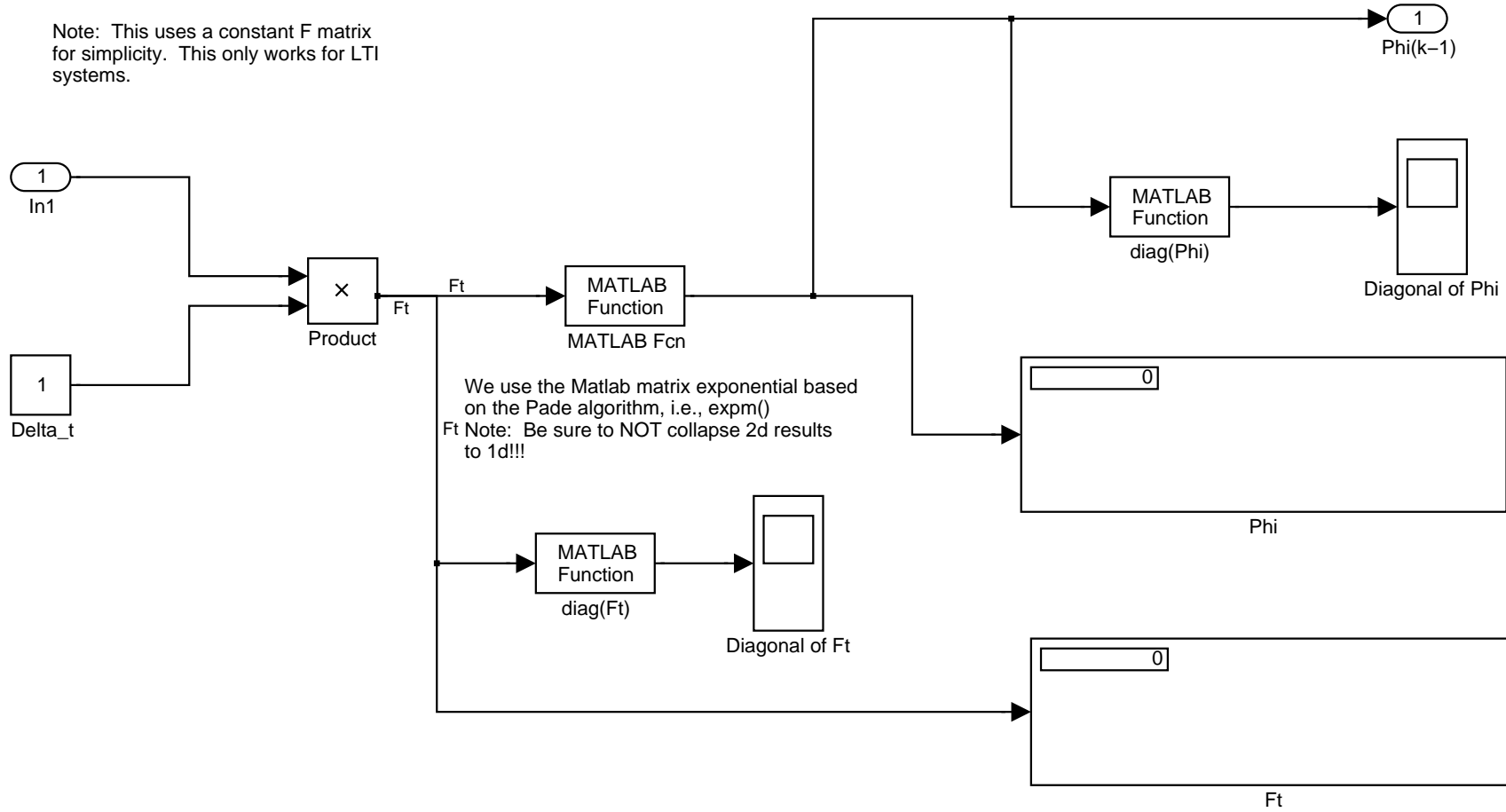


Extrapolate Error Covariance Matrix



Compute Transition Matrix

Note: This uses a constant F matrix for simplicity. This only works for LTI systems.



