LGR-Series Publications of the Delft Geodetic Computing Centre No. 12

The LAMBDA method for integer ambiguity estimation: implementation aspects

Paul de Jonge and Christian Tiberius

August 1996

TU Delft
Delft University of Technology

The LAMBDA method for integer ambiguity estimation: implementation aspects

Paul de Jonge and Christian Tiberius

August 1996

Delft Geodetic Computing Centre

Delft University of Technology
Faculty of Geodetic Engineering
Thijsseweg 11
2629 JA Delft
The Netherlands

Office:	Telephone:	(31) 15-2783546
	Telefax:	(31) 15-2783711
	Telex:	butud 38151
	E-mail:	lgr@geo.tudelft.nl

lgr12.ps:	Postscript version
lgr12.zip:	idem, zipped
lgr12.ps.Z:	idem, Unix compressed

ISSN: 93-0928-2122
Press: Universiteitsdrukkerij TU Delft

Contents

Preface

The development of the LAMBDA method for ambiguity resolution started with its introduction at the IAG General Meeting in Beijing in 1993 [1]. The implementation has been realized by P.J. de Jonge and C.C.J.M. Tiberius under supervision of professor P.J.G. Teunissen.

The underlying report provides a low level description of the method in its current implementation. Mrs. D. Morujão of the University of Coimbra, Portugal, and Mr. D. Charrier of SERCEL, France, are acknowledged for pointing out errors in the draft and suggesting several improvements. The second author received financial support of the Cornelis Lely foundation of Rijkswaterstaat.

Abstract

High precision relative GPS positioning is based on the very precise carrier phase measurements. In order to achieve high precision results within a short observation time span, the integer nature of the ambiguities has to be exploited. In this report the full procedure for parameter estimation based on the model of double difference GPS observations is reviewed, but the emphasis will be on the *integer estimation* of the GPS double difference ambiguities.

The LAMBDA method will be used for the integer estimation. LAMBDA stands for Least-squares AMBiguity Decorrelation Adjustment. By means of the Z-transformation, the ambiguities are decorrelated prior to the integer estimation. The integer minimization problem is then attacked by a discrete search over an ellipsoidal region, the ambiguity search ellipsoid.

The shape and orientation of the ellipsoid are governed by the variance covariance matrix of the ambiguities. The decorrelation realizes an ellipsoid that is very much sphere-like. It can be searched through very efficiently. The size of the ellipsoid can be controlled prior to the search using the volume function. The volume gives an indication of the number of candidates contained in the ellipsoid. A request for only a few candidates can be made, and this enables a straightforward implementation of the search. A limited number of candidates will be output of which one is the integer least-squares estimate for the vector of ambiguities.

The LAMBDA method provides, based on the float ambiguities and their variance covariance matrix, the integer least-squares estimate for the ambiguities. Therewith, the fixed solution can be computed. By the decorrelation, the integer estimation can be carried out very fast and efficiently. The total procedure typically takes 30 ms or less on a 486-66 MHz PC for a baseline with 12 ambiguities.

The method has been introduced in [1]. Preliminary fast positioning results using the LAMBDA method are given in [8], [9] and [10]. In this report, the implementation aspects of the method are discussed. A detailed description of the method is given, as well as the algorithms in the stylized Matlab notation of [7].

1 Introduction

High precision relative GPS positioning is based on the very precise carrier phase measurements. In order to achieve high precision results within a short observation time span, the integer nature of the ambiguities has to be exploited. In this report we will review the full procedure for parameter estimation based on the model of double difference GPS observations. The integer estimation therein is treated extensively. Note however, that integer estimation is not restricted to the double difference model; it can be applied as well in the models of un-differenced or single differenced observations with the rank-defect properly taken into account.

We will use the LAMBDA method for the integer estimation. LAMBDA stands for Least-squares AMBiguity Decorrelation Adjustment. After applying a decorrelating transformation, a sequential conditional adjustment is made upon the ambiguities. As a result, integer least-squares estimates for the ambiguities are obtained. The method has been introduced in [1]. Preliminary fast positioning results using the LAMBDA method are given in [8], [9] and [10].

The central chapters of this report are 2, 3 and 4. Chapter 2 describes the full procedure for parameter estimation. We start with the model of double difference observations, and first the float solution is computed. Based on the real valued estimate for the double difference ambiguities and the corresponding variance covariance matrix, the integer estimation is carried out. The procedure concludes with the fixed solution. Chapter 5 provides a flow diagram.

The integer minimization, which amounts to a discrete search over an ellipsoidal region, the ambiguity search ellipsoid, is treated in chapter 4. The volume of this ellipsoid, which can be computed prior to the search, plays an important role in controlling the size of the ellipsoid.

The Z-transformation is treated in chapter 3. By a systematic pair-wise decorrelation of the ambiguities, the – usually extremely elongated – ambiguity search ellipsoid is transformed towards a sphere-like shape. The Z-transformation is applied prior to the actual integer estimation. The discrete search can then be carried out very efficiently. The Z-transformation, chapter 3, and the search, chapter 4, are illustrated by means of a numerical example with three ambiguities.

2
Parameter estimation

2.1 Observation equations

In the discussion below we will set up the model of observation equations. We use a simple mathematical model appropriate for short baselines. Two types of unknowns occur: baseline coordinates and double difference ambiguities. We assume to have available only carrier phase observations. Code observations can be included as well; they are not related to the ambiguity terms.

The linearized double difference observation equations are collected in the following linear system of equations, see also section 2.1 in [2]:

$$y = Bb + Aa + e \tag{2.1}$$

where:

y the vector of observed minus computed double difference carrier phase measurements; $y \in R^m$

b the vector that contains the increments of the p baseline coordinates

a the vector of n double difference ambiguities

B the $m \times p$ design matrix for the baseline coordinates

A the $m \times n$ design matrix for the ambiguity terms

e the vector of unmodelled effects and measurement noise

The $m \times (p+n)$ design matrix $[\ B \quad A\ ]$ is assumed to have full rank equal $(p+n)$, i.e. enough observations have been made to determine the baseline coordinates and the double difference ambiguities. The variance-covariance matrix of the observables $\underline{y}$ is Q_y, the matrix is symmetric and positive definite.

Other unknowns, like e.g. atmospheric delay parameters can be included as well; they are then added to the vector b. In other words, all double difference ambiguities are contained in a, all other parameters in b. The double difference ambiguities are placed at the end for reasons of computational efficiency.

2.2 Estimation criterion

The least-squares principle will be used to compute estimates for the baseline coordinates and the *integer* double difference ambiguities:

$$\min_{b,a} \|y - Bb - Aa\|^2_{Q_y^{-1}} \text{ with } b \in R^p \text{ and } a \in Z^n \tag{2.2}$$

The parameter estimation is carried out in three steps, see section 2.2 in [2]. The steps are the float solution, the integer ambiguity estimation and the fixed solution. Although model validation is important as well, we will concentrate in this report on the estimation of the unknown parameters. In particular we will treat the integer ambiguity estimation.

2.3 Float solution

The minimization (2.2) is carried out with $b \in R^p, a \in R^n$. Real valued estimates for the double difference ambiguities will be obtained.

Usually the full rank least-squares problem is solved via normal equations, see section 5.3 of [7]. The system of normal equations $Nx = r$, with N the normal matrix and r the right hand side, for (2.1) reads

$$\underbrace{\begin{bmatrix} B^*Q_y^{-1}B & B^*Q_y^{-1}A \\ A^*Q_y^{-1}B & A^*Q_y^{-1}A \end{bmatrix}}_{N} \underbrace{\begin{bmatrix} b \\ a \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} B^*Q_y^{-1}y \\ A^*Q_y^{-1}y \end{bmatrix}}_{r} \tag{2.3}$$

The Cholesky factorization of the normal matrix is made, see theorem 4.2.5 of [7]: $N = CC^*$, with matrix C a full rank $(p+n)$ lower triangular matrix. Algorithm 4.2.2. of [7] can be used for the Cholesky factorization. The system $CC^*x = r$ is then solved by for- and backward substitution. The estimates and the variance-covariance matrix are:

$$\begin{bmatrix} \hat{b} \\ \hat{a} \end{bmatrix} \begin{bmatrix} Q_{\hat{b}} & Q_{\hat{b}\hat{a}} \\ Q_{\hat{a}\hat{b}} & Q_{\hat{a}} \end{bmatrix} \tag{2.4}$$

In the integer ambiguity estimation we will use the Cholesky factor G of the inverse of the variance-covariance matrix $Q_{\hat{a}}$. This is why the unknowns have been put in the order: first baseline coordinates and then double difference ambiguities. The Cholesky factor C is partitioned as

$$C = \begin{bmatrix} * & 0 \\ * & G \end{bmatrix} \tag{2.5}$$

with G a lower triangular matrix of dimension n, for which holds that $Q_{\hat{a}}^{-1} = GG^*$. The Cholesky factor G becomes available at no extra cost in the float solution.

2.4 Integer ambiguity estimation

The second step consists of

$$\min_a \|\hat{a} - a\|^2_{Q_{\hat{a}}^{-1}} \text{ with } a \in Z^n \tag{2.6}$$

This minimization yields the integer least-squares estimate for the vector of ambiguities: $\check{a}$. The computation of the integer estimate will be treated in detail in chapters 3 and 4.

