Integer Parameter Estimation in Linear Models with Applications to GPS[1]

Arash Hassibi & Stephen Boyd
Information Systems Laboratory
Department of Electrical Engineering
Stanford, CA 94305-4055 USA
arash@ISL.Stanford.EDU & boyd@ISL.Stanford.EDU

Abstract

We consider parameter estimation in linear models when some of the parameters are known to be integers. Such problems arise, for example, in positioning using phase measurements in the global positioning system (GPS.) Given a linear model, we address two problems:

1. The problem of *estimating* the parameters.
2. The problem of *verifying* the parameter estimates.

Under Gaussian measurement noise:

- Maximum likelihood estimates of the parameters are given by solving an integer least-squares problem. Theoretically, this problem is very difficult to solve ($\mathcal{NP}$-hard.)
- Verifying the parameter estimates (computing the probability of correct integer parameter estimation) is related to computing the integral of a Gaussian PDF over the Voronoi cell of a lattice. This problem is also very difficult computationally.

However, by using a polynomial-time algorithm due to Lenstra, Lenstra, and Lovász (LLL algorithm):

- The integer least-squares problem associated with estimating the parameters can be solved efficiently in practice.
- Sharp upper and lower bounds can be found on the probability of correct integer parameter estimation.

We conclude the paper with simulation results that are based on a GPS setup.

Key words: Integer parameter estimation, Linear model, Integer least-squares, GPS.

[1]Research supported in part by AFOSR (under F49620-95-1-0318 and AASERT grant F49620-93-1-0411), and NSF (under ECS-9222391 and ECS-9222391). The US Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

1 Problem Statement

1.1 Setup

Throughout this paper, we assume that the observation $y \in \mathbf{R}^n$ is related to the unknown vectors $x \in \mathbf{R}^p$ (*real*) and $z \in \mathbf{Z}^q$ (*integer*) through

$$y = Ax + Bz + v, \tag{1}$$

where $A \in \mathbf{R}^{n\times p}$ (full column rank) and $B \in \mathbf{R}^{n\times q}$ are known matrices. The measurement noise $v \in \mathbf{R}^n$ is assumed to be Gaussian with zero-mean and covariance matrix identity, *i.e.*, $N(0, I_{n\times n})$. This can be assumed without loss of generality, as we can always rescale equation (1) by the square root of the covariance matrix of v. Given y, A, and B, our goal is to *find* and *verify* estimates of the unknown parameters x and z. Some previous results are given in [6] and [7] and references therein.

1.2 Estimation problem

We consider maximum likelihood (ML) estimates x_{ML} and z_{ML} for x and z respectively that maximize the probability of observing y, *i.e.*,

$$(x_{\text{ML}}, z_{\text{ML}}) = \underset{(x,z)\,\in\,\mathbf{R}^p\times\mathbf{Z}^q}{\mathbf{argmax}} \left\{p_{Y|X,Z}(y|x,z)\right\}. \tag{2}$$

1.3 Verification problem

Since z is an integer-valued vector, we have the chance of estimating (or detecting) it exactly. As a result, for the verification step, we compute the probability of estimating z correctly, *i.e.*,

$$P_{\text{c}} = \mathbf{Prob}(z_{\text{ML}} = z), \tag{3}$$

Clearly, the larger this value the higher the reliability on the estimates x_{ML} and z_{ML}. Another reliability measure can be

$$\mathbf{Prob}(\|x - x_{\text{ML}}\| < \epsilon), \tag{4}$$

where $\|\cdot\|$ denotes the 2-norm. This gives us the probability of the real parameter estimate lying in a ball

of radius ϵ of its actual value. A combined reliability measure is

$$\begin{aligned}&\mathbf{Prob}(\|x - x_{\text{ML}}\| < \epsilon \ \ \& \ \ z_{\text{ML}} = z) = \\ &\mathbf{Prob}(\|x - x_{\text{ML}}\| < \epsilon \,|\, z_{\text{ML}} = z) \cdot \mathbf{Prob}(z_{\text{ML}} = z).\end{aligned} \tag{5}$$

In practical cases, ϵ is *small* only if $z_{\text{ML}} = z$, and therefore, (4) and (5) are *almost* equal.

