

Applied Linear Optimal Control

Examples and Algorithms

ARTHUR E. BRYSON

Stanford University



PUBLISHED BY THE PRESS SYNDICATE OF THE UNIVERSITY OF CAMBRIDGE
The Pitt Building, Trumpington Street, Cambridge, United Kingdom

CAMBRIDGE UNIVERSITY PRESS
The Edinburgh Building, Cambridge CB2 2RU, UK
40 West 20th Street, New York, NY 10011-4211, USA
477 Williamstown Road, Port Melbourne, VIC 3207, Australia
Ruiz de Alarcón 13, 28014 Madrid, Spain
Dock House, The Waterfront, Cape Town 8001, South Africa
<http://www.cambridge.org>

© Arthur E. Bryson 2002

This book is in copyright. Subject to statutory exception
and to the provisions of relevant collective licensing agreements,
no reproduction of any part may take place without
the written permission of Cambridge University Press.

First published 2002

Printed in the United States of America

Typefaces Times Ten 10/13 pt. and Helvetica Neue *System* L^AT_EX 2_ε [TB]

A catalog record for this book is available from the British Library.

Library of Congress Cataloging in Publication Data

Bryson, Arthur E. (Arthur Earl)

Applied linear optimal control : examples and algorithms / Arthur E. Bryson.

p. cm.

Includes bibliographical references and index.

ISBN 0-521-81285-2 – ISBN 0-521-01231-7 (pb.)

1. Linear control systems. I. Title.

TJ220 .B78 2002

629.8'32 – dc21

2001052553

ISBN 0 521 81285 2 hardback

ISBN 0 521 01231 7 paperback

Contents

Preface	<i>page</i> ix
OPTTEST Toolbox	xiii
Acronyms and Abbreviations	xvii
Nomenclature	xix
1 Static Estimation	1
1.1 Random Scalars	1
1.2 Random Vectors	2
1.3 Generating Gaussian Vectors	5
1.4 Static Linear Estimation	8
1.5 Static Nonlinear Estimation	13
1.6 Chapter Summary	18
2 Random Processes	20
2.1 Discrete Random Processes	20
2.2 Discrete Gauss–Markov Processes	22
2.3 Prediction and Simulation of Discrete Gauss–Markov Processes	25
2.4 Continuous Gauss–Markov Processes	29
2.5 Prediction and Simulation of Continuous Gauss–Markov Processes	36
2.6 Chapter Summary	40
3 Dynamic Estimation – Filters	43
3.1 Introduction	43
3.2 Discrete Filters	44
3.3 Continuous Filters	57
3.4 Discrete Backward Filters	73
3.5 Continuous Backward Filters	77
3.6 Chapter Summary	80

4	Dynamic Estimation – Smoothers	83
4.1	Introduction	83
4.2	Discrete Smoother Problem and Batch Algorithm	83
4.3	Smoother Two-Point Boundary-Value Problem and Recursive Algorithms	85
4.4	Continuous Smoothers	92
4.5	Chapter Summary	97
5	Linear–Quadratic State-Feedback Follower–Controllers	98
5.1	Introduction	98
5.2	Discrete and Zero-Order-Hold LQ SFB Follower–Controllers	98
5.3	Continuous LQ SFB Follower–Controllers	118
5.4	Chapter Summary	122
6	Linear–Quadratic Gaussian Follower–Controllers	123
6.1	Introduction	123
6.2	Discrete LQG Controllers	123
6.3	Continuous LQG Controllers	143
6.4	Chapter Summary	151
7	Smoothers for Controlled Plants	152
7.1	Introduction	152
7.2	Batch Smoother for Controlled Plants	152
7.3	Recursive Discrete Smoother for Controlled Plants	155
7.4	Continuous Smoothers for Controlled Plants	155
7.5	Parameter Identification as Extended Nonlinear Smoothing	158
7.6	Chapter Summary	166
8	Time-Invariant Filters	167
8.1	Introduction	167
8.2	Discrete Time-Invariant Filters	167
8.3	Continuous Time-Invariant Filters	175
8.4	Chapter Summary	181
9	Time-Invariant Linear–Quadratic State-Feedback Follower–Controllers	183
9.1	Introduction	183
9.2	Discrete TI SFB Controllers	184
9.3	Discrete Model Following and Disturbance Attenuation	191
9.4	Continuous TI SFB Controllers	201
9.5	Model Following and Disturbance Attenuation	213
9.6	Chapter Summary	222
10	Time-Invariant Linear–Quadratic Gaussian Controllers	224
10.1	Introduction	224
10.2	Discrete TI LQG Controllers	224
10.3	Discrete TI LQG Model Following and Disturbance Rejection	236
10.4	Continuous TI LQG Controllers	240

10.5	Continuous TI Model Following and Disturbance Rejection with Random Inputs	249
10.6	Chapter Summary	255
11	Worst-Case Controllers	258
11.1	Introduction	258
11.2	Discrete LQW Controllers with SFB	259
11.3	Discrete LQW Estimators	264
11.4	Discrete LQW Controllers with ESFB	268
11.5	Continuous LQW Controllers with SFB	271
11.6	Continuous LQW Estimators	276
11.7	Continuous LQW Controllers with ESFB	279
11.8	Discrete TI LQW Controllers	282
11.9	Continuous TI LQW Controllers	284
11.10	Best-Case Controllers	286
11.11	Chapter Summary	287
12	Parameter-Robust LQG Controllers	289
12.1	Introduction	289
12.2	A Parameter-Robustness Measure	289
12.3	Optimal Robust Controllers	291
12.4	Other Robust Design Methods	306
12.5	Chapter Summary	307
Appendix A	Filters and Controllers with Colored Measurement Noise	308
A.1	Introduction	308
A.2	TV Discrete Filtering	308
A.3	TV Continuous Filtering	312
A.4	TV Discrete LQG Controllers	316
A.5	TV Continuous LQG Controllers	319
A.6	TI Discrete Filtering	321
A.7	TI Continuous Filtering	323
A.8	TI Discrete LQG Controllers	324
A.9	TI Continuous LQG Controllers	325
A.10	Appendix Summary	330
Appendix B	Plant Models	332
B.1	Introduction	332
B.2	Ground Vehicles and Robots	332
B.3	Aircraft and Helicopters	343
B.4	Spacecraft	349
	References	354
	Index	359