2.5 Fixed solution

The final solution, with the ambiguities fixed to their integer least-squares estimates $\check{a}$, reads

$$\check{b} = \hat{b} - Q_{\hat{b}\hat{a}} Q_{\hat{a}}^{-1} (\hat{a} - \check{a}) \tag{2.7}$$

The least-squares estimates $\check{b}$ and $\check{a}$ are the solution to the constrained minimization (2.2). Note that in practice $\check{b}$ is not computed with (2.7). The final estimates for the baseline coordinates follow from the first p equations of system (2.3) in which the vector of integer estimates $\check{a}$ has been substituted for a. Note that the Cholesky factor of matrix $B^* Q_y^{-1} B$ is already available. Estimate $\check{b}$ is then easily obtained via for- and backward substitution. The variance-covariance matrix $Q_{\hat{b}|\check{a}}$, see section 2.2 of [2], can be computed via inversion of the Cholesky factor.

3 Integer ambiguity estimation: transformation

3.1 Introduction

For the computation of the integer least-squares estimate $\check{a}$, we use the LAMBDA method. It has been introduced in [1] and reviewed in [5]. The two main features of the LAMBDA method are

1. the decorrelation of the ambiguities, realized by a reparametrization, or as we call it, a Z-transformation, of the original ambiguities a to new ambiguities $z = Z^*a$, and
2. the actual integer ambiguity estimation

Through the Z-transformation the variance-covariance matrix is transformed accordingly:

$$Q_{\hat{z}} = Z^* Q_{\hat{a}} Z \tag{3.1}$$

The efficiency of the method comes from the decorrelation step and has been explained in detail by analysis of the precision and correlation of the GPS double difference ambiguities in [4]-[6].

The actual integer minimization is then made upon the transformed ambiguities. In practice the minimization (2.6) amounts to a search over grid points inside the n-dimensional ambiguity hyper-ellipsoid, defined by the variance-covariance matrix of the ambiguities

$$(\hat{z} - z)^* Q_{\hat{z}}^{-1} (\hat{z} - z) \leq \chi^2 \tag{3.2}$$

The search must yield the grid point that is nearest to the real valued estimate, with nearness measured in the metric of the variance-covariance matrix, see section 3 of [1]. Note that an n-dimensional ellipsoid, is actually an $(n-1)$ dimensional surface in the R^n.

In this and the following chapter, the algorithmic details of the LAMBDA method will be discussed. The input basically consists of $\hat{a}$ and $Q_{\hat{a}}$, the estimate for the vector of ambiguities and the variance-covariance matrix from the float solution.

3.2 The decorrelating or Z-transformation, introduction

To explain the concept of the decorrelating transformation we decompose the variance-covariance matrix $Q_{\hat{a}}$ into

$$Q_{\hat{a}} = L^{-*} D^{-1} L^{-1} \tag{3.3}$$

Note that this corresponds to the inverse of the LDL^*-decomposition of $Q_{\hat{a}}^{-1}$ which is easily derived from the already computed Cholesky factor (see equation (2.5)). The principle of the decorrelation is to find a matrix Z, which is an integer approximation of matrix L. If we would be able to find an integer matrix Z that fulfills the requirements in [3] and that exactly equals L, then with (3.1)

$$Q_{\hat{z}} = Z^* Q_{\hat{a}} Z = Z^* L^{-*} D^{-1} L^{-1} Z = D^{-1} \tag{3.4}$$

The transformed ambiguities $\hat{z}$ are fully decorrelated and the integer minimization reduces to a simple rounding of the real valued estimates, see section 2.3 of [2]. In practice a complete decorrelation will not be possible due to the integer requirement.

The result of the decorrelation process is the square $n \times n$ transformation matrix Z; for its properties we refer to [3]. The estimate $\hat{z}$ follows from $\hat{z} = Z^* \hat{a}$. The factors of the variance-covariance matrix are updated in the decorrelation process: $\tilde{D}^{-1}$ and $\tilde{L}^{-1}$. They satisfy

$$Q_{\hat{z}} = \tilde{L}^{-*} \tilde{D}^{-1} \tilde{L}^{-1} \tag{3.5}$$

The problem (2.6) has now been transformed into the following minimization

$$\min_{z} \|\hat{z} - z\|^2_{Q_{\hat{z}}^{-1}} \text{ with } z \in Z^n \tag{3.6}$$

Inversion of $\tilde{L}^{-1}$ yields $\tilde{L}$ and inversion of $\tilde{D}^{-1}$ yields $\tilde{D}$. They satisfy

$$Q_{\hat{z}}^{-1} = \tilde{L} \tilde{D} \tilde{L}^* \tag{3.7}$$

This decomposition in $\tilde{L}$ and $\tilde{D}$ is then used in the search, see chapter 4.

3.3 The decomposition of the variance-covariance matrix

The ambiguity decorrelating transformation can be computed in several ways depending on the matrix one starts with, and on the kind of factorization one uses. One can either use the variance-covariance matrix of the ambiguities or its inverse. The factorization can be either an LDL^*, or an L^*DL factorization. Schematic we have

$$\begin{array}{lcll} Q_{\hat{a}}^{-1} = L_1 D_1 L_1^* & ; & Q_{\hat{a}} = L_1^{-*} D_1^{-1} L_1^{-1} & \text{cf. [1]} \\ Q_{\hat{a}}^{-1} = L_2^* D_2 L_2 & ; & Q_{\hat{a}} = L_2^{-1} D_2^{-1} L_2^{-*} & \\ Q_{\hat{a}} = L_3^* D_3 L_3 & ; & Q_{\hat{a}}^{-1} = L_3^{-1} D_3^{-1} L_3^{-*} & \\ Q_{\hat{a}} = L_4 D_4 L_4^* & ; & Q_{\hat{a}}^{-1} = L_4^{-*} D_4^{-1} L_4^{-1} & \text{cf. [2]} \end{array} \tag{3.8}$$

where:

L_i is a unit lower triangular matrix (i.e. with ones on the diagonal) and

D_i is a diagonal matrix, $D_i = diag(d_1, \ldots, d_n)$

Note that $L_1 = L_3^{-1}, L_2 = L_4^{-1}$ and that $D_1 = D_3^{-1}, D_2 = D_4^{-1}$. In [1] the LDL^* factorization of the inverse of the variance-covariance matrix is used, and in [2] the LDL^* factorization of the variance-covariance matrix itself.

The implementation of the LAMBDA method described in this report is based on the description in [1]. The reason for it is that the Cholesky factor G is already available in the float solution (see equation (2.5)). The LDL^*-decomposition is easily constructed from the Cholesky factor. It holds that

$$G = L\sqrt{D} \tag{3.9}$$

Note that matrix $Q_{\hat{a}}^{-1}$ is symmetric positive definite: for the diagonal elements of D it holds that $d_i > 0 \; \forall \; i = 1, \ldots, n$.

The corresponding factorization of the variance-covariance matrix reads

$$Q_{\hat{a}} = L^{-*}D^{-1}L^{-1} \tag{3.10}$$

An interpretation can be given to the elements of matrix D^{-1}, see section 5 of [1]: they are the conditional variances of the double difference ambiguities.

$$d_i^{-1} = \sigma^2_{\hat{a}_{i|i+1,\ldots,n}} \tag{3.11}$$

The inversion of the factor can be done in place, an algorithm for it can be found in chapter 6 of [11], and in appendix A of this report. The procedure proposed here is computationally more efficient than first computing the variance-covariance matrix, and then factoring it.

Instead of inverting the LDL^*-decomposition, one can compute $Q_{\hat{a}} = L^{-*}D^{-1}L^{-1}$ directly. This should be done when the factor L is not already available. We will give two algorithms for this factorization.

The first, *FMFAC5*, is an outer product formulation, and is used to explain some features of the construction of the transformation matrix Z in section 3.4.

The second algorithm, *FMFAC6*, is a 'bordering method' formulation (see [17]), and has as additional feature a check for positive-definiteness of the variance-covariance matrix Q. An indication for the numerical condition of matrix Q is given by the smallest collinearity number c_{min}, [ibid]. This number ranges from one to zero; if the number is less than a pre-defined constant ϵ (for the cases of 1 second time span we take $\epsilon = 10^{-9}$), the matrix is considered to be singular.

Algorithm *FMFAC5*: Given the symmetric positive definite matrix Q, an L^*DL factorization is computed by the outer product method. Matrix L may over write matrix Q; the latter is destroyed during the computation. Only the lower triangle of Q is accessed.

```
for $i = n : -1 : 1$
    $D(i,i) = Q(i,i)$
    $L(i,1:i) = Q(i,1:i)/\sqrt{Q(i,i)}$
    for $j = 1 : i-1$
        $Q(j,1:j) = Q(j,1:j) - L(i,1:j)L(i,j)$
    end
    $L(i,1:i) = L(i,1:i)/L(i,i)$
end
```

Algorithm *FMFAC6*: Given the symmetric positive definite matrix Q, an L^*DL factorization is computed via a backward Cholesky factorization in a bordering method formulation. Also the smallest collinearity number c_{min} is computed. Matrix L over writes matrix Q. Only the lower triangle of Q is accessed. The function *dot* computes the dot or inner product of two vectors.

```
$L = Q$
$c_{min} = 1$
for $j = n : -1 : 1$
    for $i = n : -1 : j+1$
        $L(i,j) = (L(i,j) - dot(L(i+1:n,j), L(i+1:n,i)))/L(i,i)$
    end
    $t = L(j,j) - dot(L(j+1:n,j), L(j+1:n,j))$
    $c = t/L(j,j)$
    if $c < c_{min}$
        $c_{min} = c$
    end
    $L(j,j) = \sqrt{t}$
end
for $i = 1 : n$
    $L(i,1:i-1) = L(i,1:i-1)/L(i,i)$
    $D(i) = L(i,i)L(i,i)$
    $L(i,i) = 1$
end
```

3.4 Modifying the decomposition

Note: The construction of the transformation matrix Z is based on the variance-covariance matrix $Q_{\hat{a}}$. To simplify notation we will assume for the remainder of this chapter 3 that we have the L^*DL factorization of $Q_{\hat{a}}$ (the third option of (3.8) instead of the first), thus $d_i = \sigma^2_{\hat{a}_{i|i+1,\ldots,n}}$.