2 Problem formulation

Since v is Gaussian we get

$$(x_{\text{ML}}, z_{\text{ML}}) = \underset{(x,z) \in \mathbf{R}^p \times \mathbf{Z}^q}{\mathbf{argmin}} \|y - Ax - Bz\|^2,$$

Using completion of squares arguments it can be shown that

$$\begin{aligned}(x_{\text{ML}}, z_{\text{ML}}) = \underset{(x,z) \in \mathbf{R}^p \times \mathbf{Z}^q}{\mathbf{argmin}} \Big\{ &\left(x - \hat{x}_{|z}\right)^T \Xi^{-1} \left(x - \hat{x}_{|z}\right) + \\ &(z - \hat{z})^T \Upsilon^{-1} (z - \hat{z}) + y^T \Delta y \Big\},\end{aligned} \tag{6}$$

where

$$\left\{ \begin{array}{rcl} \Xi &=& (A^T A)^{-1}, \\ \Upsilon &=& \left(B^T \left(I - A\Xi A^T\right) B\right)^{-1}, \\ \hat{x}_{|z} &=& \Xi A^T (y - Bz), \\ \hat{z} &=& \Upsilon B^T \left(I - A\Xi A^T\right) y, \end{array} \right. \tag{7}$$

and $\Delta > 0$ is a constant matrix. Clearly, from (6)

$$z_{\text{ML}} = \underset{z \in \mathbf{Z}^q}{\mathbf{argmin}} \left\{(z - \hat{z})^T \Upsilon^{-1} (z - \hat{z})\right\}, \tag{8}$$

$$x_{\text{ML}} = \hat{x}_{|z_{\text{ML}}}. \tag{9}$$

As a result, z_{ML} (and therefore x_{ML}) is given by the solution to a least-squares problem involving integer variables. Because of the integer nature of z, computing z_{ML} is very difficult and is known to be $\mathcal{NP}$-hard (cf. [1].)

In order to calculate the measures of reliability P_{c} and $\mathbf{Prob}(\|x - \hat{x}\| < \epsilon \,|\, z = z_{\text{ML}})$ we need to know the probability distributions of our estimates. Note that we can easily read the covariance matrices of $\hat{z}$ and $\hat{x}_{|z}$ from (6) as

$$\mathbf{cov}(\hat{z}) = \Upsilon \quad \text{and} \quad \mathbf{cov}\left(\hat{x}_{|z}\right) = \Xi.$$

Therefore

$$\hat{z} \sim N(z, \Upsilon) \ \ \text{and} \ \ \hat{x}_{|z} \sim N(x, \Xi), \tag{10}$$

so for example $\hat{z} = z + u$ where $u \sim N(0, \Upsilon)$. Multiplying both sides by $G := \Upsilon^{-1/2}$ (the whitening matrix of u) and by defining $\tilde{y} := G\hat{z}$ we get

$$\tilde{y} = Gz + \bar{u}, \quad \text{where} \ \ \bar{u} \sim N(0, I_{q \times q}). \tag{11}$$

Also, (8) can be written in terms of G and $\tilde{y}$ in the following equivalent form

$$z_{\text{ML}} = \underset{z \in \mathbf{Z}^q}{\mathbf{argmin}} \|\tilde{y} - Gz\|^2. \tag{12}$$

The set $\{Gz \mid z \in \mathbf{Z}^q\}$ is a lattice in $\mathbf{R}^q$. Equation (12) states that z_{ML} is found by computing the closest lattice point to the vector $\tilde{y}$. In addition, according to (11), $\tilde{y}$ is off from Gz by $\bar{u}$. Therefore, as long as $\bar{u}$ is small enough such that $\tilde{y}$ remains closer to Gz than anyother point in the lattice, the estimate of z is correct (*i.e.*, $z_{\text{ML}} = z$.) This is equivalent to $Gz + \bar{u}$ remaining inside the Voronoi cell of the lattice point Gz[1]. Thus

$$P_{\text{c}} = \mathbf{Prob}(\bar{u} \in V_0) \quad \text{where} \ \ \bar{u} \sim N(0, I_{q \times q}), \tag{13}$$

and V_0 is the Voronoi cell of the origin. According to (10), $\hat{x}_{|z} = x + w$ where $w \sim N(0, \Xi)$, and if $z = z_{\text{ML}}$ we have

$$\mathbf{Prob}(\|x - x_{\text{ML}}\| < \epsilon \,|\, z = z_{\text{ML}}) = \mathbf{Prob}(\|\Xi^{1/2} w\| < \epsilon), \tag{14}$$

which is equal to the probability of w falling inside the ellipsoid $\|\Xi^{1/2} x\| \leq \epsilon$.

Formulas (8), (9), (12), (13), and (14) give the estimates and their measures of reliability.

3 Verification Problem

As noted in §1, the probability of detecting the integer parameter z correctly, P_{c} in (13), can be used to verify the ML estimates. The reliability measure given in (5) also requires the calculation of P_c. Therefore, an essential step in verifying x_{ML} and z_{ML} is the calculation of P_{c}.

It turns out that computing P_{c} is very difficult computationally and therefore we are motivated to find fast ways to approximate P_{c} that enable real-time reliability tests instead. In fact, as we will see, easily computable upper and lower bounds on P_{c} exist which become tight as P_{c} approaches unity.