CHAPTER ONE

Static Estimation

1.1 Random Scalars

A random scalar variable x is described completely by its *density function* $p(x)$, where $p(x^{(0)}) dx$ is the probability that x will take on values between $x^{(0)}$ and $x^{(0)} + dx$ (see Fig. 1.1). Clearly $p(x) \geq 0$ and

$$\int_{-\infty}^{\infty} p(x) dx = 1. \quad (1.1)$$

It is described approximately by its *mean value* \bar{x} and its *variance* σ^2 , where

$$\bar{x} \equiv E(x) = \int_{-\infty}^{\infty} xp(x) dx, \quad (1.2)$$

$$\sigma^2 \equiv E(x - \bar{x})^2 = \int_{-\infty}^{\infty} (x - \bar{x})^2 p(x) dx \quad (1.3)$$

Here σ is called the *standard deviation* or *root-mean-square (RMS) deviation* from the mean value.

Two important scalar density functions are (see Fig. 1.2):

- Uniform density:

$$p(x) = \begin{cases} 1/2a & \text{if } |x - \bar{x}| < a, \\ 0 & \text{if } |x - \bar{x}| > a, \end{cases} \quad (1.4)$$

which implies that the standard deviation is $\sigma = a/\sqrt{3}$.

- Gaussian density:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(x - \bar{x})^2}{2\sigma^2}\right]. \quad (1.5)$$

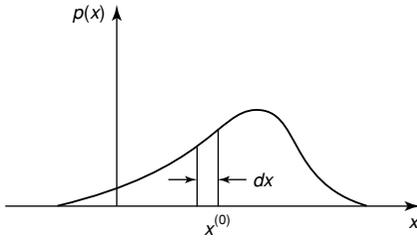


FIGURE 1.1 Density Function of a Random Scalar Variable

1.2 Random Vectors

A random vector $x = [x_1, x_2, \dots, x_n]^T$ is described completely by its *density function* $p(x) = p(x_1, x_2, \dots, x_n)$, where $p(x^{(0)}) dx_1 dx_2 \cdots dx_n$ is the probability that x will take on values in the volume element $x_i^{(0)} < x_i < x_i^{(0)} + dx_i$, $i = 1, 2, \dots, n$ (see Fig. 1.3). Clearly $p(x) \geq 0$ and

$$\int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} p(x) dx_1 dx_2 \cdots dx_n = 1. \quad (1.6)$$

The vector x is described approximately by its *mean value* \bar{x} and its *covariance matrix* X , namely

$$\bar{x} \equiv E(x) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} xp(x) dx_1 dx_2 \cdots dx_n, \quad (1.7)$$

$$\begin{aligned} X &\equiv E(x - \bar{x})(x - \bar{x})^T \\ &= \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} (x - \bar{x})(x - \bar{x})^T p(x) dx_1 dx_2 \cdots dx_n. \end{aligned} \quad (1.8)$$

The covariance matrix

$$X = \begin{bmatrix} X_{11} & \cdots & X_{1n} \\ \vdots & & \vdots \\ X_{n1} & \cdots & X_{nn} \end{bmatrix} \quad (1.9)$$

is symmetric, i.e., $X_{ij} = X_{ji}$. Note that $X_{ii} = \sigma_i^2$, and X_{ij} is the *covariance* or *correlation* of x_i and x_j . If $X_{ij} = 0$, then x_i and x_j are said to be *uncorrelated*.

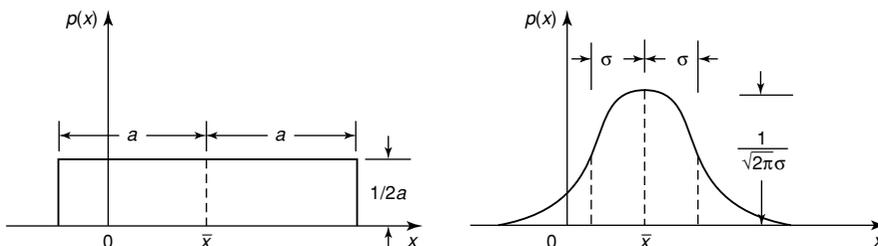
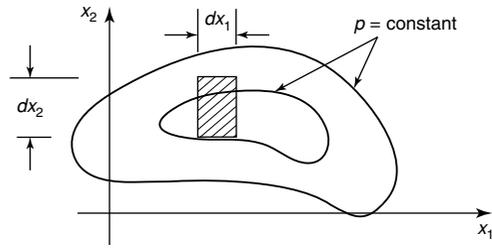


FIGURE 1.2 Uniform and Gaussian Density Functions for a Random Scalar Variable

FIGURE 1.3 Density Function of a Random Two-Vector



Two important density functions are:

- Uniform density:

$$p(x) = \begin{cases} 1/V & \text{inside } \Gamma, \\ 0 & \text{otherwise,} \end{cases}$$

where V is the volume inside Γ . In two dimensions, the “volume” is an area and Γ is a closed curve (see Fig. 1.4). The mean value of the vector (\bar{x}_1, \bar{x}_2) is at the centroid of the area inside Γ . The variances X_{11}, X_{22} are the second moments of area inside Γ about centroidal x_1, x_2 axes; the covariance X_{12} is the corresponding cross-moment of area. The principal axes y_1, y_2 , rotated about the centroid, are such that $Y_{12} = 0$.