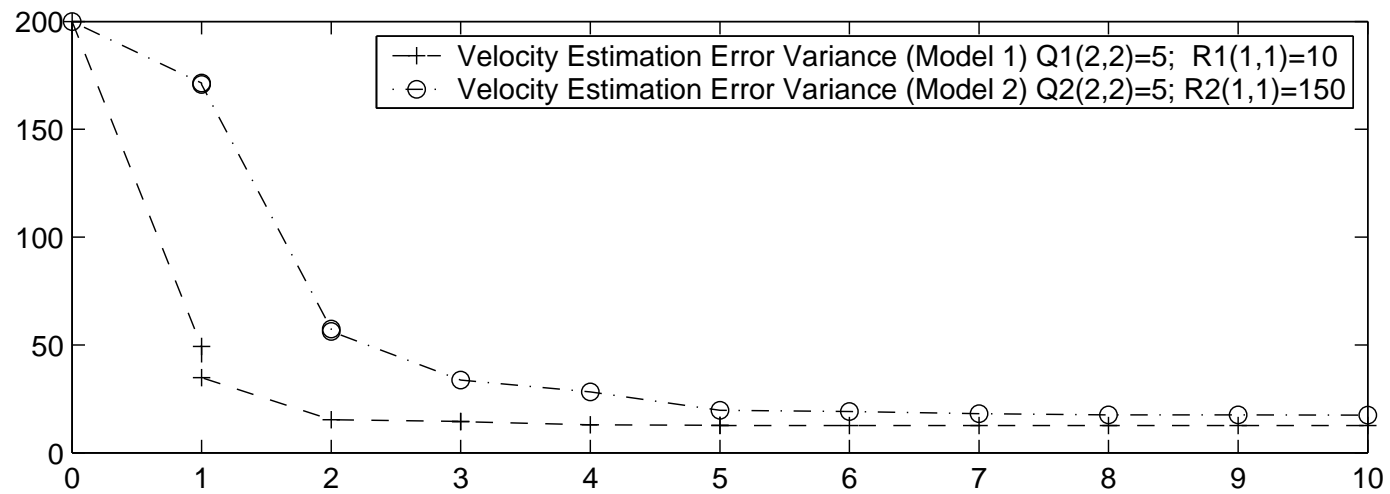
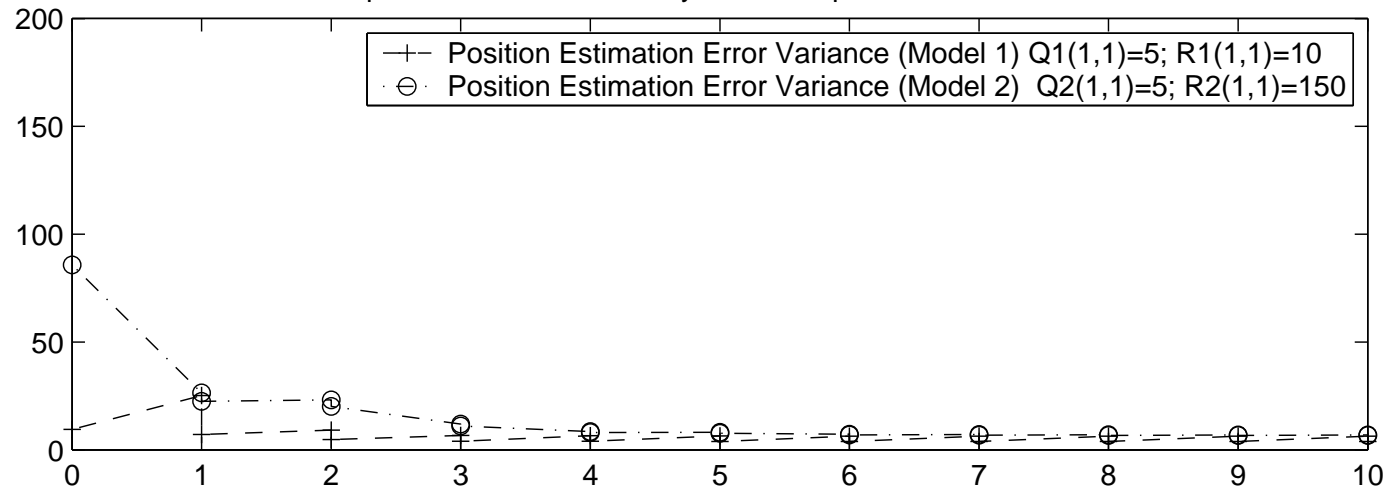
Transition Matrix as Matrix Exponential

- If system dynamics matrix F is constant, i.e., not time-varying over the interval of interest, then $\Phi(\Delta t)$ can be expressed as matrix exponential:

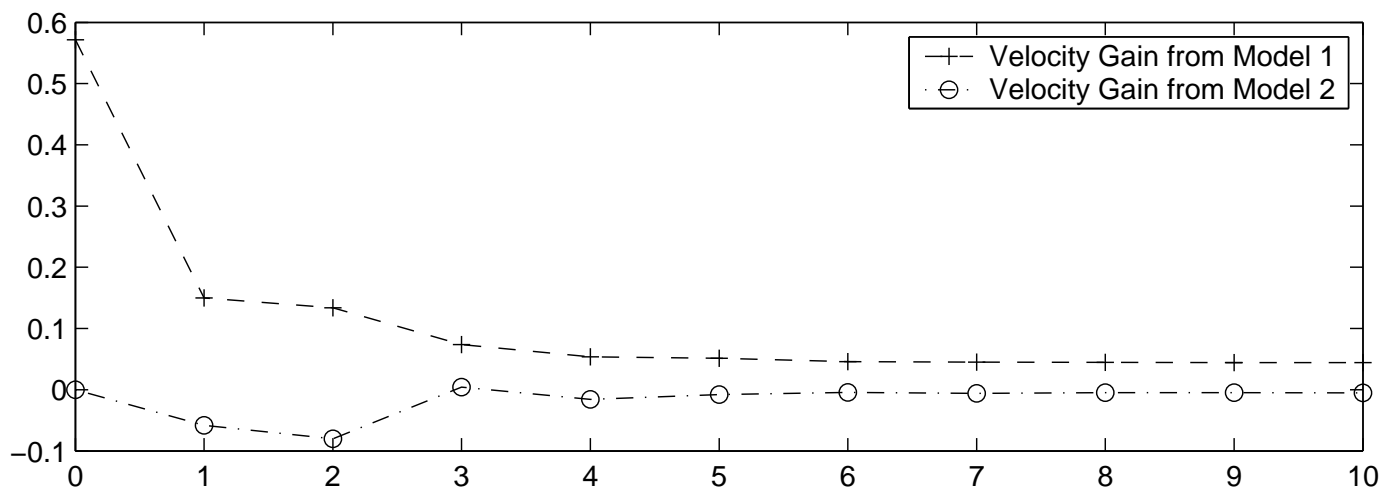
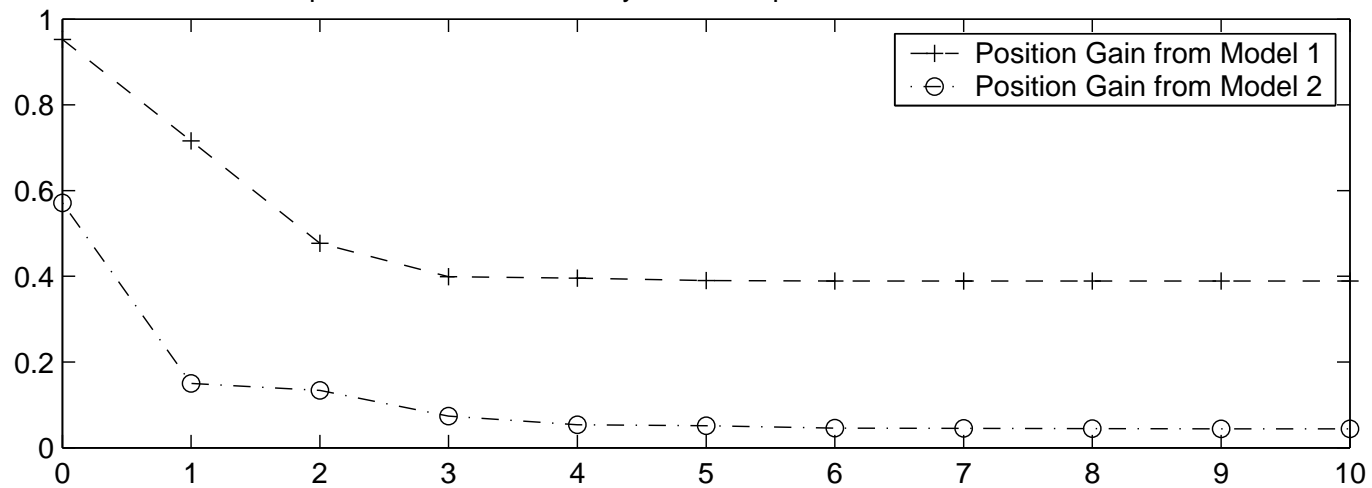
$$\Phi(\Delta t) = I + F\Delta t + \frac{1}{2!} [F\Delta t]^2 + \frac{1}{3!} [F\Delta t]^3 + \dots$$

- If Δt is small enough, the approximation $\Phi(\Delta t) \approx I + F\Delta t$ may be adequate

Comparitive Covariance Analysis of Damped Harmonic Oscillator



Comparitive Covariance Analysis of Damped Harmonic Oscillator: Gains



Trajectories

- In navigation applications, the system dynamics vary according to where you are on the earth
- Some quantities are dependent on your latitude and longitude.
- E.g., the vertical component of earth rate depends on your latitude

Trajectories

- Therefore, in order to perform a covariance analysis of a navigation system, a trajectory generator that computes truth value dynamic quantities such as body rates and accelerations is necessary
- This is the "truth model" that can then be corrupted by sensor error sources

Simulation vs. Real-Time

- Real-time time constraints
- Different processing rates
- Data Buffering
- Measurement Scheduling
- Keeping the Filter Awake
- Covariance Bumping

Time Constraints

- Simulation
 - offline, non-realtime
 - each calculation (transition matrix, Kalman gain, etc.) can run to completion
 - do not have to worry about rate group partitioning
- Real-Time
 - time is of the essence

Real-Time: Different Processing Rates

- In real-time apps, you cannot waste processing cycles
 - there are numerous other mission processing tasks to accomplish besides filtering
- Some, fast varying quantities need to be updated often; others, less often

Real-Time: Different Processing Rates

- For example: body rates and quantities highly-dependent on them need to be updated as rapidly as possible
- Depends, of course, on platform dynamics
- Transport rates don't need to be updated as often

Rate Allocation Example: Combat Talon I

- Kalman Gain Calculation, State and Covariance Update: 0.1 Hz (was originally 0.05 Hz)
- State and Covariance Extrapolation: 4 Hz
 - dependent on some 16 Hz Fast Nav calcs: body rates and attitudes
 - dependent of some 8 Hz Slow Nav calcs: transport rates