In §3.1 we see that the choice of G for a given lattice is highly nonunique. Although P_{c} only depends on the lattice and not on the specific choice of G, this is not true for the upper and lower bounds on P_{c}. Therefore, we can sharpen these bounds by optimizing over the family of admissible matrix representations for the lattice. We introduce the LLL algorithm that is an extremely useful tool for sharpening bounds on P_{c}, and as will become clear later, it is also useful for improving the efficiency of the algorithm for solving the integer least-squares problem for computing z_{ML}.

[1] The Voronoi cell of a point Gz in a lattice, is the set of all points in space that are closer to Gz than anyother point in the lattice.

3.1 Lattice Bases

Let us denote a lattice $L \in \mathbf{R}^q$ generated by the matrix $G = [g_1\, g_2 \cdots g_q] \in \mathbf{R}^{q\times q}$ by

$$L = \mathbf{L}(G) = \{Gz \mid z \in \mathbf{Z}^q\}.$$

The set of vectors $\{g_1, g_2, \ldots, g_q\}$ is called 'a' *basis* for L since all lattice points are linear combinations of integer multiples of these vectors. We say 'a' and not 'the' because a lattice L has infinitely many bases. The lattice generated by the matrix GF, where F is any *unimodular* matrix, and that of G are identical. Such matrices F have the property that the elements of F and F^{-1} are integers (or equivalently, the elements of F are integers and $\mathbf{det}F = \pm 1$.)

Of the many possible bases of a lattice, one would like to select one that is in some sense nice or simple, or *reduced*. There are many different notions of reduced bases in the literature; one of them consists of requiring that all its vectors be short in the sense that the product $\|g_1\|\,\|g_2\|\cdots\|g_q\|$ is minimum. This problem is known as the *minimum basis problem* and has been proved to be $\mathcal{NP}$-hard (cf. [1].)

Although the *minimum basis problem* is $\mathcal{NP}$-hard there exists a *polynomial-time algorithm* for computing a basis for a given lattice that is in some sense *reduced*. This algorithm is due to A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász (LLL) and is very efficient in practice. Given a lattice generator matrix G, the LLL algorithm gives a new lattice generator matrix GF with F unimodular such that GF has two nice properties. Roughly speaking these are:

- The columns of GF (the new basis vectors) are *almost* orthogonal.
- The $2-$norm of the columns of GF (the length of the new basis vectors) are not arbitrarily large.

For an exact discussion of these properties refer to Chapter 5 of [1]. The LLL algorithm is described in the appendix.

3.2 Calculating P_{c}

Computing P_{c} as given in (13) requires the calculation of a suitable description of V_0 the Voronoi cell of the origin in the lattice generated by G. This is possible of course, but it is very difficult computationally (cf. [8].) Adding to this the computational cost of the numerical algorithm to perform the integral of the Gaussian probability density function over V_0, we conclude that, although the exact calculation of P_{c} is possible, it requires extensive computation that might be infeasible in practice or not worthwhile. Therefore, finding easily computable bounds on P_{c} become of practical importance.

3.3 Computing upper and lower bounds on P_{c}

3.3.1 Upper bound using $|\mathbf{det}\, G|$: A well-known result of multi-dimensional geometry is that the volume of the parallelopiped formed by the basis vectors of a (nondegenerate) lattice is given by $|\mathbf{det}\, G|$. Since the Voronoi cells of the lattice points partition the space with the same density (cells per unit volume) as the parallelopiped cells, it follows that the volume of the Voronoi cells are also equal to $|\mathbf{det}\, G|$. Therefore, $|\mathbf{det}\, G|$ which is the volume of the region for integrating the noise probability density function, gives an idea of P_{c}, and, roughly speaking, the larger this value, the larger the probability of correct integer estimation. Of course, the shape of the Voronoi cell is also a major factor in P_{c}, however, when the variance of noise in every dimension is equal, an upper bound for P_{c} is found if we assume the Voronoi cell is a q-dimensional sphere.

The volume of a q-dimensional sphere of radius ρ is $\alpha_q \rho^q$ where $\alpha_q = \pi^{q/2}/,\, (q/2+1)$ (, $(\cdot)$ is the Gamma function.) Therefore, the radius of a q-dimensional sphere with volume $|\mathbf{det}\, G|$ is $\rho = \sqrt[q]{|\mathbf{det}\, G|/\alpha_q}$ and an upper bound on P_{c} becomes

$$P_{\mathrm{c,up}} = \mathbf{Prob}\left(\|v\| < \sqrt[q]{|\mathbf{det}\, G|/\alpha_q}\right),$$

with $v \sim N(0, I_{q\times q})$. The sum of squares of q independent zero-mean, unit variance normally distributed random variables is a χ^2 distribution with q degrees of freedom. If we denote the cumulative distribution function (CDF) of a χ^2 random variable with q degress of freedom by $F_{\chi^2}(\chi^2; q)$ we get

$$P_{\mathrm{c,up}} = F_{\chi^2}\left(\left(|\mathbf{det}\, G|/\alpha_q\right)^{2/q}; q\right). \tag{15}$$

This bound is a very cheap one computationally since it only requires the determinant of G followed by a χ^2 CDF table lookup.