- Gaussian density for an n -dimensional vector:

$$p(x) = \frac{1}{(2\pi)^{n/2} |X|^{1/2}} \exp\left[-\frac{1}{2}(x - \bar{x})^T X^{-1}(x - \bar{x})\right].$$

The contours of constant p are hyperellipsoids defined by

$$\left(\frac{y_1}{\sigma_1}\right)^2 + \dots + \left(\frac{y_n}{\sigma_n}\right)^2 = m^2.$$

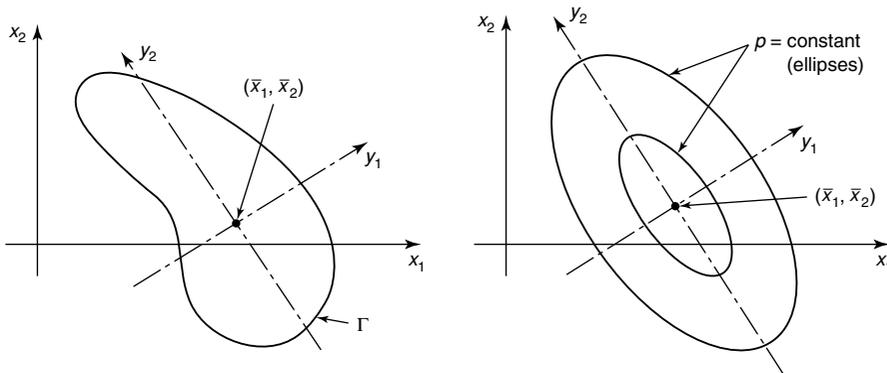


FIGURE 1.4 Two-Dimensional Uniform and Gaussian Density Functions

The probability of x having values inside one of these ellipsoids is

n	$m=1$	2	3
1	.683	.955	.997
2	.394	.865	.989
3	.200	.739	.971

The *marginal density function* is defined as

$$p(x_1, x_2, \dots, x_j) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} p(x_1, x_2, \dots, x_n) dx_{j+1} \cdots dx_n, \quad (1.10)$$

where $j < n$.

The *conditional density function* (CDF) $p(x_2 | x_1)$ is defined so that $p[x_2^{(0)} | x_1^{(0)}] dx_2$ is the probability that x_2 will take on values between $x_2^{(0)}$ and $x_2^{(0)} + dx_2$, given that x_1 has the value $x_1^{(0)}$. It follows that

$$p(x_1, x_2) = p(x_1 | x_2) \cdot p(x_2) \quad (1.11)$$

$$= p(x_2 | x_1) \cdot p(x_1). \quad (1.12)$$

From this follows *Bayes's rule*, namely that

$$p(x_2 | x_1) = p(x_1 | x_2) \cdot p(x_2) / p(x_1). \quad (1.13)$$

x_1 and x_2 are said to be *independent random vectors* if and only if

$$p(x_1, x_2) = p(x_1) \cdot p(x_2), \quad (1.14)$$

which implies that

$$p(x_1 | x_2) = p(x_1), \quad p(x_2 | x_1) = p(x_2). \quad (1.15)$$

Note that independence implies no correlation but *not vice versa*.

Linear Transformations of Gaussian Random Vectors

A very important property of gaussian random vectors is that *a linear combination of gaussian random vectors is also a gaussian vector*. No other density has this property.

Stated analytically, if x and y are independent gaussian random vectors, i.e., $x = N(\bar{x}, X)$, $y = N(\bar{y}, Y)$, and

$$z = Ax + By, \quad (1.16)$$

where A, B are known matrices, then it follows that $z = N(\bar{z}, Z)$ with

$$\bar{z} = A\bar{x} + B\bar{y}, \quad Z = AXA^T + BYB^T. \quad (1.17)$$

(1.17) follows simply from the linearity of the expectation operator and the independence of x and y . To show that z is still gaussian is more complicated (see for example Ref. He).

1.3 Generating Gaussian Vectors

A gaussian random vector with a specified mean value and a specified covariance matrix can be generated using a zero-mean uncorrelated gaussian random vector by using the *square root of a positive-definite symmetric matrix* (see below).

Suppose we wish to generate an n -dimensional gaussian random vector

$$x = N[\bar{x}, X]. \quad (1.18)$$

We first factor X into $S^T S$ using the MATLAB command `SQRTM` or `CHOL` (see the following subsection). Then we generate an n -dimensional zero-mean gaussian vector z with unit covariance matrix I_n , i.e., a vector whose components are uncorrelated and have unit variances; this is done in MATLAB using the command `RANDN`. The desired vector is then

$$x = \bar{x} + S^T z, \quad (1.19)$$

because

$$E\{[S^T z][z^T S]\} = S^T E[zz^T] S = S^T I_n S \equiv X. \quad (1.20)$$

The Square Root of a Positive-Definite Symmetric Matrix

The square root of a positive-definite symmetric matrix is not unique. Two commonly used square roots are the *symmetric square root* and the *triangular (or Cholesky) square root*.

The Cholesky square root may be interpreted as successive “completing of squares” row by row. One may start with the first row and work to the last row, which produces an upper-triangular square root, or vice versa which yields a lower-triangular square root. This is illustrated by an example below.

EXAMPLE 1.3.1 SQUARE ROOT OF A 3-BY-3 SYMMETRIC MATRIX

Consider the matrix

$$P = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 8 & 2 \\ 3 & 2 & 14 \end{bmatrix}, \quad (1.21)$$

which, as the weighting matrix in a quadratic form, gives

$$J = x^T P x = x_1^2 + 4x_1x_2 + 6x_1x_3 + 8x_2^2 + 4x_2x_3 + 14x_3^2. \quad (1.22)$$

This expression can be factored into the sum of three squared quantities with no cross-products, starting with x_1 , as follows:

$$\begin{aligned} J &= (x_1 + 2x_2 + 3x_3)^2 - (2x_2 + 3x_3)^2 + 8x_2^2 + 4x_2x_3 + 14x_3^2 \\ &= y_1^2 + 4x_2^2 - 8x_2x_3 + 5x_3^2 \quad (y_1 \triangleq x_1 + 2x_2 + 3x_3) \\ &= y_1^2 + (2x_2 - 2x_3)^2 - 4x_3^2 + 5x_3^2 \\ &= y_1^2 + y_2^2 + y_3^2 \quad (y_2 \triangleq 2x_2 - 2x_3, \quad y_3 \triangleq x_3) \\ &= y^T y, \end{aligned}$$

where

$$y = Ux, \quad U = \begin{bmatrix} 1 & 2 & 3 \\ 0 & 2 & -2 \\ 0 & 0 & 1 \end{bmatrix} \Rightarrow P = U^T U. \quad (1.23)$$

U is an *upper-triangular* matrix.