Buffering

- Keep low rate data from being clobbered by higher rate data
- Keep high rate data from being clobbered by low rate data
- Buffering: Orderly passage and distribution of data up and down the rate group hierarchy

Buffering Example: Combat Talon I

- Slow rate group exec buffers data to and from faster rate groups
- Faster rate groups do not push or pull data with respect to slower groups
 - REP8HZ buffers data to and from the 16 Hz rate group, but does not control buffering to and from 4 Hz
 - Hence REP16HZ does not call any BU' routines

Buffering Example: Combat Talon I Approach

- Buffer from faster at beginning of start of slow rate group
- Buffer to faster at end of slow rate group
- Within BU'* routine:
 1. Disable interrupts
 2. Perform buffering assignments
 3. Re-enable interrupts

```
117 DEF PROC BU'BUFFER'TO'05HZ'FROM'FASTER;  
118 BEGIN  
119  
120     DISABLE THE INTERRUPTS  
121 %  
122         DISABLE'INTRUPT;  
123  
124     BUFFER SYSTEM LATITUDE AND LONGITUDE FOR G/S  
125 %  
126         SYSTEM'LATITUDE'05 = SYSTEM'LATITUDE'16;  
127         SYSTEM'LONGITUDE'05 = SYSTEM'LONGITUDE'16;  
128  
129     ENABLE THE INTERRUPTS  
130 %  
131         ENABLE'INTRUPT;
```

Measurement Scheduling

- Art / Science unto itself
- Determining which measurements to process when
- Usually fixed schedule with potential operator override
 - CT I Position Fix and Altitude Cal

Measurement Scheduling

- Sequential Processing of Measurements
 - Scalarize to avoid matrix inversion of $HPH^T + R$
 - Or at least reduce dimension
 - feasible if R matrix can be block partitioned into uncorrelated blocks

Measurement Scheduling

- Examining eigenevalues of P
- The larger the eigenvalue, the less observable the associated state eigenvector
- State eigenvector is linear combination of states
- Measurements can be scheduled to improve observability

Keeping Filters Awake

- Filter goes to sleep means error covariance for certain states becomes so small that measurements no longer have an effect
- Can be the result of round-off problems
- Presence of non-random, deterministic quantities can be contributors
 - Random constants are nearly deterministic
 - Watch out for states not driven by process noise!

Keeping Filters Awake

- Recall running average example
- That was a random constant
- Gain $1/N \rightarrow 0$ as $N \rightarrow \infty$
- CT I Filter has 12 states modeled as random constants, i.e., the non-varying states

Covariance Bumping

- Covariance bumping refers to the addition of process noise to keep the filter from going to sleep
- Usually based on ad hoc rules

In CT I GPS Observation Processing, three different kinds of bumping:

```
146  DEF  ITEM  POSITION'BUMP  STATIC REAL = 10.0;
147  DEF  ITEM  V'BUMP          STATIC REAL = 10.0;
148  DEF  ITEM  COVARIANCE'BUMP STATIC REAL = 50.0;
...
182  OBS'COV'SCALER = POSITION'BUMP *
      OBS'COUNT'SCALER;
---
185  NOISE'VECTOR(3) = (Y'KAL'OBS(INU,11) *
      OBS'COUNT'SCALER * OBS'COV'SCALER * V'BUMP) **2;
---
245  OBS'COV'SCALER = COVARIANCE'BUMP *
      OBS'COUNT'SCALER;
```

Exercises

1. For fixed $R = 3.0$, what will the the effect of increasing Q be in the model `harmosc_riccati.mat`?
2. Run the simulation with various values of Q and check to see if the predicted behavior is achieved.