The construction of the $n \times n$ Z matrix in (3.1) consists of a sequence of integer approximated Gauss transformations and permutations. Both are admissible ambiguity transformations (see e.g. [1] and [3]). The actual decorrelation is carried out by the integer Gauss transformation. If necessary the ambiguities are reordered, by a permutation, in order to allow for further decorrelation.

Before we continue, we will discuss the triangular decomposition after transformation, analogously to the decomposition discussed in [2] (formulae 47-49 at page 79).

The unit lower triangular matrix L and the diagonal matrix D are partitioned into

$$L = \begin{bmatrix} L_{11} & & \\ L_{21} & L_{22} & \\ L_{31} & L_{32} & L_{33} \end{bmatrix} \text{ and } D = \begin{bmatrix} D_{11} & & \\ & D_{22} & \\ & & D_{33} \end{bmatrix} \tag{3.12}$$

Sub-matrix L_{22} is of order 2, L_{11} of order $i-1$, and L_{33} of order $n-i-1$. The dimension of the other sub-matrices in L and D are determined accordingly. If we apply a two-dimensional ambiguity transformation on the i-th and the $(i+1)$-th ambiguity, with the block-diagonal matrix

$$Z = \begin{bmatrix} I_{i-1} & & \\ & Z_{22} & \\ & & I_{n-i-1} \end{bmatrix} \tag{3.13}$$

with Z_{22} a square and full rank 2×2 matrix, we get the new triangular decomposition $L'^* D' L'$, with

$$L' = \begin{bmatrix} L_{11} & & \\ \bar{L}_{21} & \bar{L}_{22} & \\ L_{31} & \bar{L}_{32} & L_{33} \end{bmatrix} \text{ and } D' = \begin{bmatrix} D_{11} & & \\ & \bar{D}_{22} & \\ & & D_{33} \end{bmatrix} \tag{3.14}$$

The fact that only L_{21}, L_{22}, L_{32} and D_{22} do change, can be explained by looking at the outer product form of the L^*DL factorization as given in section 3.3. We have (the variance-covariance matrix is symmetric):

$$Z^* Q_{\hat{a}} Z = \begin{bmatrix} Q_{11} & & \\ Z_{22}^* Q_{21} & Z_{22}^* Q_{22} Z_{22} & \\ Q_{31} & Q_{32} Z_{22} & Q_{33} \end{bmatrix} = \begin{bmatrix} Q_{11} & & \\ \bar{Q}_{21} & \bar{Q}_{22} & \\ Q_{31} & \bar{Q}_{32} & Q_{33} \end{bmatrix} \tag{3.15}$$

Let us look at the updating step of the algorithm i.e. the part where an outer product is subtracted from the matrix. The parts of the outer product that are affected by Z always coincide with the parts of Q affected by Z. So the 'reduced' Q has the same structure as in (3.15). Since L is basically obtained from extracting rows from the reduced Q divided by the square root of the corresponding diagonal element, and D is set equal to the diagonal element, the modified decomposition will be as indicated in (3.14).

The modified factor can be related to the original one. We know that

$$L'^* D' L' = Z^* L^* D L Z$$

From this relation we easily derive for $\bar{L}_{32}$

$$L_{33}^{*}D_{33}\bar{L}_{32} = L_{33}^{*}D_{33}L_{32}Z_{22}$$

or,

$$\bar{L}_{32} = L_{32}Z_{22} \tag{3.16}$$

For $\bar{L}_{22}$ holds that

$$\bar{L}_{22}^{*}\bar{D}_{22}\bar{L}_{22} + \bar{L}_{32}^{*}D_{33}\bar{L}_{32} = Z_{22}^{*}(L_{22}^{*}D_{22}L_{22} + L_{32}^{*}D_{33}L_{32})Z_{22}$$

or, using (3.16)

$$\bar{L}_{22}^{*}\bar{D}_{22}\bar{L}_{22} = Z_{22}^{*}(L_{22}^{*}D_{22}L_{22})Z_{22} \tag{3.17}$$

And for $\bar{L}_{21}$ holds that

$$\bar{L}_{22}^{*}D_{22}\bar{L}_{21} + \bar{L}_{32}^{*}D_{33}L_{31} = Z_{22}^{*}(L_{22}^{*}D_{22}L_{21} + L_{32}^{*}D_{33}L_{31})$$

from which can be derived after substituting (3.16)

$$\bar{L}_{22}^{*}\bar{D}_{22}\bar{L}_{21} = Z_{22}^{*}L_{22}^{*}D_{22}L_{21}$$

Using the relation $Z_{22}^{*}L_{22}^{*}D_{22} = \bar{L}_{22}^{*}\bar{D}_{22}\bar{L}_{22}(L_{22}Z_{22})^{-1}$ which follows from equation (3.17) we finally get

$$\bar{L}_{21} = \bar{L}_{22}(L_{22}Z_{22})^{-1}L_{21} \tag{3.18}$$

Conclusion: once matrix Q is factored as $Q = L^{*}DL$, the factorization of $Z^{*}QZ$ with Z defined in (3.13), can be efficiently computed from the existing factorization. In the unit lower triangular matrix L only the rows i and $i+1$ and the columns i and $i+1$ change. The modification of the diagonal matrix D is limited to the elements (i,i) and $(i+1,i+1)$. The transformation of the variance-covariance matrix can be realized by updating the factors L and D.

3.5 The integer Gauss transformation

The relations found in section 3.4 can be further simplified if we restrict the transformation (sub-)matrix to a single Gauss-transformation (see [7] par 3.2.1):

$$Z_{22} = \begin{bmatrix} 1 & 0 \\ \alpha & 1 \end{bmatrix} \tag{3.19}$$

Then

$$\bar{L}_{32} = \begin{bmatrix} l_{i+2,i} + \alpha l_{i+2,i+1} & l_{i+2,i+1} \\ l_{i+3,i} + \alpha l_{i+3,i+1} & l_{i+3,i+1} \\ \vdots & \vdots \\ l_{n,i} + \alpha l_{n,i+1} & l_{n,i+1} \end{bmatrix} \tag{3.20}$$

$$\bar{L}_{22} = \begin{bmatrix} 1 & 0 \\ l_{i+1,i} + \alpha & 1 \end{bmatrix} \text{ and } \bar{D}_{22} = \begin{bmatrix} d_i & 0 \\ 0 & d_{i+1} \end{bmatrix} \tag{3.21}$$

$$\bar{L}_{21} = L_{21} \tag{3.22}$$

We see that only the unit lower triangular matrix L changes. The diagonal matrix D is left unchanged by a single Gauss transformation. In other words: the conditional variances do not change, only the conditional covariances and thus the 'normal' variances change. This can intuitively be understood since the order of ambiguities was not changed.

To achieve full decorrelation we would like to choose $\alpha = -l_{i+1,i}$. In general $l_{i+1,i} \notin Z$. To meet the requirements in [3], α is approximated by $-[l_{i+1,i}]$, where [.] stands for the nearest integer operator (*nint*). With this integer approximation, we can make the absolute value of any non-diagonal element of L less than or equal to .5. The integer Gauss transformation was discussed in [3] and [8].

In general, if we take for Z the unit matrix with an additional α at position (i, j), with $i > j$, the elements that change are

$$l'_{i,j} = l_{i,j} + \alpha \tag{3.23}$$

$$l'_{k,j} = l_{k,j} + \alpha l_{k,i} \text{ for } k = i+1, \ldots, n \tag{3.24}$$

The algorithm for the computation of a Z-transformation matrix, that will make the absolute value of all non-diagonal elements of L less than or equal to .5, is given as

Algorithm *ZTRAN*: Given the unit lower triangular factor L from the L^*DL factorization of the variance-covariance matrix $Q_{\hat{a}}$, a Z-transformation matrix is computed. Z will be lower triangular on output if set to the unit matrix on input. The computation of this matrix is performed *column wise* from *right to left*, i.e. from n to 1. Instead of decorrelating all n columns, one can apply the algorithm to only one column k, by changing the interval for i to $i = k : k$. The vector with estimates $\hat{a}$ is transformed to $\hat{z} = Z^*\hat{a}$. The original factor L is transformed to $L' = LZ$ for which holds: $L'^*DL' = Q_{\hat{z}}$. Factor L remains lower triangular.

```
for $i = n : -1 : 1$
    for $j = i + 1 : n$
        $\mu = nint(L(j, i))$
        if $\mu \neq 0$
            $L(j : n, i) = L(j : n, i) - \mu L(j : n, j)$
            $Z(1 : n, i) = Z(1 : n, i) - \mu Z(1 : n, j)$
            $a(i) = a(i) - \mu a(j)$
        end
    end
end
```

In the inner loop, if $\mu \neq 0$, column j times μ is subtracted from column i, for matrix Z and factor L.