To find the lower-triangular square root, reverse the order of the quadratic form:

$$J = 14x_3^2 + 4x_3x_2 + 6x_3x_1 + 8x_2^2 + 4x_2x_1 + x_1^2. \quad (1.24)$$

Completing the square of the first three terms gives

$$J = 14 \left(x_3 + \frac{2}{14}x_2 + \frac{3}{14}x_1 \right)^2 - \frac{4}{14}x_2^2 - \frac{9}{14}x_1^2 - \frac{12}{14}x_1x_2 + 8x_2^2 + 4x_2x_1 + x_1^2, \quad (1.25)$$

and adding the remainder terms gives

$$J = y_3^2 + \frac{54}{7}x_2^2 + \frac{22}{7}x_2x_1 + \frac{5}{14}x_1^2, \quad y_3 \triangleq \sqrt{14} \left(x_3 + \frac{2}{14}x_2 + \frac{3}{14}x_1 \right). \quad (1.26)$$

Completing the square of the second and third terms in (1.26) gives

$$J = y_3^2 + \frac{54}{7} \left(x_2 + \frac{11}{54}x_1 \right)^2 - \frac{54}{7} \left(\frac{11}{54} \right)^2 x_1^2 + \frac{5}{14}x_1^2, \quad (1.27)$$

and adding the remainder terms gives

$$J = y_3^2 + y_2^2 + \frac{1}{27}x_1^2, \quad y_2 \triangleq \sqrt{\frac{54}{7}} \left(x_2 + \frac{11}{54}x_1 \right). \quad (1.28)$$

Finally,

$$J = y_3^2 + y_2^2 + y_1^2, \quad y_1 \triangleq \frac{1}{\sqrt{27}}x_1. \quad (1.29)$$

Hence

$$y = Lx, \quad L = \begin{bmatrix} 1/\sqrt{27} & 0 & 0 \\ 11/\sqrt{378} & \sqrt{54/7} & 0 \\ 3/\sqrt{14} & 2/\sqrt{14} & \sqrt{14} \end{bmatrix} \Rightarrow P = L^T L. \quad (1.30)$$

Efficient codes have been developed for both the symmetric and the Cholesky square roots; both are available as commands in MATLAB, Matrix-X, and Control-C.

PROBLEMS

1.3.1 LIKELIHOOD ELLIPSES IN TWO DIMENSIONS

Consider a normally distributed two-dimensional vector with $\bar{x} = 0$ and

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix} = \begin{bmatrix} 4 & 1 \\ 1 & 1 \end{bmatrix}.$$

(a) Show that the eigenvalues of this covariance matrix are $\sigma_1^2 = 4.30$, $\sigma_2^2 = 0.70$, and the corresponding eigenvectors are the columns of T where

$$T \triangleq \begin{bmatrix} .957 & -.290 \\ .297 & .957 \end{bmatrix}.$$

(b) The likelihood ellipses are defined by constant values of ℓ , where

$$\ell^2 = [x_1 \quad x_2] \begin{bmatrix} 4 & 1 \\ 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix},$$

or, in principal axes,

$$\ell^2 = [y_1 \quad y_2] \begin{bmatrix} 4.30 & 0 \\ 0 & 0.70 \end{bmatrix}^{-1} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \equiv \frac{y_1^2}{4.30} + \frac{y_2^2}{0.70},$$

where $x \triangleq Ty$. Plot the likelihood ellipses for $\ell = 1, 2, 3$. Check your plots using PTELPs listed in Section 1.5 and on the disk.

(c) Show that the probability of finding x inside the $\ell = l$ ellipse is 0.394, inside the $\ell = 2$ ellipse is 0.865, and inside the $\ell = 3$ ellipse is 0.989.

1.3.2 RAYLEIGH, BOLTZMANN, AND χ^2 DENSITY FUNCTIONS

Let v be an n -dimensional gaussian random vector with components v_1, v_2, \dots, v_n with zero mean, and suppose that the components are uncorrelated and all have variance σ^2 .

(a) For $n = 2$ show that the probability density function for the magnitude of v is

$$p(v) = \frac{v}{\sigma^2} \exp\left(-\frac{v^2}{2\sigma^2}\right),$$

where $v^2 = v_1^2 + v_2^2$. This is called the *Rayleigh density function*.

Hint: Change to cylindrical coordinates and use the circular symmetry.

(b) For $n = 3$ show that the probability density function for the magnitude of v is

$$p(v) = \frac{2}{\sqrt{\pi}} \frac{v^2}{\sigma^3} \exp\left(-\frac{v^2}{2\sigma^2}\right).$$

where $v^2 = v_1^2 + v_2^2 + v_3^2$. In the kinetic theory of gases, $p(v)$ is called the Maxwell-Boltzmann density function, where v is the velocity magnitude of molecules and $\sigma^2 = kT/m$, with T is the temperature, k is Boltzmann's constant, and m is the mass of a molecule.

Hint: Change to spherical coordinates and use the spherical symmetry.

(c) For arbitrary n show that the probability density function for the magnitude of v is

$$p(v) = \frac{2\pi^{n/2}}{\Gamma(n/2)} v^{(n-1)} \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^n \exp\left(-\frac{v^2}{2\sigma^2}\right) dv.$$

This density is called the χ^2 (chi-squared) density in n dimensions.