3.3.2 Lower bound based on $d_{\min}$: Given the lattice $L = \mathbf{L}(G)$ with $G \in \mathbf{R}^{q\times q}$, the *shortest lattice vector* or the *minimum distance* of the lattice $d_{\min}$ is defined by

$$d_{\min} = \min_{z \in \mathbf{Z}^q,\ z \neq 0} \|Gz\|. \tag{16}$$

$d_{\min}$ is actually the distance between the origin and its closest neighbor in the lattice. A ball of radius $d_{\min}/2$ is the largest ball centered at the origin that lies in V_0.

Clearly, the probability of noise falling in a ball centered at the origin and of radius $d/2$ where $d \leq d_{\min}$ gives us a lower bound on P_{c}. This lower bound is given by

$$P_{\mathrm{c,low}} = F_{\chi^2}\left(\frac{d^2}{4}; q\right) \quad \text{where} \quad d \leq d_{\min}. \tag{17}$$

So far we haven't discussed how to compute $d_{\min}$. Unfortunately, computing $d_{\min}$ is conjectured to be $\mathcal{NP}$-hard, and to date, no polynomial-time algorithm has been found to compute $d_{\min}$. Therefore, methods are preferred that compute bounds on $d_{\min}$ that have very low computational complexity. These bounds can be used in (17) to get very fast bounds on $P_{\rm c}$. There are many ways to bound $d_{\min}$ as given in [8]. One method is the following.

Gram-Schmidt method lower bound on $d_{\min}$. Suppose $G = [g_1\, g_2 \cdots g_q]$ and $(g_1^*, g_2^*, \ldots, g_q^*)$ is the Gram-Schmidt orthogonalization of $(g_1, g_2, \ldots, g_q)$ as defined in (22). Then

$$d_{\min} \geq d = \mathbf{min}(\|g_1^*\|, \|g_2^*\|, \ldots, \|g_q^*\|). \qquad (18)$$

Re-defining G by finding a *reduced* basis for the lattice using the LLL algorithm would result in a tighter bound since the basis vectors (and hence the g_i^*s) will become shorter.

4 Estimation Problem

4.1 Nearest Lattice Vector Problem

The main computational task of the estimation step is the solution to the integer least-squares problems (8) or (12). Problem (12) is known as the *nearest lattice vector problem* and is $\mathcal{NP}$-hard (cf. [1].) However, in practice, there are reasonably efficient ways to solve this problem which is the main topic of this section.

When $\Upsilon \geq 0$ in (8) is diagonal, $z_{\rm ML}$ can be simply found by rounding the components of $\hat{z}$ to the nearest integer since the integer variables are seperable as

$$(z-\hat{z})^T \Upsilon^{-1} (z-\hat{z}) = \sum_{i=1}^{q} (z_i - \hat{z}_i)^2 / \Upsilon_{ii},$$

where Υ_{ii} is the ith diagonal entry of Υ. A diagonal Υ corresponds to a G with orthogonal columns. Intuitively, this observation suggests that if Υ is almost diagonal, or equivalently, the columns of G are almost orthogonal, rounding off $\hat{z}$ in (8) would give a close approximation to $z_{\rm ML}$. However, G is not always initially given as almost orthogonal even if the lattice $L = \mathbf{L}(G)$ is orthogonal.

As noted before, the LLL algorithm is a useful tool for finding an almost orthogonal basis for a given lattice $L = \mathbf{L}(G)$. Rounding can then be applied to $\hat{z}$ to get a hopefully good approximation for $z_{\rm ML}$. This approximation can serve as a good initial guess for $z_{\rm ML}$ in any algorithm for solving the global optimization problems (8) or (12). A similar idea can be found in [6] and [7].