3.6 The reordering of the conditional variances

As was explained in [2] and [6] the spectrum of the conditional variances shows, in the case of a single baseline, a distinctive discontinuity when passing from the third to the fourth ambiguity. The size of this discontinuity is governed by the length of the observation time span. In [4] and [6] it was shown that it is this discontinuity in the spectrum that causes the search to be highly inefficient. Carrying out the integer estimation (2.6) will be a very time consuming task. As we saw in section 3.5, the integer Gauss transformation decorrelates the ambiguities, but it leaves the conditional variances, and thus the discontinuity, intact. Looking at the definition of conditional variance, it seems logical to change the order of the ambiguities if we want to change the spectrum. In [6], [12] and [13] this problem, and the solution for it, is explained geometrically in terms of the form and orientation of the ambiguity search space.

To change the order of the i-th and the $(i+1)$-th ambiguity we again apply a local transformation but now using the 2-by-2 permutation matrix P for Z_{22}. With (3.16), (3.17) and (3.18) we have

$$\begin{aligned} \bar{L}_{32} &= L_{32}P && (3.25)\\ \bar{L}_{22}^*\bar{D}_{22}\bar{L}_{22} &= P^*L_{22}^*D_{22}L_{22}P && (3.26)\\ \bar{L}_{21} &= \bar{L}_{22}(L_{22}P)^{-1}L_{21} && (3.27) \end{aligned}$$

where

$$P = P^* = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

If we work out these relations we get

$$\bar{L}_{32} = \begin{bmatrix} l_{i+2,i+1} & l_{i+2,i} \\ l_{i+3,i+1} & l_{i+3,i} \\ \vdots & \vdots \\ l_{n,i+1} & l_{n,i} \end{bmatrix} \qquad (3.28)$$

$$\bar{L}_{22} = \begin{bmatrix} 1 & 0 \\ l'_{i+1,i} & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{l_{i+1,i}d_{i+1}}{d_i + l_{i+1,i}^2 d_{i+1}} & 1 \end{bmatrix} \qquad (3.29)$$

$$\bar{D}_{22} = \begin{bmatrix} d'_i & 0 \\ 0 & d'_{i+1} \end{bmatrix} = \begin{bmatrix} d_{i+1} - \frac{l_{i+1,i}^2 d_{i+1}^2}{d_i + l_{i+1,i}^2 d_{i+1}} & 0 \\ 0 & d_i + l_{i+1,i}^2 d_{i+1} \end{bmatrix} \qquad (3.30)$$

$$\bar{L}_{21} = \begin{bmatrix} -l_{i+1,i} & 1 \\ \frac{-l_{i+1,i}^2 d_{i+1}}{d_i + l_{i+1,i}^2 d_{i+1}} + 1 & \frac{l_{i+1,i}d_{i+1}}{d_i + l_{i+1,i}^2 d_{i+1}} \end{bmatrix} L_{21} \qquad (3.31)$$

Simplifying equations (3.29)-(3.31) yields

$$\begin{aligned} d'_{i+1} &= d_i + l^2_{i+1,i} d_{i+1} && (3.32) \\ d'_i &= \frac{d_i}{d'_{i+1}} d_{i+1} && (3.33) \\ l'_{i+1,i} &= \frac{d_{i+1}}{d'_{i+1}} l_{i+1,i} && (3.34) \end{aligned}$$

and

$$\bar{L}_{21} = \begin{bmatrix} -l_{i+1,i} & 1 \\ \frac{d_i}{d'_{i+1}} & l'_{i+1,i} \end{bmatrix} L_{21} \qquad (3.35)$$

The factors L and D are updated, as shown in (3.14) and (3.15), as to correspond to the transformed (permuted) ambiguities.

3.7 Putting it all together

For the actual integer minimization we strive for largely decorrelated ambiguities, and furthermore to have the most precise ambiguity at position n. The reason for the latter was explained in [1]. In other words, we strive for

$$d_n \leq \ldots \leq d_1 \text{ with } d_i \text{ from } D \text{ of } Q_{\hat{a}} = L^* D L$$

and therefore we interchange two conditional variances if

$$d'_{i+1} < d_{i+1} \qquad (3.36)$$

Alternatingly we will have a decorrelation and a reordering step. We start with the last ambiguity and we try to reach the first one. At each step i we check whether the interchange of d_i and d_{i+1} will decrease the value for the latter. After each interchange we start again at the last ambiguity. The algorithm ends if during one sweep from n to 1 no further interchanges can be performed.

The algorithm requires element $l_{i+1,i}$ to be as small as possible. Therefore we take care that the absolute values of the off-diagonal elements of columns $i \ldots n$ are less or equal to .5, by applying the column wise decorrelating Z-transformation of section (3.5). In algorithm *SRC1* we keep track by variable $i1$ what is the last column fulfilling this criterion of 'cleanness'. The logical 'sw' keeps track if there was an interchange of diagonal elements during a sweep. If during one complete sweep from n to 1, no diagonal elements were interchanged, the algorithm terminates.

Algorithm *SRC1*: Given the unit lower triangular factor L and the diagonal matrix D from the L^*DL factorization (third option in (3.8)) of the variance-covariance matrix $Q_{\hat{a}}$, the Z-transformation matrix is computed. On input the matrix Z is set to the unit matrix. The vector with estimates $\hat{a}$ is transformed to $\hat{z} = Z^* \hat{a}$ (thus using the transpose of Z). The original factor L and diagonal matrix D are transformed to L' and D' for which holds: $L'^* D' L' = Q_{\hat{z}}$.

```
$i1 = n - 1$ ; $sw =$ true
while $sw =$ true
    $i = n$ ; $sw =$ false
    while $sw =$ false and $i > 1$
        $i = i - 1$
        if $i \leq i1$
            apply algorithm $ZTRAN$ to column $i$
        end
        $\delta = D(i,i) + L(i+1,i)^2 D(i+1,i+1)$                      cf. eq. (3.32)
        if $\delta < D(i+1,i+1)$
            $\lambda_3 = D(i+1,i+1)L(i+1,i)/\delta$ ; $\eta = D(i,i)/\delta$        cf. eq. (3.34)
            $D(i,i) = \eta D(i+1,i+1)$                                  cf. eq. (3.33)
            $D(i+1,i+1) = \delta$
            for $j = 1 : i-1$
                $\lambda_1 = L(i,j)$ ; $\lambda_2 = L(i+1,j)$
                $L(i,j) = -L(i+1,i)\lambda_1 + \lambda_2$                   cf. eq. (3.35)
                $L(i+1,j) = \eta\lambda_1 + \lambda_3\lambda_2$
            end
            $L(i+1,i) = \lambda_3$
            swap columns $L(i+2:n,i)$ and $L(i+2:n,i+1)$     cf. eq. (3.28)
            swap columns $Z(1:n,i)$ and $Z(1:n,i+1)$
            swap elements $a(i)$ and $a(i+1)$
            $i1 = i$ ; $sw =$ true
        end
    end
end
```

3.8 Example Z-transformation

As an illustration of the Z-transformation we will apply it to an artificial three-dimensional example. The synthetic variance covariance matrix results from the addition of a scaled unit matrix and a rank-2 matrix with elements that are significantly larger than the scale factor of the first matrix, see also [6].

$$Q_{\hat{a}} = \begin{bmatrix} 6.290 & 5.978 & 0.544 \\ 5.978 & 6.292 & 2.340 \\ 0.544 & 2.340 & 6.288 \end{bmatrix} \tag{3.37}$$

The ambiguities have large variances and in particular a_1 and a_2 are heavily correlated.

$$\begin{array}{ll} \sigma_{\hat{a}_1} = 2.508 & \rho_{\hat{a}_1\hat{a}_2} = 0.950 \\ \sigma_{\hat{a}_2} = 2.508 & \rho_{\hat{a}_1\hat{a}_3} = 0.086 \\ \sigma_{\hat{a}_3} = 2.508 & \rho_{\hat{a}_2\hat{a}_3} = 0.372 \end{array} \tag{3.38}$$

The matrix Z^* reads

$$Z^* = \begin{bmatrix} 1 & -1 & 0 \\ -2 & 3 & -1 \\ 3 & -3 & 1 \end{bmatrix} \tag{3.39}$$

Note that the matrix Z^* given here differs from the one obtained using algorithm *SRC1*. Reflections (change of sign) and permutations (reordering) of the ambiguities do not change the amount of correlation between the ambiguities. A permutation of the transformed ambiguities has been applied for plotting purposes (cf. figure 4.5).

Matrix Z^* has integer elements and $|Z^*| = 1$, see [3]. The variance covariance matrix of the transformed ambiguities z reads

$$Q_{\hat{z}} = \begin{bmatrix} 0.626 & 0.230 & 0.082 \\ 0.230 & 4.476 & 0.334 \\ 0.082 & 0.334 & 1.146 \end{bmatrix} \tag{3.40}$$

and it can be seen that the ambiguities are largely decorrelated.

$$\begin{array}{ll} \sigma_{\hat{z}_1} = 0.791 & \rho_{\hat{z}_1\hat{z}_2} = 0.137 \\ \sigma_{\hat{z}_2} = 2.116 & \rho_{\hat{z}_1\hat{z}_3} = 0.097 \\ \sigma_{\hat{z}_3} = 1.071 & \rho_{\hat{z}_2\hat{z}_3} = 0.147 \end{array} \tag{3.41}$$

3.9 Concluding remarks

In this chapter we discussed the implementation aspects of the first part of the LAMBDA method, viz. the transformation that decorrelates the ambiguities. The second part, the sequential conditional least-squares adjustment which translates into a depth-first search through the ambiguity search space, defined by the variance-covariance matrix of the ambiguities, will be the topic of chapter 4.