Hint: Change to hyperspherical coordinates and use the hyperspherical symmetry. The volume element of an n -dimensional hyperspherical shell of hyperradius v and hyperthickness dv is (Ref. He, p. 220)

$$\frac{2\pi^{n/2}}{\Gamma(n/2)} v^{(n-1)} dv,$$

where $\Gamma(m + 1) = m\Gamma(m)$ and $\Gamma(1) = 1$, $\Gamma(1/2) = \sqrt{\pi}$.

1.3.3 SQUARE ROOT OF 3-BY-3 SYMMETRIC MATRIX

Given

$$P = \begin{bmatrix} 1 & -2 & -1 \\ -2 & 6 & 2 \\ -1 & 2 & 10 \end{bmatrix}.$$

- Find the upper-triangular Cholesky square root of P by completing the square. Check your result by multiplication and by the use of the `CHOL` command in MATLAB.
- Find the lower-triangular Cholesky square root of P by completing the square.

Hint: Reverse the order of the quadratic form.

Check your result by multiplication and by the use of the `CHOL` command in MATLAB.

1.3.4 GAUSSIAN SAMPLE FROM UNIFORM SAMPLES

MATLAB has a random number generator `RAND` where the number density is approximately uniform from 0 to 1. For simulation of discrete Gauss–Markov processes, one would like to have a random number generator where the density is gaussian with zero mean and unit variance. Adding N numbers with a uniform density produces numbers with a density that tends toward a gaussian density, a special case of the central limit theorem. Thus, a MATLAB code to generate numbers with an approximately gaussian density having a zero mean and a unit variance from numbers having a uniform distribution between 0 and 1 is

```
w=0; for k=1:12; w=w+rand; end; w=w-6;
```

- Verify analytically that this algorithm produces numbers with zero mean and unit variance, if the random number generator produces numbers with uniform density on the interval from 0 to 1.
- Using MATLAB, generate 1000 numbers using the code in (a), and make a histogram of the numbers using an interval of 0.3 (see the MATLAB code `HIST.M`). Also plot on the histogram

$$\frac{1000(0.3)}{\sqrt{2\pi}} \exp\left(-\frac{w^2}{2}\right),$$

which is the corresponding gaussian density with zero mean and unit variance.

- MATLAB also has a gaussian number generator `RANDN`. Redo (b) with this method of generating gaussian numbers.

1.4 Static Linear Estimation

Before considering estimation for dynamic systems, we consider the simpler case of estimation for static systems. Suppose we wish to find an estimate of a vector x , given a vector of measurements z that is linearly related to x :

$$z = Cx + v, \tag{1.31}$$

where C is known, and the measurement error vector v is gaussian with zero mean

and covariance matrix V , i.e.,

$$v = N(0, V). \quad (1.32)$$

We are also given a prior estimate of x as a gaussian vector with mean value \bar{x} and covariance matrix \bar{P} , i.e.,

$$x = N(\bar{x}, \bar{P}). \quad (1.33)$$

Following Kalman's nomenclature (Ref. Ka1), we shall call the posterior estimate (after incorporating the measurements) \hat{x} and the corresponding covariance matrix \hat{P} , i.e.,

$$x = N(\hat{x}, \hat{P}). \quad (1.34)$$

A simple derivation of the optimal estimate uses Bayes's rule to find the value of x that maximizes the likelihood function

$$p(x | z) = \frac{p(z | x)p(x)}{p(z)}. \quad (1.35)$$

From the data given above

$$p(z | x) = N(Cx, V), \quad p(x) = N(\bar{x}, \bar{P}), \quad p(z) = N(C\bar{x}, C\bar{P}C^T + V). \quad (1.36)$$

Thus

$$p(x | z) \sim \exp(-J), \quad (1.37)$$

where

$$2J = (z - Cx)^T V^{-1}(z - Cx) + (x - \bar{x})^T \bar{P}^{-1}(x - \bar{x}), \quad (1.38)$$

because $p(z)$ is not a function of x . Thus maximizing $p(x | z)$ corresponds to minimizing the quadratic form J with respect to x . The differential of J is

$$dJ = dx^T [\bar{P}^{-1}(x - \bar{x}) - C^T V^{-1}(z - Cx)], \quad (1.39)$$

and, for a stationary value of J with arbitrary dx , the coefficient of dx^T in (1.39) must be zero, which implies that the most likely value of x is

$$\hat{x} = \bar{x} + K(z - C\bar{x}), \quad (1.40)$$

where

$$K = \hat{P}C^T V^{-1}, \quad \hat{P}^{-1} = \bar{P}^{-1} + C^T V^{-1}C. \quad (1.41)$$

Another derivation of (1.40)–(1.41) is to complete the square in in (1.38), which gives

$$2J = (x - \hat{x})^T \hat{P}^{-1}(x - \hat{x}) + \text{constant},$$

which implies

$$x = N(\hat{x}, \hat{P}) \text{ after measurement,}$$

or

$$\hat{x} = E(x | z), \quad \hat{P} = E[(x - \hat{x})(x - \hat{x})^T | z]. \quad (1.42)$$

Still another approach is to use triangular or $U^T DU$ factorization (see Appendix A).

\bar{P} and/or V Singular – the Matrix Inversion Lemma

(1.41) may be expressed in another form, which is useful when either \bar{P} or V (or both) are singular, using the *matrix inversion lemma*. This lemma states that if

$$A^{-1} = B^{-1} + C^T D^{-1} C, \quad (1.43)$$

then

$$A = B - BC^T(D + CBC^T)^{-1}CB. \quad (1.44)$$

(1.44) involves inverting only one matrix, which may have a lower dimension than A and B ; the lemma is easily verified by multiplying (1.43) and (1.44) to yield the identity matrix.