Day 4, Segment 2

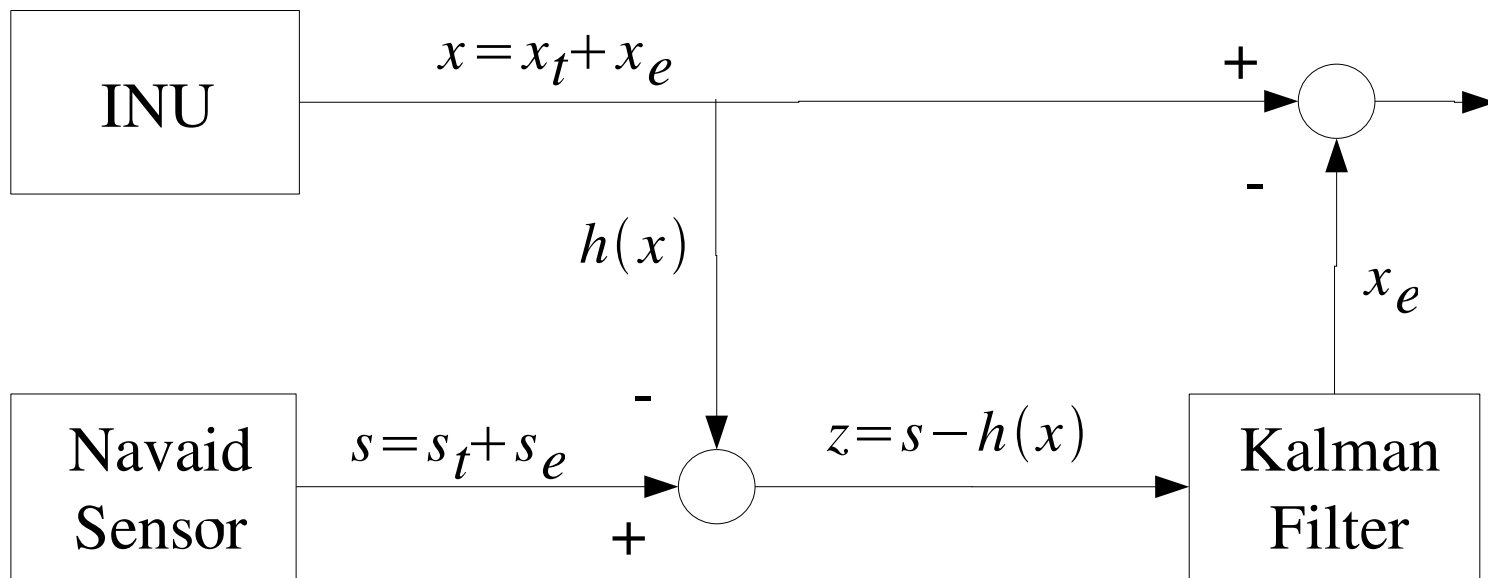
Survey of Nav aids

Topics

- GPS
- Position Fixes
- Radar
- Doppler Velocity Sensor (DVS)
- External Tracking Aids

- Digital Terrain Elevation Data (DTED)

INS Error State Kalman Filter Feedforward Mechanization



Note: $z = s - h(x) = s_t - h(x_t) + s_e - h(x_e)$

Navigation Aids (Nav aids)

- Measurement / Observation sources that aid in estimating INS error sources and dynamics
- Because we are interested in estimating INS errors, we use differencing to cancel out the whole state information
- INS outputs $x = x_t + x_e$, where x_t is the true whole state information generated by INS, i.e., position, velocity, attitude, etc.
- Nav aid sensor outputs $s = x_t + x_e$, where x_t is once again the true quantities corrupted by sensor errors x_e .

Navigation Aids (Nav aids)

- Note that the nav aid sensor usually only outputs part of the whole state vector, e.g., position and velocity, but not attitude (in the case of GPS)
- That's one reason why we have the h function
- The INS signal is mapped by a linearized measurement function (or matrix) h such that $h(x) = h(x_t) + h(x_e)$ and $h(x_t) = s_t$

Navigation Aids (Nav aids)

- Thus, we have

$$\begin{aligned}z &= s - h(x) \\&= s_t + s_e - h(x_t) - h(x_e) \\&= s_t - h(x_t) + s_e - h(x_e) \\&= h(x_t) - h(x_t) + s_e - h(x_e) \\&= s_e - h(x_e)\end{aligned}$$

- Our measurement, therefore, is the difference between the navaid sensor errors and the INS errors

Navigation Aids (Nav aids)

- If s_e is more than just white measurement noise, then we would need to model this vector in the filter according to known or assumed dynamics
- Hence, sometimes adding a new nav aid means adding new states to your filter

Navigation Aids (Nav aids)

- It is crucial to examine the innovations sequence for any non-white time correlations!

$$z_k - h [\hat{x}_k(-)] = s_e(k) - h [(x_e(k)) - h [\hat{x}_k(-)]]$$

where we have added the index k for clarity

- We should have $E \left[(z_k - h [\hat{x}_k(-)])(z_j - h [\hat{x}_j(-)])^T \right] = 0$ when $k \neq j$, which will in general be true provided $E [s_e(k)] = 0$
- If this is not the case, then it is probably s_e that is the culprit and you'd better start modeling!

- The condition

$$E \left[(z_k - h [\hat{x}_k(-)])(z_j - h [\hat{x}_j(-)])^T \right] = 0$$

says that the innovations sequence is uncorrelated in time.

- Recall that the innovations sequence is $\nu_k = z_k - h [\hat{x}_k(-)]$

- Thus, we should have $E [\nu_k \nu_j^T] = 0$

- This means that the optimal estimator $\hat{x}_k(+)$ always makes optimal use of the new information provided by ν_k

GPS: The Global Positioning System

- 24+ satellites orbiting at 10,900 miles above surface
- Three Segments: Control, Space, User
- Time-tagged signals and known satellite position enable calculation of signal time of flight and hence range traversed from satellite to receiver
- Range to four satellites required to determine time plus three position coordinates

GPS: The Global Positioning System

- Satellites have precise atomic clocks
- User set receiver has less precise clock
- Time is of the essence: comparing satellite-generated time-tag with user clock time of arrival determines transit time
- Note that user clock can be a source of systematic error