4.2 Suboptimal Polynomial-Time Algorithms

In this subsection, we address suboptimal polynomial-time algorithms for calculating the minimum of $\|\tilde{y} - Gz\|$ over the integer lattice. These suboptimal algorithms are important for a few reasons. First, they can be performed efficiently with a guaranteed low worst case complexity. Secondly, suboptimal algorithms provide a relatively good initial guess or relatively tight upper bound for any global optimization algorithm. Finally, these suboptimal algorithms might find the global optimum as they often do in practice. If any lower bound d on $d_{\min}$ ($d \leq d_{\min}$) is known, a sufficient condition for the suboptimal minimizer $z_{\rm sub}$ to be the global minimizer $z_{\rm ML}$ is simply given by

$$\|\tilde{y} - Gz_{\rm sub}\| \leq \frac{d}{2} \Longrightarrow z_{\rm sub} = z_{\rm ML}, \qquad (19)$$

as there is only one lattice point in a ball centered at $Gz_{\rm sub}$ and with radius $d_{\min}/2$. Note that d is usually a byproduct of the verification step, and therefore, condition (19) can be checked without any additional computation for finding d.

One suboptimal polynomial-time algorithm is the following (cf. [4].)

Suboptimal algorithm for finding an approximately nearest lattice point based on rounding. Suppose $G \in \mathbf{R}^{q\times q}$ and $\tilde{y} \in \mathbf{R}^q$ are given. A suboptimal solution $z_{\rm sub} \in \mathbf{Z}^q$ in the sense that

$$\|\tilde{y} - Gz_{\rm sub}\| \leq \left(1 + 2q\,(4.5)^{q/2}\right) \min_{z\,\in\,\mathbf{Z}^q} \|\tilde{y} - Gz\|, \qquad (20)$$

exists and is given by[2] $z_{\rm sub} = F\lceil \bar{G}^{-1}\tilde{y}\rfloor$ where $\bar{G}$ is the lattice generator matrix after performing the LLL algorithm on G and $\bar{G} = GF$.

Note that although the worst case bound in (20) appears to be loose, this suboptimal algorithm works *much* better in practice as reported in the literature (cf. [5].)

Another suboptimal polynomial-time algorithm for finding an *approximately nearest* lattice point is due to Babai (1986) (cf. [1], [4] and [8].)

4.3 Searching for Integral Points Inside an Ellipsoid

Once a candidate (or guess) $z = \bar{z}$ to the minimizer of $\|\tilde{y} - Gz\|$ is found, we need to check whether there exists anyother $z \in \mathbf{Z}^q$ satisfying $\|\tilde{y} - Gz\| < \|\tilde{y} - G\bar{z}\|$. If no such z exists then $\bar{z}$ is the global minimizer, otherwise, we can find a better candidate for the minimum of $\|\tilde{y} - Gz\|$ that we need to check for its global minimality as well and so on. Therefore, an important step in finding

[2] $\lceil\cdot\rfloor$ is the componentwise rounding operation to the nearest integer.

the minimum of $\|\tilde{y}-Gz\|$ is searching for integral points (points with integer coordinates z) inside an ellipsoid $\|\tilde{y}-Gz\| < r$. Refer to [8] for a method to perform this *exhaustive search*. It is shown that by putting G into a lower triangular form using a unitary transformation, this search can be performed more efficiently.

4.4 Global Optimization Algorithm

Basically, the global optimization algorithm for finding the minimizer of $\|\tilde{y} - Gz\|$ consists of a *good* initial guess (using suboptimal algorithms of §4.2) followed by an *efficient* exhaustive search (§4.3.) Refer to [8] for details.

4.5 Summary

In this section we sketched a method for solving the *integer least-squares problem* for resolving the integer parameters in the linear model. In practice, this method is very efficient for problems with a few ten integer variables. The success of this method mainly relies on a guaranteed reasonably good initial guess for the minimizer by using the LLL algorithm.

After z is estimated by the global optimization algorithm in §4.4, the maximum likelihood estimate of x, the real parameter, is straightforward. x_{ML} can be found as in (9).

5 Extensions

In this section we point out extensions to model (1). If there is *a priori* knowledge on the distribution of x, similar formulas as in §2 can be given for the (*maximum a posteriori*) estimates (cf. [8].) Now suppose that we assume a state-space structure for the real parameter $x^T = [x_0^T \ \cdots x_N^T]$ as

$$\begin{cases} \xi_{k+1} &= F_k\xi_k + G_k u_k, \ \ \xi(0) = \xi_0, \\ x_k &= H_k\xi_k + v_k \ \text{ for } \ k = 0, 1, 2, \ldots, N \end{cases}$$

and the observation y_k is assumed to depend on the unknowns $x_k \in \mathbf{R}^p$ and $z \in \mathbf{R}^q$ through the linear relationship

$$y_k = A_k x_k + J_k z + w_k$$

in which ξ_0, u_k, v_k and w_k are Gaussian variables. In this case, the matrices G and $\tilde{y}$ in (12) can be updated recursively such that

$$z_{\text{ML}}^{(k)} = \underset{z \in \mathbf{Z}^q}{\textbf{argmin}} \ \|\tilde{y}^{(k)} - G^{(k)}z\|, \quad k = 1, \ldots, N. \ (21)$$

This recursive method (similar to the Kalman filter) has many numerical advantages and reduces data storage for applications in which the data comes in sequentially (*e.g.*, GPS surveying.) A standard *two-pass smoothing algorithm* (cf. [3]) can be used to find the estimates of x_i for $i = 1, \ldots, k$ once $z_{\text{ML}}^{(k)}$ is computed from (21). Refer to [8] for details.