Using this lemma, (1.41) can be written as

$$K \triangleq \bar{P}C^T(V + C\bar{P}C^T)^{-1}, \quad \hat{P} = \bar{P} - K(V + C\bar{P}C^T)K^T. \quad (1.45)$$

EXAMPLE 1.4.1 LINEAR ESTIMATION OF TWO PARAMETERS WITH THREE MEASUREMENTS

Consider three lines in a two-dimensional space, $z = Cx + v$, where

$$C = \begin{bmatrix} 1/\sqrt{5} & -2/\sqrt{5} \\ 1/\sqrt{2} & 1/\sqrt{2} \\ 0 & 1 \end{bmatrix}, \quad V = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0.01 \end{bmatrix},$$

the i th row of C is a unit vector perpendicular to the i th line, and z_i is the measured distance to the i th line from the origin. We generate a sample measurement error v using MATLAB, namely `v=sqrt(V)*randn(3,1)` with `V=.01*eye(3)` (see `e01_4_1.m` in the Examples folder on the disk). For $x = [1 \ 2]^T$, we obtain

$$z = [-1.4790 \quad 2.1720 \quad 1.9629]^T,$$

The batch method [no prior estimate ($\bar{P}^{-1} = 0$)] gives

$$\hat{P} = (C^T V^{-1} C)^{-1} = \begin{bmatrix} .0144 & -.0006 \\ -.0006 & .0044 \end{bmatrix}, \quad \hat{x} = \hat{P}C^T V^{-1} z = \begin{bmatrix} 0.9557 \\ 2.0548 \end{bmatrix},$$

so the RMS estimate error in $\hat{x}(1)$ is $0.1199 = (0.0144)^{1/2}$, and in $\hat{x}(2)$ is $0.0661 = (0.0044)^{1/2}$.

The sequential method uses two of the measurements, say the first and second, to obtain a *prior estimate*, which gives

$$\bar{x} = \begin{bmatrix} 0.9454 \\ 2.1263 \end{bmatrix}, \quad \bar{P}_0 = (C_0^T V_0^{-1} C_0)^{-1} = \begin{bmatrix} .0144 & -.0011 \\ -.0011 & .0078 \end{bmatrix}.$$

Then use the third measurement to give the posterior estimate. Using (1.40) and (1.41), we have

$$\hat{P} = (\bar{P}_0^{-1} + C_3^T V_3^{-1} C_3)^{-1} = \begin{bmatrix} .0144 & -.0006 \\ -.0006 & .0044 \end{bmatrix},$$

$$\hat{x} = \bar{x} + \hat{P} C_3^T V_3^{-1} (z_3 - C_3 \bar{x}) = \begin{bmatrix} 0.9557 \\ 2.0548 \end{bmatrix},$$

which agree exactly with the batch method above.

Figure 1.5 shows the batch estimate with the three lines and the 39% likelihood ellipse. Figure 1.6 shows the sequential estimate. Note that line 3 gives no information about $x(1)$, so that the variance of the estimate in the x_1 direction is unchanged (i.e., the projection of the 39% likelihood ellipse on the x_1 axis is unchanged).