The transformation was materialized in matrix Z in this chapter. After the actual integer minimization, the new ambiguities z have to be transformed back to the original ambiguities a, see section 4.13. Several alternatives are possible. They are discussed in [18].

One can also compute instead of Z, the matrix Z^{-*} directly. In algorithm *ZTRAN*, section 3.5, subtracting μ times column j from column i, yields the new column i. For computing Z^{-*} one has instead to add μ times column i to column j, to obtain the new column j.

With other alternatives, the transformation is constructed in an *implicit* manner. The fixed solution (2.7) can be computed using the transformed integer ambiguities $\check{z}$ directly, cf. equation (3) in [2]. Instead of starting with $Z = I$ on the input and computing matrix Z, one can insert matrix $Q_{\hat{b}\hat{a}}$ and transform it into $Q_{\hat{b}\hat{z}}$. Matrix $Q_{\hat{z}}^{-1}$ is obtained from the updated factors L and D.

4 Integer ambiguity estimation: search

4.1 Introduction

In this chapter the actual integer ambiguity estimation will be discussed. The integer estimation is also referred to as search. Based on the results of the float solution (see section 2.3) a search will be performed in order to come up with the most likely integer candidate[1] for the vector of ambiguities.

Decorrelating the ambiguities, as discussed in the previous chapter, is not a prerequisite for the integer ambiguity estimation. The search can be performed on the original ambiguities a as well, instead of on the transformed ambiguities z. The decorrelation, however, is largely beneficial for the efficiency of the search, see [4]-[6]. In the sequel the ambiguities are denoted by a, whether they represent transformed ambiguities or not.

The implementation of the search is based on the correspondence of the LDL^*-decomposition of matrix $Q_{\hat{a}}^{-1}$ (the first option of (3.8)) and the sequential conditional least-squares estimation, see section 5 of [1]. The input of this step consists of matrices L and D and the real valued estimate $\hat{a}$. The output is the integer least-squares estimate $\check{a}$.

The sequential conditional adjustment is treated in a straightforward manner in the sections 4.2 and 4.3. The implementation is given in 4.5 and 4.6. The size of the ambiguity search ellipsoid can be controlled prior to the search by means of the volume: sections 4.9 through 4.11. The search is then illustrated by means of an example with three ambiguities. As the search is made on the transformed ambiguities z, a back transformation is needed, section 4.13. It finally provides the integer least-squares estimate for the vector of original double difference ambiguities: $\check{a}$.

[1] What is a candidate? It is a grid point that is inside or on the ambiguity search ellipsoid; it satisfies equation (4.1)

4.2 Sequential conditional least-squares estimation

As discussed in [1], no standard techniques are available for solving (2.6). A discrete search is employed instead. An ellipsoidal region in R^n is taken, on the basis of which a search is performed for the minimizer of (2.6).

$$(\hat{a}-a)^* Q_{\hat{a}}^{-1} (\hat{a}-a) \leq \chi^2 \tag{4.1}$$

For a discussion on the value for χ^2, the constant that controls the size of the ellipsoidal region, the reader is referred to [15], [19] and sections 4.9-4.11 on the volume of the ellipsoidal region.

With the LDL^*-decomposition of matrix $Q_{\hat{a}}^{-1}$, expanding (4.1) yields

$$\sum_{i=1}^{n} d_i \left[(a_i - \hat{a}_i) + \sum_{j=i+1}^{n} l_{ji} \, (a_j - \hat{a}_j) \right]^2 \leq \chi^2 \tag{4.2}$$

Equation (4.2) is just an algebraic development of (4.1). In section 4.3 we will pursue on this development, as the algorithm for the integer estimation is based on (4.2).

As mentioned above, the search can also be given a statistical interpretation: the sequential conditional adjustment. The term between the square brackets is the difference of a_i and $\hat{a}_{i|i+1,\ldots,n}$ and together with (3.11), (4.2) can be rewritten in

$$\sum_{i=1}^{n} \frac{(a_i - \hat{a}_{i|i+1,\ldots,n})^2}{\sigma^2_{\hat{a}_{i|i+1,\ldots,n}}} \leq \chi^2 \tag{4.3}$$

see also equation (25) of [1]. The conditional estimate $\hat{a}_{i|i+1,\ldots,n}$ is the estimate for a_i conditioned on a_j with $j = i+1, \ldots, n$. The conditional estimate for ambiguity i thus reads

$$\hat{a}_{i|i+1,\ldots,n} = \hat{a}_i - \sum_{j=i+1}^{n} l_{ji}(a_j - \hat{a}_j) \tag{4.4}$$

Equation (4.4) clearly shows that conditioning on a_j for $j = i+1, \ldots, n$ affects the estimate for a_i due to the correlation between the ambiguities. Only in case there is no correlation, $L = I$, we have

$$\hat{a}_{i|i+1,\ldots,n} = \hat{a}_i \tag{4.5}$$

In the integer ambiguity estimation using the sequential conditional least-squares adjustment, the ambiguities a_j are conditioned on integers. The variance of the conditional estimator equals $\sigma^2_{\hat{a}_{i|i+1,\ldots,n}} = d_i^{-1}$, see (3.11).

4.3 Computation of the bounds

By means of a sequential conditional adjustment, the full ellipsoid will be searched through for candidates for the vector of ambiguities. From (4.2) we can construct the following bounds for ambiguity a_{i+1}; the ambiguities a_n through a_{i+2} are already conditioned, the ambiguities a_i through a_1 are not conditioned yet.

$$\begin{aligned}&\underbrace{\left[(a_{i+1}-\hat{a}_{i+1})+\sum_{j=i+2}^{n} l_{j,i+1}(a_j-\hat{a}_j)\right]^2}_{left_{i+1}} \leq \\ &\qquad \underbrace{\frac{\chi^2}{d_{i+1}}-\frac{1}{d_{i+1}}\sum_{l=i+2}^{n} d_l\left[(a_l-\hat{a}_l)+\sum_{j=l+1}^{n} l_{jl}(a_j-\hat{a}_j)\right]^2}_{right_{i+1}}\end{aligned} \tag{4.6}$$

Once ambiguity $i+1$ has been fixed to integer a_{i+1}, we compute the bounds for ambiguity i:

$$\begin{aligned}&\underbrace{\left[(a_i-\hat{a}_i)+\sum_{j=i+1}^{n} l_{ji}(a_j-\hat{a}_j)\right]^2}_{left_i} \leq \\ &\qquad \underbrace{\frac{\chi^2}{d_i}-\frac{1}{d_i}\sum_{l=i+1}^{n} d_l\left[(a_l-\hat{a}_l)+\sum_{j=l+1}^{n} l_{jl}(a_j-\hat{a}_j)\right]^2}_{right_i}\end{aligned} \tag{4.7}$$

Equations (4.6) and (4.7) hold for $i \in [1, n-1]$. To show that the bounds can be computed recursively, we split off the term $l = i+1$ from the summation on the right hand side of (4.7),

$$\begin{aligned}&\underbrace{\left[(a_i-\hat{a}_i)+\sum_{j=i+1}^{n} l_{ji}\,(a_j-\hat{a}_j)\right]^2}_{left_i} \leq \\ &\frac{d_{i+1}}{d_i}\{\underbrace{\frac{\chi^2}{d_{i+1}}-\frac{1}{d_{i+1}}\sum_{l=i+2}^{n} d_l\left[(a_l-\hat{a}_l)+\sum_{j=l+1}^{n} l_{jl}\,(a_j-\hat{a}_j)\right]^2}_{right_{i+1}} \\ &\qquad -\underbrace{\left[(a_{i+1}-\hat{a}_{i+1})+\sum_{j=i+2}^{n} l_{j,i+1}\,(a_j-\hat{a}_j)\right]^2}_{left_{i+1}}\}\end{aligned} \tag{4.8}$$

Recognizing on the right hand side the terms $right_{i+1}$, see (4.6) and $left_{i+1}$, the equation can be simplified to

$$\underbrace{\left[(a_i - \hat{a}_i) + \sum_{j=i+1}^{n} l_{ji}(a_j - \hat{a}_j)\right]^2}_{left_i} \leq \underbrace{\frac{d_{i+1}}{d_i}(right_{i+1} - left_{i+1})}_{right_i} \tag{4.9}$$

This shows that, in the sequential conditional adjustment, the bounds for the ambiguities $n-1$ through 1 can be computed recursively. The recursion starts with the conditioning of a_n (substitution of $i = n$ in (4.2)):

$$\underbrace{(a_n - \hat{a}_n)^2}_{left_n} \leq \underbrace{\frac{\chi^2}{d_n}}_{right_n} \tag{4.10}$$

The interval with valid integers for ambiguity a_i follows now from elaboration on (4.9)

$$|(a_i - \hat{a}_i) + \sum_{j=i+1}^{n} l_{ji}(a_j - \hat{a}_j)| \leq \sqrt{right_i} \tag{4.11}$$

or

$$-\sqrt{right_i} \leq (a_i - \hat{a}_i) + \sum_{j=i+1}^{n} l_{ji}(a_j - \hat{a}_j) \leq \sqrt{right_i} \tag{4.12}$$