GPS: The Global Positioning System

- Three types of navaid coupling:
 - Loose coupling
 - Tight coupling
 - Deep coupling

Loosely-Coupled GPS

- GPS receiver-generated position and velocity are differenced with INS position and velocity
- GPS position and velocity are generally pre-filtered (with a Kalman filter) before being used in inertial filter
- GPS position and velocity are also composite quantities from all satellites
- Least accurate approach to using GPS as a navaid

Loosely-Coupled GPS

- The GPS receiver does not have the advantage of using inertial information to completely separate process noise from measurement noise
- Simplifies Kalman integration: Don't necessarily have to model GPS receiver clock errors (assume receiver has already done so)

Tightly-Coupled GPS

- Pseudorange and delta pseudorange for each individual satellite are direct measurements presented to the filter
- Contains only measurement noise, not receiver filter's model of process noise
- Gives INS filter more fine-grained control over GPS measurements
- Higher accuracy

Deeply-Coupled

- Uses high-rate inertial data to aid in code and carrier phase tracking loops
- Most accurate solution when combined with tightly-coupled INS filter as well

GPS Errors

- Ephemeris data: Satellite's reported position can be in error
- Satellite clock: Negligible since SA (selective availability) has been turned off
- Ionosphere: Pseudorange errors caused by free electrons in the ionosphere

GPS Errors (cont)

- Troposphere: Pseudorange errors caused by temperature, pressure, humidity influences on the speed of light
- Multipath: Errors caused by locking on to reflected signals
- Receiver Noise: Errors caused by receiver's thermal noise and software inaccuracies

Error Source	C/A-Code Error Budget (meters)	P-Code Error Budget (meters)
Multipath	12.0	1.2
Receiver Noise / Software	11.1	1.1
Ionospheric Delays	9.0	3.1
Satellite Clock and Ephemeris Errors	3.9	3.9
Tropospheric Delays	2.0	2.0

Using Loosely-Coupled GPS Measurements

- Position
 - Form primary GPS position observation by subtracting Kalman-corrected INS reference position from raw GPS position
 - Next, the innovation (or measurement residual) is formed by subtracting off the extrapolated state estimate of position error
 - Extrapolated position error estimate needs to compensate for accumulated velocity error during Kalman cycle (or some fraction thereof)

Combat Talon I GPS Position Observation, KF'GPS'OBS'PROC,
we have:

```
165 COMPUTE RESIDUALS
166     RESIDUALS(1) = Y'KAL'OBS(INU,1) * OBS'COUNT'SCALER - SVVAR(1)
167 - 5.0 * SVVAR(4);
168     RESIDUALS(2) = Y'KAL'OBS(INU,2) * OBS'COUNT'SCALER - SVVAR(2)
169 - 5.0 * SVVAR(5);
170     RESIDUALS(3) = Y'KAL'OBS(INU,7) * OBS'COUNT'SCALER - SVVAR(3)
171     - 5.0 * SVVAR(6);
```

GPS Velocity Measurements

- Based on carrier phase measurements in receiver
- Generally independent of position measurements
- Observation is once again difference between Kalman-compensated INS reference velocity and GPS-indicated velocity

GPS Measurement Noise Covariance

- Based on EHE (estimated horizontal error) and EVE (estimated vertical error)
- These numbers highly dependent on GDOPs

Geometric Dillution of Precision (GDOP)

- Satellites are not geostationary
- H matrix is time-varying
- Observability varies with time

- $GDOP = \sqrt{\text{tr}(H^T H)^{-1}}$

GDOP Chimney

- Large DGOP is bad; normal values 3-5
- Singularity periods of time (about 1 hr) when GDOP values spike upward with a peak as much as 1000
- GDOP 50 \approx 500 m position error
- Chimney's effects should be less acute if receiver has enough channels to track many satellites and to handle constellation changes

Position Fix

- Overfly Fix
- Known location
- Delta formed between overfly location and indicated, Kalman-compensated INS position
- Usually a one-shot opportunity, i.e., not periodic like GPS

Radar Fixes

- Instantaneous azimuth and elevation angles to a target of known latitude and longitude
- Or converted to local Cartesian coordinates (north, east, and altitude)
- Amount of cursor movement required to align cursor with target yields position fix delta
- Otherwise, processed in filter like GPS or visual position fix

Doppler Velocity Sensor (DVS)

- DVS provides genuine velocity measurement with respect to ground
- DVS provides heading and drift velocities V_H and V_D
- Nonlinear measurement relationship to INS-indicated velocities V_x and V_y and platform azimuth β :

$$V_H = -V_x \sin \beta + V_y \cos \beta$$

$$V_D = V_x \cos \beta + V_y \sin \beta$$

DVS

- Relationship between (V_x, V_y) and (V_H, V_D) is non-linear h
- To generate a linearized H matrix we compute partial derivatives (Jacobians) to yield:

$$H = \begin{bmatrix} 0 & -\sin \beta & 0 & 0 & \cos \beta & 0 & 0 & 0 & -V_x \cos \beta - V_y \sin \beta \\ 0 & \cos \beta & 0 & 0 & \sin \beta & 0 & 0 & 0 & -V_x \sin \beta + V_y \cos \beta \end{bmatrix}$$

for a canonical 9-state filter.