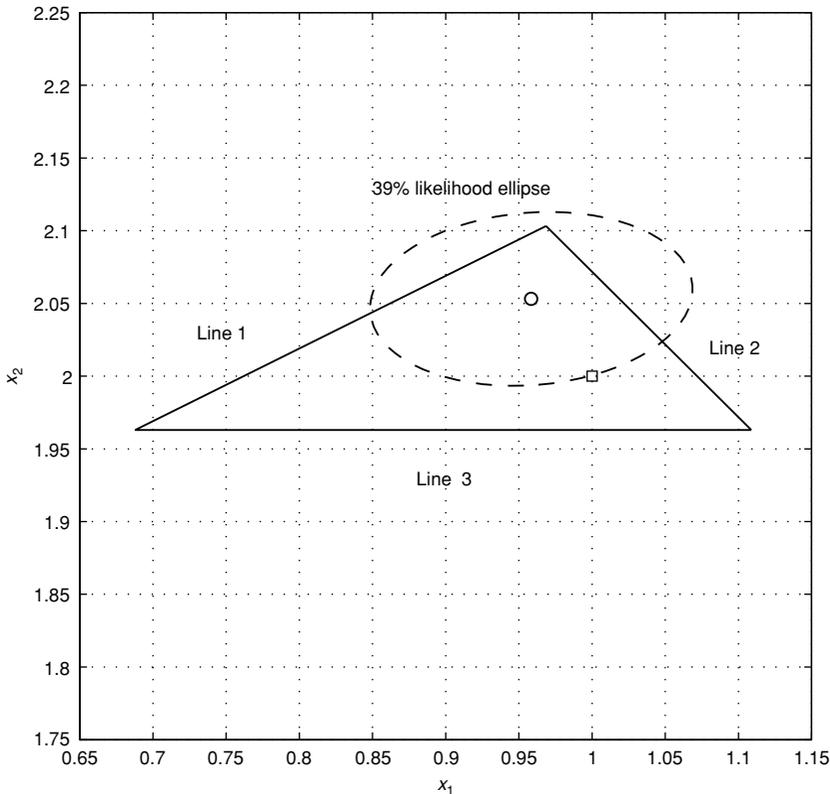


FIGURE 1.5 2-D Batch Position Estimation from Three Measured Lines

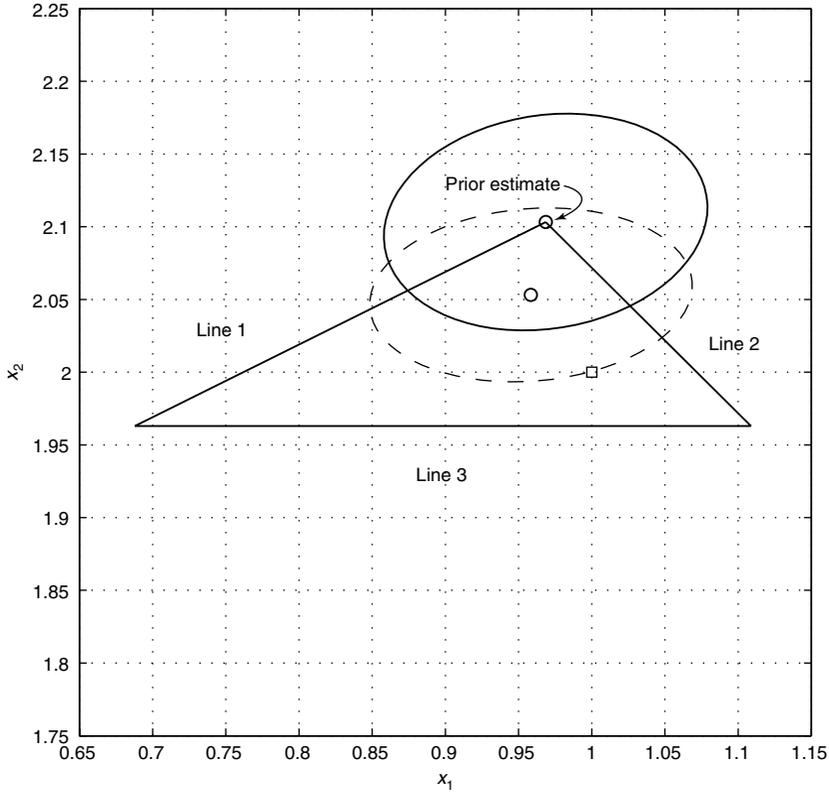


FIGURE 1.6 2-D Sequential Position Estimation; Prior Estimate Using Lines 1 and 2; Posterior Estimate Using Line 3

PROBLEMS

1.4.1 2-D POSITION ESTIMATION FROM THREE MEASURED LINES

- Repeat Example 1.4.1 using measurements 1 and 3 for the prior estimate.
- Repeat Example 1.4.1 using measurements 2 and 3 for the prior estimate.

1.4.2 3-D POSITION ESTIMATION FROM MEASURED DISTANCES TO FOUR PLANES

Consider four planes in a three-dimensional space that do not intersect at the same point: $z = Cx + v$, where C is a 4-by-3 matrix whose i th row is a unit vector perpendicular to the i th plane, z_i is the distance to the i th plane from the origin, and v is a zero-mean gaussian random vector with covariance matrix V . Here we take

$$C = \begin{bmatrix} -2/\sqrt{5} & 0 & 1/\sqrt{5} \\ 3/\sqrt{35} & 5/\sqrt{35} & 1/\sqrt{35} \\ 3/\sqrt{14} & -2/\sqrt{14} & 1/\sqrt{14} \\ 0 & 0 & 1 \end{bmatrix}, \quad V = 0.04 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

- Find z for $x = [1 \ 2 \ 3]^T$ and a random sample of v from `RANDN` in MATLAB.
- Use the batch method to find \hat{x} and \hat{P} from z , C , and V .
- Take the four planes three at a time to find the four vertices of the tetrahedron that

encloses \hat{x} . Using the MATLAB command `PLOT3`, plot x , \hat{x} , the six edges of the tetrahedron, and the axes of the one- σ ellipsoid with center at \hat{x} . Also plot the two side views and the top view of these objects.

- (d) Use the sequential method to find \hat{x} and \hat{P} . Use three of the measurements to make an initial estimate, and the fourth measurement to update this estimate. Do this for all four of the possible ways, verifying that they all give the same result as the batch estimate in (b).

1.5 Static Nonlinear Estimation

Many estimation problems are *nonlinear* rather than linear. For example, we may have

$$z = c(x) + v, \tag{1.46}$$

where $c(x)$ is a known nonlinear function of x and v is a random vector. Suppose we believe that v has zero mean value and covariance matrix V . Then a reasonable criterion for estimating x , given the measurement vector z , is to find x to minimize the nonlinear function

$$J = \frac{1}{2}(x - \bar{x})^T \bar{P}^{-1}(x - \bar{x}) + \frac{1}{2}[z - c(x)]^T V^{-1}[z - c(x)]. \tag{1.47}$$

This is a *nonlinear weighted least-squares fit*, an idea first given by Gauss in 1795.

Taking a differential of J , we have

$$dJ = \{(x - \bar{x})^T \bar{P}^{-1} - [z - c(x)]^T V^{-1}C\} dx + \frac{1}{2} dx^T \left\{ \bar{P}^{-1} + C^T V^{-1}C - [z - c(x)]^T V^{-1} \frac{dC}{dx} \right\} dx + \dots, \tag{1.48}$$

where

$$C \triangleq dc/dx, \tag{1.49}$$

which is evaluated at x . The term containing the second derivative matrix $dC/dx \equiv d^2c/dx^2$ is usually negligible, since z is close to $c(x)$. Thus, minimizing dJ with respect to dx yields $dx = -\hat{P} \cdot \text{gr}$, where $\hat{P} \triangleq (\bar{P}^{-1} + C^T V^{-1}C)^{-1}$, $\text{gr} = \bar{P}^{-1}(x - \bar{x}) - C^T V^{-1}[z - c(x)]$. x can be determined using a Newton-Raphson type algorithm such as the one given below.

- Guess x .
- * Calculate $\bar{z} = c(x)$ and $C = dc/dx$ (evaluated at x).
- Calculate $\hat{P} \triangleq [\bar{P}^{-1} + C^T V^{-1}C]^{-1}$ and $\text{gr} \triangleq \bar{P}^{-1}(x - \bar{x}) - C^T V^{-1}[z - c(x)]$.
- $dx = -\hat{P} \cdot \text{gr}$.
- If $|dx| < \epsilon$ then stop.
- Replace x by $x + dx$ and go to (*).

Here \hat{P} is the posterior error covariance of the estimate of x . This algorithm is implemented in the MATLAB code `NL_EST` (included in the `OPTEST` Toolbox on

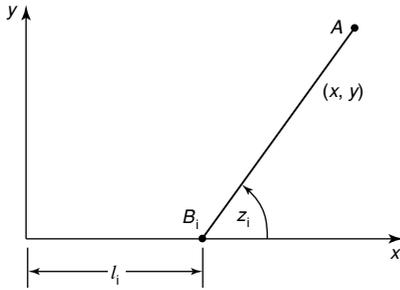


FIGURE 1.7 Position Estimation using Angle Measurements on a Base Line

the disk). The user must supply a subroutine that calculates $\bar{z} = c(x)$ and $C = dc/dx$ evaluated at x as illustrated in Example 1.5.1 below.