6 Simulations

The simulations in this section are based on a setup similar to the global positioning system (GPS) but in two dimensions (Figure 1.) In general, navigation using phase measurements from sinusoidal signals transmitted from far away locations in space with known coordinates can always be cast as an integer parameter estimation problem in a linear model. The unknown integers enter the equations as the number of carrier signal cycles between the receiver and the satellites when the carrier signal is initially phase locked.

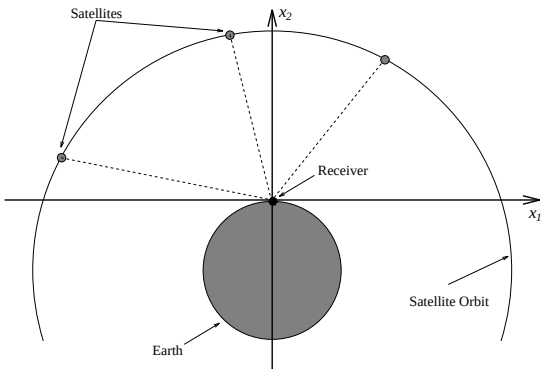

Figure 1: Simulations in this section are based on a simplified GPS setup in two dimensions. The radius of the the shaded circle (earth) is 6357km. Three satellites are assumed that orbit the earth at an altitude of 20200km and period of 12 hours.

In our simulation, we have assumed a *constant* receiver located at $x^T = [-50 \ 100]$ (of course this is not known *a priori*.) However, x is known to be Gaussian with mean zero and variance $\sigma_x = 100$m in both the x_1 and x_2 directions. The wavelength of the carrier signal and angular velocity for all three satellites is $\lambda = 0.19$m and $\omega = 1/120^o\text{sec}^{-1}$ respectively. The satellites make angles of 90^o, 120^o and 45^o with the x_1 axis initially and the direction of rotation for all of them is clockwise. The variance of phase measurement noise in units of length is taken as $\sigma = 0.01$m. The receiver measures the (sinusoidal) carrier signal phase from each of these satellites every $T = 2$ seconds for a period of 200 seconds. Refer to [2] and [8] for details.

Figure 2 gives the exact P_{c} (computed using Monte Carlo with 1500 random variates), and the upper and lower bounds $P_{\text{c,up}}$ and $P_{\text{c,low}}$ (lower bound using Gram-Schmidt method lower bound on $d_{\min}$) as a func-

tion of time. Clearly, the upper bound $P_{\mathrm{c,up}}$ appears to be a very good approximation to P_{c} and as $P_{\mathrm{c}} \to 1$, the lower bound $P_{\mathrm{c,low}}$ becomes exact as well. This is a very good feature as we are not conservative in bounding P_{c} when P_{c} is high and there is high reliability in our estimates.

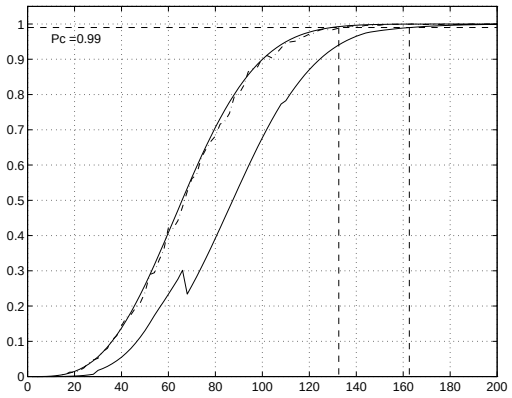

Figure 2: P_{c} (*dash-dot curve*), $P_{\mathrm{c,up}}$ (upper bound using $|\det G|$; *upper solid curve*), $P_{\mathrm{c,low}}$ (lower bound using lower bound on $d_{\min}$ using Gram-Schmidt method; *lower solid curve*) as a function of time.

In practice we might never compute the exact P_{c} and we have to live with (the inexact but much more easily computable) bounds. Given these bounds as in Figure 2, we can conclude that P_{c} is not greater than 0.99 after 132 seconds, so with a confidence level of 99%, our reliability on z_{ML} is low. We have to wait for another 30 seconds until the lower bound hits the $P_{\mathrm{c}} = 0.99$ line. From hereon, it is guaranteed that $P_{\mathrm{c}} > 0.99$ and therefore z_{ML} is reliable.