Equation (4.12) can be further developed into

$$-\sqrt{right_i} - \sum_{j=i+1}^{n} l_{ji}(a_j - \hat{a}_j) \leq a_i - \hat{a}_i \leq \sqrt{right_i} - \sum_{j=i+1}^{n} l_{ji}(a_j - \hat{a}_j) \tag{4.13}$$

and

$$\hat{a}_i - \sqrt{right_i} - \sum_{j=i+1}^{n} l_{ji}(a_j - \hat{a}_j) \leq a_i \leq \hat{a}_i + \sqrt{right_i} - \sum_{j=i+1}^{n} l_{ji}(a_j - \hat{a}_j) \tag{4.14}$$

This defines the interval for ambiguity i. It will be searched through in a straightforward manner from left to right, i.e. from the lower to the upper bound. Each valid integer will be tried, one at a time, and the adjustment proceeds with the next ambiguity a_{i-1} (the so-called depth-first search). If for a certain ambiguity a_l no valid integers can be found, one returns to the previous ambiguity a_{l+1} and takes the next valid integer for this ambiguity. Once an integer is encountered that satisfies interval (4.14) for ambiguity a_1, a full candidate vector is found. The search terminates when all valid integers encountered, have been treated and one is back at the last ambiguity a_n. The full ellipsoid has been searched through.

To summarize: the sequential adjustment starts with a conditioning on a_n and ends with a conditioning on a_1. In this way the bounds for the ambiguities a_n through a_1

are constructed in a recursive way. When the sequential adjustment is at ambiguity i, with the definition of the conditional estimate (4.4), the interval (4.12) can be rewritten into

$$-\sqrt{right_i} \leq (a_i - \hat{a}_{i|i+1,\ldots,n}) \leq \sqrt{right_i} \tag{4.15}$$

which shows that the interval for a_i is centered at the conditional estimate $\hat{a}_{i|i+1,\ldots,n}$. At this moment the integer nearest to the conditional estimate, $nint(\hat{a}_{i|i+1,\ldots,n})$, is the most likely candidate for ambiguity i. From (4.3) it can be seen that the conditional variances play a decisive role in the bounds for the ambiguities. The smaller the conditional variance, the smaller the interval (4.14).

4.4 Computation of the norm

In the previous section the full ellipsoid was searched through and as a result we have available all grid points that are inside the ellipsoid. One of them, the one which yields the minimum for (2.6), is the integer least-squares estimate $\check{a}$.

The squared norm $t(a) = \|\hat{a}-a\|^2_{Q_{\hat{a}}^{-1}}$ of a candidate can be computed by substitution of a into (4.1). It can also be computed from the bounds for ambiguity a_1 at the moment the candidate is encountered in the search. The squared norm, the left hand side of (4.2), can be rewritten into (by respectively adding and subtracting χ^2 and splitting off the first term)

$$t(a) = \sum_{i=1}^{n} d_i \left[(a_i - \hat{a}_i) + \sum_{j=i+1}^{n} l_{ji}(a_j - \hat{a}_j) \right]^2$$

$$= \chi^2 - d_1 \{ \underbrace{\frac{\chi^2}{d_1} - \frac{1}{d_1} \sum_{i=2}^{n} d_i \left[(a_i - \hat{a}_i) + \sum_{j=i+1}^{n} l_{ji}(a_j - \hat{a}_j) \right]^2}_{right_1}$$

$$- \underbrace{\left[(a_1 - \hat{a}_1) + \sum_{j=2}^{n} l_{j1}(a_j - \hat{a}_j) \right]^2}_{left_1} \} \tag{4.16}$$

The squared norm $t(a)$ is less or equal to χ^2, as the grid point is on or inside the ellipsoid.

If, at level $i = 1$, more than one candidate is available, the squared norm of successive candidate vectors a' can be computed very easily once the squared norm of one candidate a has been computed already. This is because $a'_i = a_i$ for $i = 2, \ldots, n$ and

$a_1' = a_1 + \nabla$ (where ∇ can be chosen to be any integer). So now the squared norm becomes

$$t(a') = t(a) + d_1\{2\nabla[\underbrace{(a_1 - \hat{a}_1)}_{dist_1} + \underbrace{\sum_{j=2}^{n} l_{j1}(a_j - \hat{a}_j)}_{lef_1}] + \nabla^2\} \tag{4.17}$$

4.5 The search algorithm

The search algorithm *FI71* is the realization of the sequential conditional estimation.

Algorithm *FI71*: Given the LDL^*-decomposition of $Q_{\hat{a}}^{-1}$, a suitable value for χ^2, and the float ambiguity vector $\hat{a}$, '*maxcan*' candidates inside the ellipsoidal search space with minimum distance to the vector of float ambiguities will be computed.

```
$right(n+1) = \chi^2$ ; $left(n+1) = 0$ ; $ende$=false; $ncan = 0$
for $i = 1 : n-1$
    $dq(i) = D(i+1, i+1)/D(i,i)$
end
$dq(n) = 1/D(i,i)$
$i = n+1$
while $ende$=false
    $i = i-1$
    $lef(i) = dot(L(i+1:n, i), dist(i+1:n))$
    $right(i) = (right(i+1) - left(i+1)) \cdot dq(i)$
    $reach = \sqrt{right(i)}$
    $\delta = a(i) - reach - lef(i)$
    $dist(i) = JNT1(\delta) - a(i)$
    if $dist(i) > reach - lef(i)$
        call $BACKTS$ ($n$, $i$, $end$, $dist$, $lef$, $left$, $ende$)
    else
        $end(i) = reach - lef(i) - 1$
        $left(i) = (dist(i) + lef(i))^2$
    end
    if $i = 1$
        call $COLLECTs$ ($n$, $maxcan$, $D$, $lef$, $left$, $right$, $dist$, $end$, $ncan$,
            $disall$, $cands$, $tmax$, $imax$)
        call $BACKTS$ ($n$, $i$, $end$, $dist$, $lef$, $left$, $ende$)
    end
end
```

If during the search at some level no (more) integers can be found we 'back-track' to a higher level (towards n). This function is performed by *BACKTS*.

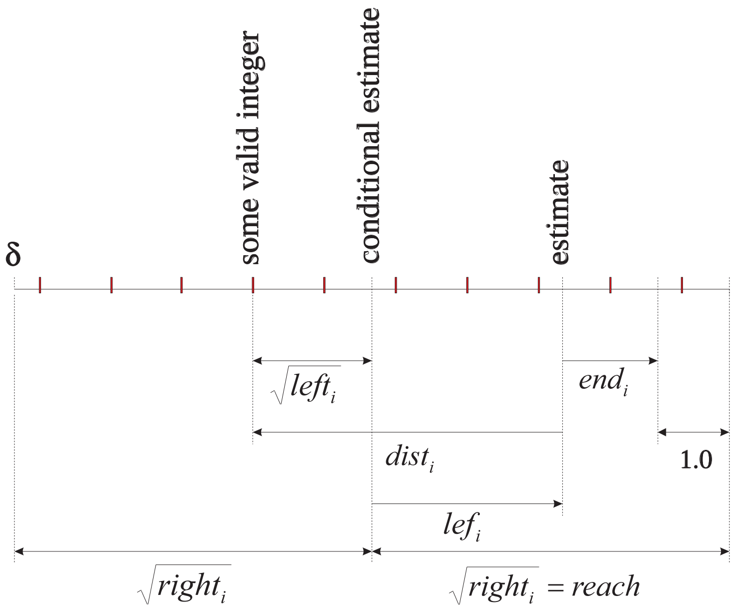

Figure 4.1: The relation between some important variables from *FI71*. An arrow pointing to the right indicates a positive number, an arrow pointing to the left a negative number, and a two-sided arrow indicates that the number is intrinsically positive.

Algorithm *BACKTS* (*n, i, end, dist, lef, left, ende*)

```
     $j = i + 1$
     for $i = j : n$
          if $dist(i) \leq end(i)$
               $dist(i) = dist(i) + 1$
               $left(i) = (dist(i) + lef(i))^2$
               go to 10
          end
     end
     $ende$=true
10   continue
```

The performance of the algorithm could possibly be improved by recognizing that to compute *lef*(i) it is not necessary to form every time the inner product between column i of L and vector *dist*. If we were at this level in the previous cycle, we know that only the value of the i-th ambiguity has changed; more specifically, its value has increased by 1. Consequently only *dist*(i) has decreased by 1. This means that we can compute the new inner product form the inner product at the previous cycle by adding $L(i+1,i) \cdot 1$ to it. To incorporate this in *FI71* we have to add

$iold = i$

just before the '**while**'-loop, and replace

$lef(i) = dot(L(i+1:n,i), dist(i+1:n))$

by

```
**if** $iold \leq i$
    $lef(i) = lef(i) + L(i+1,i)$
**else**
    $lef(i) = dot(L(i+1:n,i), dist(i+1:n))$
**end**
$iold = i$
```

For the function *JNT1* (‘ceil’ (d) = rounding d towards $+\infty$) there are several implementations possible; use the one that works fastest on your computer.

```
Function $JNT1(d)$

$JNT1(d) = int(d - int(d) - 1) + int(d) + 1$
```

or

```
Function $JNT1(d)$

**if** $d \leq 0$
    $JNT1 = int(d)$
**else**
    $JNT1 = int(d) + 1$
**end**
```

4.6 Collecting and storing the candidates

Most of the validation techniques in use nowadays, need two integer candidates, viz. the one which has minimum norm (the best), and the second best. Algorithm *COLLECTs* takes care for the storage of the candidates.