External Tracking Aids

- External platform tracks your vehicle and sends position / velocity info (JTIDS)
- Bearing and distance measurements
 - VOR, DME, TACAN

Digital Terrain Elevation Data (DTED)

- Published by National Geospatial Intelligence Agency (NGA)
 - formerly NIMA, formerly DMA
- 1 deg geocells of terrain height for each lat/long pair
- DTED Levels 0,1,2

DTED

- As navaid, a trail of radar altimeter measurements are stored
- Trail is correlated with DTED to determine offset
- SITAN, TERPROM

Day 4, Segment 3

Filter Maintenance

Topics

- Filter Performance and Integrity Monitoring
- Replacing Nav aids
- Adding New Nav aids
- Augmenting the State Vector

Filter Performance and Integrity Monitoring

- Covariance Monitoring
- Innovations Monitoring
- Corrective Action

Covariance Monitoring

- Ensuring Symmetry
- Managing Observability
- Ensuring Positive Definiteness

Covariance Monitoring: Ensuring Symmetry

- P is supposed to be positive definite and symmetric
- Symmetry means that $P^T = P$, i.e., for each element p_{ij} of P we must have $p_{ij} = p_{ji}$
- Roundoff error can cause loss of symmetry
- The best way to ensure symmetry is to work only with items on or (say) above the diagonal and simply define elements below the diagonal by fiat.

Covariance Monitoring: Ensuring Symmetry

- Both $P(+)$ and $P(-)$ need to be symmetrized
 - After extrapolation: symmetrize
 - After update: symmetrize

Covariance Monitoring: Managing Observability

- Observability means there is a linear path from the measurements to states
- Structural observability determined by Φ and H :

If H is $1 \times n$, where n is the size of the state vector, then the state x_0 is said to be **observable** from n scalar measurements $\{z_0, \dots, z_{n-1}\}$ if the block matrix

$$\begin{bmatrix} H^T & \Phi^T H^T & \dots & (\Phi^T)^{n-1} H^T \end{bmatrix}$$

is of rank n

Covariance Monitoring: Managing Observability

- If unobservable states are also unstable, estimation errors will be unstable and P will show symptoms
 - $\text{diag}(P) \rightarrow \infty$
 - I.e., one or more diagonal elements of P will grow without bound
- Not the same as divergence: "feature" of the design

Covariance Monitoring: Managing Observability

- Note also that the state estimation error variances (i.e., the diagonals of P) will naturally grow in the absence of measurements
- Going for a long time without incorporating a measurement is equivalent to lack of observability in the associated states
- Use covariance analysis simulation to detect non-observability
- The only real solution is to add new measurements

Covariance Monitoring: Ensuring Positive Definiteness

- P is supposed to be positive definite
- This means that $x'Px > 0$ whenever $x \neq 0$
- States modeled as random constants or other deterministic states can cause loss of positivity
- Add covariance bumping: Add small amounts of process noise to diagonal elements of Q for these states

Replacing Nav aids

- Replacing a given sensor with another of the same type usually requires no changes to the state model
 - unless replacement sensor has significant error dynamics that need to be modeled (see adding new sensors and state augmentation)
- Measurement noise covariance generally has to change however

Adding New Nav aids

- What kinds of nav aids are desirable?
- What are the steps involved in adding a new nav aid?

Adding New Nav aids

Desirable Characteristics:

- Nav aids which render observable otherwise unobservable states
- Those having a simple relationship (preferably linear) to the state vector
- Having only white noise error dynamics

Augmenting the State Vector

- State Vector Augmentation: Expanding the size of the state vector
 - i.e., adding new states
- Why?
 - new nav aids with their own error dynamics (e.g., GPS)
 - correlated (non-white) process noise
 - correlated (non-white) measurement noise

Augmenting the State Vector

- Why? (continued)
 - Correlation \Rightarrow dynamics
 - * No dynamics in white noise
 - If dynamics are present they must be modeled
 - Otherwise, dynamic effects will be allocated to other states

Augmenting the State Vector

- How?
 - model non-white noise as result of feeding white noise through a linear system
 - new states are outputs of new integrators
- State transition matrix Φ and error covariance matrix P grow as j^2 , where j is number of new states
- Geometry matrix H grows as $m \cdot j$, where m is number of measurements and j is number of new states

Augmenting the State Vector

Example: Ship Inertial Navigation Systems (SINS)

- Long mission times
 - Especially submarines
- Position fixes insufficient to calibrate gyro bias drift w/o further modeling

Augmenting the State Vector

Example: Ship Inertial Navigation Systems (SINS)

- Gyro bias drift due to slowly changing bias
- Can be modeled as random walk
- Random walk modeled as integrated white noise
- Kalman filter needed to estimate drift

Augmenting the State Vector

Example: Ship Inertial Navigation Systems (SINS)

- Assume platform tilts are kept close to zero through velocity measurements damping the Schuler oscillations
- North-West-Up (NWU) coordinate frame (or West-South-Up)

Augmenting the State Vector

Example: Ship Inertial Navigation Systems (SINS)

- Two angular position errors in radians: ψ_N and ψ_W
 - Note ψ_N is a small angular rotation about the north axis and thus induces a west position error.
 - Similarly, ψ_W is a small rotation about the west axis and thus induces a south position error.
- $\theta_U = \psi_Z - \psi_N \tan(\lambda)$, where ψ_Z is platform azimuth error and λ is the latitude.