We have plot the number of inner iterations in the global optimization algorithm to compute z_{ML} vs. time in Figure 3. A number of iterations equal to *zero* means that the initial value for z found in the algorithm (by performing the LLL algorithm and rounding off) is guaranteed to be the global minimizer according to (19) where d is the lower bound found on $d_{\min}$ using the Gram-Schmidt method in §3.3.2. Note the low number of iterations (cf. [8] for details.) In fact, the global optimization algorithm is very efficient in practice and can be solved instantly by a computer.

It is very interesting to note that for high P_{c} ($P_{\mathrm{c}} > 0.9$ or after 95 seconds), the only case for which we are really interested in z_{ML} because of high reliability, the number of inner iterations is zero and therefore the global optimization algorithm is extremely efficient. In general this is true, and the efficiency of the global optimization algorithm is greatly enhanced as $P_{\mathrm{c}} \to 1$.

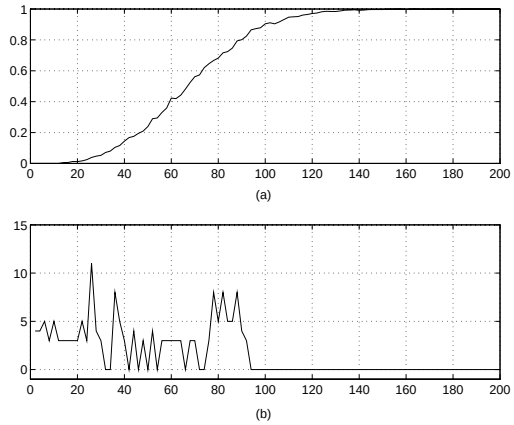

Figure 3: (a) P_{c} as a function of time. (b) Number of inner iterations in the global optimization algorithm for finding z_{ML} as a function of time. For high P_{c}, the *initial guess* for the global minimizer is *guaranteed* to be the *global minimizer* using a *very cheaply computed* lower bound on $d_{\min}$ (using the Gram-Schmidt method.) Therefore, the global optimization algorithm becomes extremely efficient for high P_{c}.

Figure 4 shows P_{c} as a function of time and the times at which the integer parameter z is resolved correctly using global optimization and the times at which it is resolved correctly using rounding off (*i.e.*, $z_{\mathrm{est}} = \lceil G^{-1}\tilde{y} \rfloor$; without using the LLL algorithm of course.) The results clearly show that simple rounding shouldn't be used in practice since even after 200 seconds we are still not able to resolve z by this method. On the other hand, using global optimization, z is resolved after 90 seconds or when $P_{\mathrm{c}} > 0.8$.

7 Conclusions

In this paper we considered parameter estimation in linear models when some of the parameters are known to be integers. Simulation results show that if we neglect the integer nature of the parameters (treat these parameters as being real and then round off), we get *very inexact* estimates in practice. The main computational effort in the *estimation step* turns out to be the solution to an integer least-squares problem which is in fact $\mathcal{NP}$-hard. *Verifying* the maximum likelihood estimates seems to be a problem as hard as the *estimation step* (conjectured to be $\mathcal{NP}$-hard.) The probability of correct integer parameter estimation P_{c} was chosen as a useful reliability measure for our estimates. A *polynomial-time algorithm* due to Lenstra, Lenstra and Lovász (LLL algorithm) found to be very useful in the estimation and verification problems.

Very easily computable bounds can be found on P_{c} and

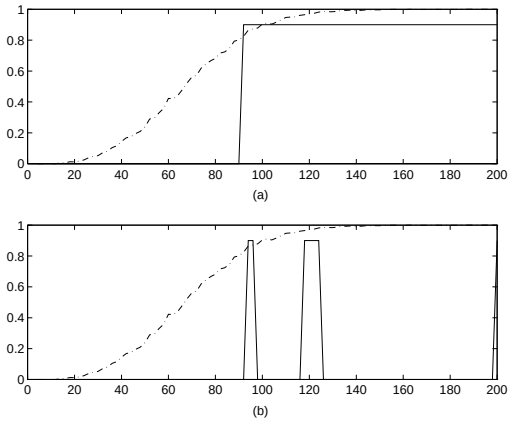

Figure 4: Simulation result when $x^T = [0\ 0]$. (*a*) P_c (*dash-dot curve*) and the times at which z is resolved correctly using global optimization (*solid curve*, a *high* means that z is resolved correctly or $z_{ML} = z$, while a *low* means that it is not) as a function of time. (*b*) P_c and the times at which z is resolved correctly using simple rounding as a function of time.

these bounds become exact as P_c is close to one; the only case that we are really interested in our parameter estimates because of their high reliability. The global optimization algorithm for solving the integer least-squares problem works very well for problem sizes of a few tens of variables which is typical in applications such as GPS. Moreover, the global optimization algorithm becomes even more efficient for high P_c.