If on level $i = 1$ a valid integer is encountered, a full candidate vector has been found. Its squared norm is computed, and the candidate and its norm are stored if, 1. the number of candidates found thus far is less than the number of candidates required (‘*maxcan*’), or 2. the squared norm is less than the maximum of the squared norms of the candidates stored thus far.

```
Algorithm COLLECTs ($n$, $maxcan$, $D$, $lef$, $left$, $right$, $\chi^2$, $dist$, $end$,
    $ncan$, $disall$, $cands$, $tmax$, $imax$)

$t = \chi^2 - (right(1) - left(1)) \cdot D(1,1)$
$end(1) = end(1) + 1$
```

```
while $dist(1) \leq end(1)$
    $ncan = ncan + 1$
    if $ncan \leq maxcan$
        call $STOREs(ncan, ncan, imax, t, tmax, dist, cands, disall)$
    else
        if $t < tmax$
            call $STOREs(maxcan, imax, imax, t, tmax, dist, cands, disall)$
        end
    end
    $t = t + (2 \cdot (dist(1) + lef(1)) + 1) \cdot D(1,1)$
    $dist(1) = dist(1) + 1$
end
```

Algorithm *STOREs*(*ican, ipos, imax, t, tmax, dist, cands, disall*)

```
for $i = 1 : n$
    $cands(i, ipos) = dist(i)$
end
$disall(ipos) = t$
$tmax = t$
$imax = ipos$
for $i = 1 : ican$
    if $disall(i) > tmax$
        $imax = i$
        $tmax = disall(i)$
    end
end
```

When *maxcan* candidates were requested and *ncan* candidates were found, then the best *min*(*ncan*, *maxcan*) candidates are stored by *STOREs*. Since the array *cands* contains the difference between the vector of float ambiguities and the integer candidates, the integer least-squares estimate is obtained by

$$\check{a} = \hat{a} + cands(1 : n, k)$$

where k is taken such that $dist(k) \leq dist(i)\ \forall\ i = 1, min(ncan, maxcan)$.

As a result of the search, the integer least-squares estimate $\check{a}$ is obtained. *disall(k)* contains the squared norm $t(\check{a}) = \|\hat{a} - \check{a}\|^2_{Q_{\hat{a}}^{-1}}$. Current validation procedures require, besides $\check{a}$, also the second best candidate $\check{a}'$. If requested (and present) also this second best candidate can be retrieved from the array *cands*, with $t(\check{a}') = \|\hat{a} - \check{a}'\|^2_{Q_{\hat{a}}^{-1}}$.

Note: When handling integer parameters on a computer, one must take care of their magnitude. The range for integer parameters usually is much smaller than that for real (double precision) parameters. It also is machine and/or compiler dependent.

To avoid that integer overflows occur, the magnitude of the original float ambiguities $\hat{a}$ is reduced such that it is between -1.0 and $+1.0$. This is performed prior to the decorrelation step, and can for example be done by the modulo function:

$$\hat{a} := mod(\hat{a}, 1.0) \tag{4.18}$$

Through this *integer* translation, the center of the ellipsoid is shifted by the vector $mod(\hat{a}, 1.0)$. The integer part $\hat{a} - mod(\hat{a}, 1.0)$ is stored, and added again to the fixed ambiguity $\check{a}$ afterwards.

4.7 Alternating search around the conditional estimate

In this and the following section, we will, as a side step, mention two implementations, that are alternatives to the straightforward search treated so far.

Instead of scanning the interval per ambiguity (4.14) from left to right for integers, one can search in an alternating way around the conditional estimate. In the conditional estimation we will start by conditioning ambiguity a_i to integer $nint(\hat{a}_{i|i+1,\ldots,n})$. Next, the second nearest, third nearest, etc are tried. The candidates for a_i are chosen in a sequence, that starts with the one closest to the center of the interval (4.15) and goes alternatingly towards the bounds, section 2.4 in [2].

With this strategy one tries, in each step of the sequential conditioning, to stay as far as possible from the borders of the ellipsoidal region (4.1).

When all candidates contained in the ellipsoid are required, this strategy has no benefit over the straightforward search in which the candidates per interval are taken from 'left to right'. When used in connection with the shrinking strategy, and when only one of a great number of candidates within the ellipsoid is required, the strategy of searching alternatingly around the conditional estimate will have benefit.

4.8 Shrinking the ellipsoidal region

The best candidate, the grid point nearest to $\hat{a}$, is the integer least-squares estimate for the vector of ambiguities. As we are in principle interested in only this candidate, the search can be designed to find it as quickly as possible. The sequential conditional estimation aims in the first place at constructing a complete n-dimensional vector with integers that fulfills (4.3), or in other words aims at finding a grid point that is inside the ellipsoid.

As soon as such a vector $\bar{a}$ is found, the corresponding squared norm $\|\hat{a} - \bar{a}\|^2_{Q_{\hat{a}}^{-1}}$ is taken as a new value for χ^2. We shrink the ellipsoidal region. The sequential conditional estimation is then continued (not started over!) in this shrunken ellipsoid, section 5 of [1]. If one, possibly after repeated shrinking, fails to find a candidate in the ellipsoid, the last found integer vector is the sought for integer least-squares estimate, $\check{a} = \bar{a}$, see section 2.4 of [2].

4.9 The volume of the ellipsoidal region

In the next three sections we will elaborate on how to control, prior to the search, the size of the ambiguity search space. We will recognize the relation between the volume of the ellipsoid and the number of candidates contained. By this relation, the straightforward search is an effective instrument in solving the minimization problem (2.6).

The volume, expressed in [cycles^n], of the ellipsoidal region (4.1) is given by

$$E_n = \chi^n \sqrt{|Q_{\hat{a}}|} V_n \tag{4.19}$$

see [14]. The volume function in (4.19) is

$$V_n = \frac{2}{n} \frac{\pi^{\frac{n}{2}}}{\Gamma(\frac{n}{2})} \tag{4.20}$$

where Γ is the gamma function, defined as

$$\Gamma(x) = \int_0^\infty e^{-t} t^{x-1} dt \text{ for } x > 0 \tag{4.21}$$

The volume function can be computed recursively for $n \geq 3$ by

$$V_n = \frac{2\pi}{n} V_{n-2} \tag{4.22}$$

with

$$V_1 = 2 \text{ and } V_2 = \pi \tag{4.23}$$

For the determinant of the variance covariance matrix we have the following relations

$$|Q_{\hat{a}}| = \prod_{i=1}^{n} \lambda_i = \prod_{i=1}^{n} \sigma^2_{\hat{a}_{i|i+1,\ldots,n}} \tag{4.24}$$

where λ_i is the i-th eigenvalue of matrix $Q_{\hat{a}}$. The volume E_n can thus easily be computed, as the conditional variances are available from matrix D, see (3.11).

4.10 Setting χ^2, I

The volume E_n turns out to be a good indicator for the number of candidates (grid points) contained in the ellipsoid. A volume of E_n cycles^n corresponds to approximately $k = nint(E_n)$ candidates. For k less than a few, the relation is not accurate. The mismatch is caused by the discrete nature of the grid points. Centering the same ellipsoid at a different location, may result in a different number of candidates, $k \in Z$, while the volume $E_n \in R$, remains unchanged.

The value χ^2 can be taken such that a certain number of candidates will be inside the ellipsoidal region. A straightforward search can then be performed to obtain the

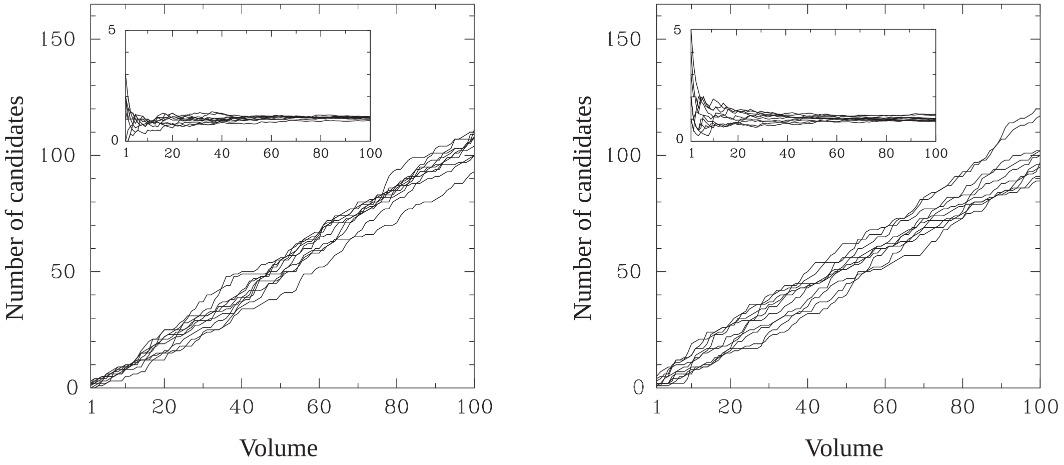

Figure 4.2: Number of candidates inside ellipsoid versus volume; left: single frequency phase data, right: dual frequency phase data

requested number of candidates. A list with the best k candidates, possibly ordered after their norms, can be set up and updated during the sequential adjustment. In this way the size of the ellipsoid is controlled prior to the search. To some extent one can already infer the quality of the integer estimator in advance.