Augmenting the State Vector

Example: Ship Inertial Navigation Systems (SINS)

- Error Dynamics:

$$\begin{aligned}\dot{\psi}_N - \Omega_U \psi_W &= \delta g_N \\ \dot{\psi}_W + \Omega_U \psi_N - \Omega_N \theta_U &= \delta g_W \\ \dot{\theta}_U + \Omega_N \psi_W &= \delta g_U\end{aligned}$$

where $\delta g_N, \delta g_W, \delta g_U$ are non-white gyro bias drift errors and $\Omega_N = \Omega \cos(\lambda)$ and $\Omega_U = \Omega \sin(\lambda)$ are components of earth rate

Augmenting the State Vector

Example: Ship Inertial Navigation Systems (SINS)

- Since $\delta g_N, \delta g_W, \delta g_U$ are non-white, the question arises as to what kind of model is appropriate
- Empirical data indicates that gyro biases exhibit random walk behavior over time
- Since random walk is the integral of white noise, white noise can be viewed (in some sense) as the derivative of random walk

Augmenting the State Vector

Example: Ship Inertial Navigation Systems (SINS)

- Hence, the following dynamics of random walk apply:

$$\dot{\delta g}_N = w_N$$

$$\dot{\delta g}_W = w_W$$

$$\dot{\delta g}_U = w_U$$

where w_N, w_W, w_U are white noise processes.

Augmenting the State Vector

Example: Ship Inertial Navigation Systems (SINS)

- We use the random walk equations to augment the state dynamics equations
- Thus we now have a six-state model:

$$\begin{bmatrix} \dot{\psi}_N \\ \dot{\psi}_W \\ \dot{\theta}_U \\ \dot{\delta g}_N \\ \dot{\delta g}_W \\ \dot{\delta g}_U \end{bmatrix} = \begin{bmatrix} 0 & \Omega_U & 0 & 1 & 0 & 0 \\ -\Omega_U & 0 & \Omega_N & 0 & 1 & 0 \\ 0 & -\Omega_N & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \psi_N \\ \psi_W \\ \theta_U \\ \delta g_N \\ \delta g_W \\ \delta g_U \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ w_N \\ w_W \\ w_U \end{bmatrix}$$

Augmenting the State Vector: Summary

- Straightforward: When new, relevant dynamic relations found, simply add the states to the state vector and add the new dynamics to the F and/or Φ matrix
- Use basic models such as low-order systems driven by white noise to model new states

Augmenting the State Vector: Summary

- Some Popular Models
 - random walk
 - random constant
 - random ramp
 - Gauss-Markov (i.e., exponentially correlated)
 - combinations of the above!

Exercises

1. Write a simple 1-state model for a gyro bias modeled as a random constant
2. Now augment this model to a 2-state model by adding a random walk

Day 4, Segment 4

Filter Redundancy and Fault Tolerance

Topics

- Combat Talon Weighting Scheme
- Detecting Faults
- Graceful Degradation

Combat Talon Weighting Scheme

- Two separate INS error dynamics filters
- Use error covariance outputs from each filter to weight state estimates

Combat Talon Weighting Scheme

- Weighting scheme:

$$w_i(x) = \frac{\sigma_j(x)}{\sigma_1(x) + \sigma_2(x)}$$

where $i \neq j \in \{1, 2\}$ and x is the parameter in question

Alternatives to Weighting: Decentralized Filtering

- Decentralized filtering without feedback
- Decentralized filtering with feedback

Decentralized Filtering without Feedback

- Assume two local filters (like CT I)
- Then for the global error covariance we have:

$$P^{-1} = P_1^{-1} + M_1^{-1} + P_2^{-1} + M_2^{-1} + M^{-1}$$

where P_1, P_2 are updated local error covariances for each filter, M_1, M_2 are extrapolated local error covariances prior to update, and M is the global filter's prior error covariance.

Decentralized Filtering without Feedback

- The global filter's estimate is

$$\hat{x} = P \left[P_1^{-1} \hat{x}_1 - M_1 m_1^{-1} + P_2^{-1} \hat{x}_2 - M_2 m_2^{-1} + M^{-1} m \right]$$

where m_1, m_2, m are the local a priori estimates for local filters 1 and 2 and the global filter, respectively.

Exercises

1. Let the x position error variance for filter 1 be 4 meters, that of filter 2 16 meters. Calculate the weighted average latitude given $\text{lat}_1=33.3$ deg, $\text{lat}_2=33.21$.
2. How would the decentralized filter's estimate differ from the weighted average?