These observations suggest the following general outline for integer parameter estimation in linear models.

General outline for integer parameter estimation in linear models. Under the setup of §1:

- Update G after every measurement, say recursively, as noted in §5.
- Compute $P_{c,up}$ in (15) which is the upper bound based on the determinant of G. In practice, this bound is very close to P_c as demonstarted in the simulations.
- When $P_{c,up}$ is large enough (say $P_{c,up} > 0.99$), compute the lower bound $P_{c,low}$ in which the Gram-Schmidt method is used to provide a lower bound on d_{min} (apply the LLL algorithm on G to get a tighter bound.)
- When $P_{c,low}$ is high (say $P_{c,low} > 0.99$), we can assume that $z = z_{ML}$. Therefore, only at this step eventually solve the global optimization problem (12). Since P_c is close to unity, the algorithm for solving (12) would be extremely efficient.
- Once $z = z_{ML}$ has been resolved, there is no integer parameter to worry about. Use standard methods to estimate the *real* parameter x (*e.g.*, *two-pass smoothing algorithm.*)

References

[1] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization.* Springer-Verlag 1988.

[2] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins. *Global Positioning System, Theory and Practice.* Springer-Verlag 1994.

[3] B. D. O. Anderson, J. B. Moore. *Optimal Filtering.* Englewood Cliffs, NJ, Prentice Hall 1979.

[4] L. Babai, "On Lovász' lattice reduction and the nearest lattice point problem," *Combinatorica 6* (1986) 1-13.

[5] A. M. Odlyzko, H. te Riele (1985), "Disproof of the Mertens conjecture," *J. reine angew. Math.* 357 (1985), 138-160.

[6] P.J.G. Teunissen, "The invertible GPS ambiguity transformations," *Manuscripta Geodaetica* (1995) 20:489-497.

[7] P.J.G. Teunissen, "The least-squares ambiguity decorrelation adjustment: a method for fast GPS integer ambiguity estimation," *J. of Geodesy* (1995) 70:65-82.

[8] A. Hassibi and S. Boyd, "Integer parameter estimation in linear models with GPS applications," Technical Report, Information Systems Laboratory, Stanford University, February 1995.

Appendix: Lenstra, Lenstra, Lovász (LLL) Algorithm

Suppose a lattice $L = \mathbf{L}(G)$ with $G = [g_1\ g_2 \cdots g_q]$ is given. A *reduced* basis for L can be obtained as follows,

- Perform the *Gram-Schmidt orthogonalization* procedure on the vectors $g_1, g_2, \ldots, g_q$, *i.e.*, compute the vectors $g_1^*, g_2^*, \ldots, g_q^*$ through the recursion

$$g_j^* = g_j - \sum_{i=1}^{j-1} \frac{g_j^T g_i^*}{\|g_j^*\|^2} g_i^* \quad \text{for } j = 1, 2, \ldots, q. \tag{22}$$

  $\{g_1^*, g_2^*, \ldots, g_q^*\}$ will be an orthogonal basis for the subspace spanned by $g_1, g_2, \ldots, g_q$. From the definition (22), it is trivial that any vector g_j can be expressed as a linear combination of the vectors $g_1^*, g_2^*, \ldots, g_q^*$ as

$$g_j = \sum_{i=1}^{j} \mu_{ji} g_i^*, \tag{23}$$

  where $\mu_{ji} = g_j^T g_i^* / \|g_j^*\|^2$ for $i = 1, 2, \ldots, j-1$ and with $\mu_{jj} = 1$.
- For $j = 1, 2, \ldots, q$, and, given j, for $i = 1, 2, \ldots, j-1$, replace g_j by $g_j - \lceil \mu_{ji} \rfloor g_i$, where $\lceil \mu_{ji} \rfloor$ is the integer nearest to μ_{ji}. Update the g_j^*'s and μ_{ji}'s.
- If there is a subscript j violating

$$\|g_{j+1}^* + \mu_{(j+1)j} g_j^*\|^2 \geq \frac{3}{4} \|g_j^*\|^2, \tag{24}$$

  then interchange g_j and g_{j+1} and return to the first step, otherwise, stop. $G = [g_1\ g_2 \cdots g_q]$ is the *reduced* generator matrix of the lattice L.

The choice of 3/4 in (24) as the allowed factor of decrease is arbitrary: any number between 1/4 and 1 would do just as well. The most natural choice (giving a best upper bound on the product $\|g_1\| \|g_2\| \cdots \|g_q\|$) would be 1 instead of 3/4; but the polynomiality of the resulting algorithm cannot be guaranteed.