EXAMPLE 1.5.1 NONLINEAR ESTIMATE WITH ANGLE MEASUREMENTS

We wish to estimate the location (x, y) of a point A in a plane by angle measurements z_i from several points B_i , $i = 1, 2, \dots, N$, on a base line (see Fig. 1.7).

The angle measurements z_i are related to the locations of A and B_i by the nonlinear relations

$$z_i = \tan^{-1} \frac{y}{x - \ell_i} + v_i,$$

where v_i is a random error made in the angle measurement. We assume that

$$E(v_i) = 0, \quad E(v_i v_j) = \begin{cases} V_i, & i = j, \\ 0, & i \neq j. \end{cases}$$

Suppose that there are three measurements, and the data are:

i	ℓ_i (ft)	z_i (deg)	V_i (deg ²)
1	0	30.1	.01
2	500	45.0	.01
3	1000	73.6	.04

The codes `pos_est` and `e01_5_1` (listed below and on the disk) calculate \bar{z} and C and the estimate, giving

$$\begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix} = \begin{bmatrix} 1204.8 \\ 701.8 \end{bmatrix} \text{ft}, \quad \hat{P} = \begin{bmatrix} 10.98 & 10.11 \\ 10.11 & 12.60 \end{bmatrix}:$$

```
% Script e01.5.1.m; nonlinear position estimation with angle
% measurements; x=[x1 x2]'; z(i)=atan(x2/(x1-el(i))).
%
m=pi/180; z=m*[30.1 45.0 73.6]'; V=(.1*m)^2*diag([1 1 4]); el=1000*
[0 .5 1]'; xb=[1100 600]'; Pb=1e12*eye(2);
[xh,Ph]=nl_est('pos.est',xb,Pb,z,V); x1=xh(1); x2=xh(2);

function [zb,C]=pos_est(x)
% Subroutine for e01.5.1.
%
```

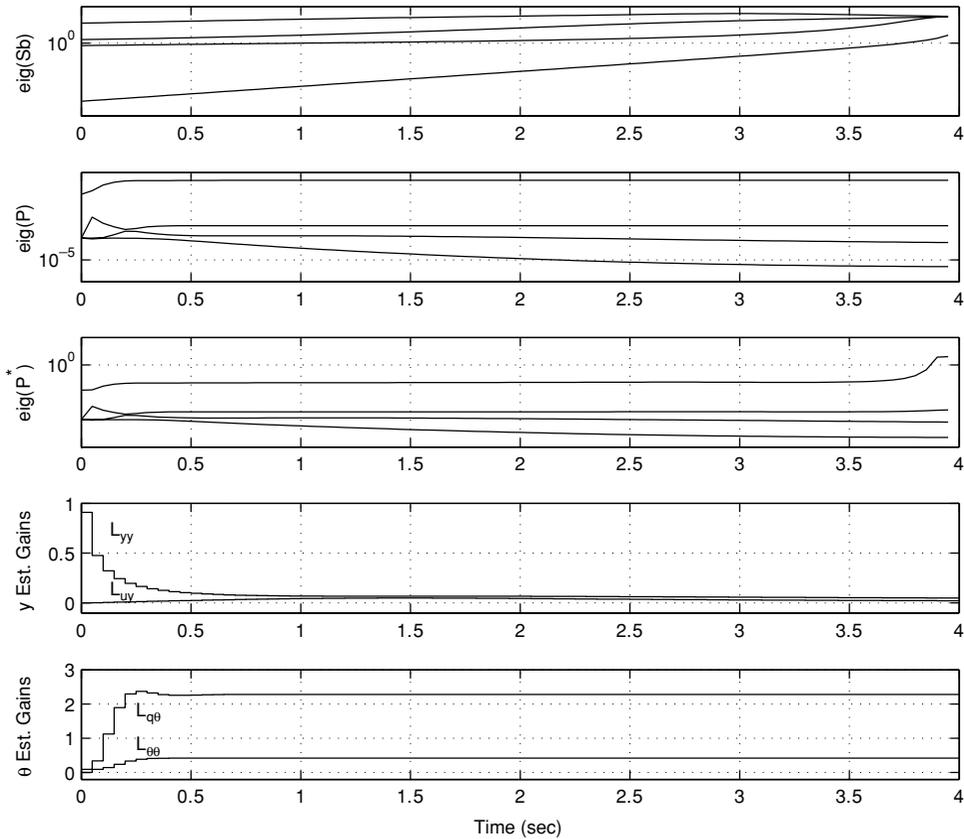


FIGURE 1.8 Numerical Example of Position Estimation using Angle Measurements

```

el=1000*[0 .5 1]; for i=1:3, zb(i)=atan(x(2)/(x(1)-el(i)));
    d(i)=(x(1)-el(i))/(cos(zb(i)))^2; C(i,:)=
    [-tan(zb(i)) 1]/d(i);
end; zb=zb';
    
```

The square roots of the eigenvalues of \hat{P} are [4.68, 1.28] ft, and the corresponding eigenvector directions are [47.0, -43.0] deg. Thus the one- σ likelihood ellipse has semiaxes [4.68, 1.28] ft, and these axes are at [47.0, -43.0] deg as shown in Fig. 1.8. The three- σ ellipse is also shown, along with the three lines of sight from B_1 , B_2 , and B_3 . A code `PLTELP` is included in the `OPTTEST` Toolbox that plots one- σ ellipses given the coordinates of the center and the covariance matrix P .

PROBLEMS

1.5.1 A SHIP COASTAL NAVIGATION PROBLEM

A navigator has an estimated position of his ship 3.00 km directly west of a landmark A and 5.00 km from another landmark B , with an estimated accuracy of ± 0.3 km in the north-south direction and ± 0.2 km in the east-west direction. He then measures the