Figure 4.2 concerns ten experiments each with two epochs (sampling interval 1 second) of single frequency data (left) and dual frequency phase data (right) to seven satellites on a 2.2 km baseline (with dual frequency this gives 12 ambiguities). The actual number of candidates contained in the ambiguity search ellipsoid is given as function of the volume of the ellipsoid [cycles^{12}]. The volume ranges from 1 to 100. All lines run under 45 degrees approximately. The volume is believed to be a good indicator for the actual number of candidates in the ambiguity search ellipsoid. The inset shows the quotient of the number of candidates and the volume, i.e. the relative error of the volume as a predictor for the number of candidates.

4.11 Setting χ^2, II

Another method to set the value for χ^2 such that at least two candidates are contained in the ellipsoidal search space, was proposed in [16], see also [19]. Since the decorrelated ambiguities have such a high precision (typically some tenths of a cycle), rounding to the nearest integer will produce a candidate with a norm close to the minimum. So setting χ^2 equal to the squared norm of this candidate will guarantee at least one and most probably not more than a few candidates. Other candidates with small norms can be found through rounding all ambiguities but one to their nearest integer, and one ambiguity to the next-nearest integer. If we have n ambiguities, this will give us n more candidates with likely small norms. Setting χ^2 to the one-but-smallest squared norm, will guarantee now at least two candidates, and most likely not more than a few.

These $n+1$ squared norms can be computed in an efficient way using again the LDL^*-decomposition of $Q_{\hat{a}}^{-1}$. Re-arranging the first part of equation (4.16) gives

$$\begin{aligned} t(a) &= \sum_{i=1}^{n} d_i \left[\sum_{j=i}^{n} l_{ji} \left(a_j - \hat{a}_j \right) \right]^2 && (4.25) \\ &= \sum_{i=1}^{n} d_i e_i^2 && (4.26) \end{aligned}$$

(Note that e_i^2 equals left_i.) Equation (4.17) showed an efficient way for computing the squared norm $t(a')$ for a candidate which differs ∇ in the last ambiguity a_1 from a candidate with known squared norm $t(a)$. We will now generalize this for the case that an arbitrary ambiguity k differs ∇:

$$a_{,k} = a + c_k \nabla \qquad (4.27)$$

with c_k a vector with zeros on all positions with exception of position k which has the value one:

$$c_k = (\underbrace{0,0,\ldots,}_{k-1} 1, \underbrace{0,0,\ldots,0}_{n-k})$$

The squared norm for the vector $a_{,k}$ can be written as

$$\begin{aligned} t(a_{,k}) &= \sum_{i=1}^{k} d_i (e_i + l_{ki}\nabla)^2 + \sum_{i=k+1}^{n} d_i e_i^2 \\ &= \sum_{i=1}^{k} d_i (e_i^2 + 2 e_i l_{ki} \nabla + l_{ki}^2 \nabla^2) + \sum_{i=k+1}^{n} d_i e_i^2 \\ &= t(a) + \sum_{i=1}^{k} d_i (2 e_i l_{ki} \nabla + l_{ki}^2 \nabla^2) && (4.28) \end{aligned}$$

Compare this result with that of equation (4.17). There we used $(\mathit{dist}_1 + \mathit{lef}_1)$ instead of e_1 since these are available during the search process. The form shown here is more efficient when we have only the LDL^*-decomposition of $Q_{\hat{a}}^{-1}$, as is the case prior to the search.

We start by taking for a

$$a = (\mathit{nint}(\hat{a}_1), \mathit{nint}(\hat{a}_2), \ldots, \mathit{nint}(\hat{a}_n)) \qquad (4.29)$$

and once we have computed its squared norm $t(a)$, we can compute very efficiently another n likely small norms $t(a_{,k})$ with (4.28) by taking for ∇

$$\nabla = \begin{cases} 1 & \text{if} \quad \mathit{nint}(\hat{a}_k) - \hat{a}_k < 0 \\ -1 & \text{if} \quad \mathit{nint}(\hat{a}_k) - \hat{a}_k > 0 \end{cases} \qquad (4.30)$$

with k from 1 to n. During the computation of the $n+1$ norms we keep track of the smallest and one-but-smallest value. χ^2 is set to the one-but-smallest value for the norm. The ellipsoid will contain at least two candidates.

To illustrate this strategy, the following experiment was done: we computed the squared norm belonging to the vector one gets when rounding all ambiguities to their nearest integer. We also computed the n squared norms one gets when rounding all ambiguities but one to the nearest integer: the remaining one is rounded to the next-nearest integer. With χ^2 set to these norms, the volume of the ambiguity search ellipsoid was computed.

This was done for the same 10 dual frequency experiments for which the relation between volume and number of candidates was demonstrated (see figure 4.2). Per experiment the volumes were sorted after increasing order, and plotted in figure 4.3, i.e. the smallest volume is plotted at $k = 1$, the second smallest at 2, etc. At left we plotted the results for the transformed problem, at right those for the original problem. Looking at the experiments with the transformed ambiguities, one can see that the one-but-smallest volume (which guarantees at least two candidates) never exceeds the value of 10. This translates into a maximum number of candidates of approximately 10 (see figure 4.2).

For the original ambiguities the volumes are in the order 10^{10} - 10^{12}, which shows that no acceptable volumes are obtained if the ambiguities are not decorrelated. Searching through ellipsoids with such large volumes would mean that we have to cope with an enormous number of candidates, which clearly would be detrimental for the efficiency.

For the case of the transformed ambiguities, we actually set χ^2 to the one-but-smallest squared norm. In table 4.1 one can find the epoch numbers for the 10 experiments, the resulting volume and the number of candidates inside the ellipsoid. This number is below 10 for all experiments.

The same was done for a 10.4 km baseline (measured simultaneously with the 2.2 km baseline, observing the same 7 satellites; the baselines have one station in common). Although here we find in some experiments up to 14 candidates, the procedure is still capable to come up with the two minimum candidates in an efficient way due to the almost lack of correlation between the ambiguities.

4.12 Example ambiguity search

The search in the ambiguity ellipsoid will be illustrated using the original ambiguities a of the three-dimensional example of section 3.8. Next, in order to show what is gained by the decorrelating transformation, also the search with the transformed ambiguities z will be demonstrated. This section is concluded by comparing a search with original and transformed ambiguities for a GPS baseline.

The (real valued) estimates for the ambiguities a are:

$$\begin{aligned} \hat{a}_1 &= 5.45 \\ \hat{a}_2 &= 3.10 \\ \hat{a}_3 &= 2.97 \end{aligned} \qquad (4.31)$$

The so-called search tree is depicted in figure 4.4 at left, and should be read from left to right. Per level, i.e. per ambiguity, the candidates encountered are represented by

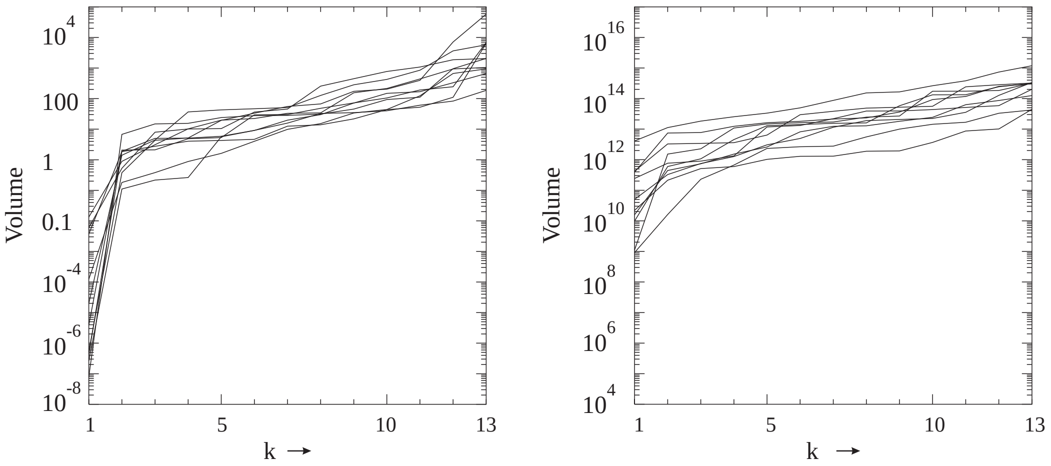

Figure 4.3: Volume of the ambiguity search space containing at least k 'near' candidates; left: transformed ellipsoid, right: original ellipsoid.

epochs	2.2 km		10.4 km	
	volume	number of candidates	volume	number of candidates
1 - 2	0.38	2	8.76	14
31 - 32	0.11	2	1.36	2
61 - 62	1.41	2	13.95	14
91 - 92	6.76	4	5.58	9
121 - 122	1.89	2	1.59	3
151 - 152	1.85	4	2.89	2
181 - 182	2.07	2	0.67	2
211 - 212	0.53	2	0.90	3
241 - 242	0.89	2	9.50	7
271 - 272	0.18	2	1.83	3

Table 4.1: Volume and number of candidates inside the ambiguity search ellipsoid, by setting χ^2 to the one-but-smallest squared norm

a node. A full candidate vector, i.e. a grid point inside the ellipsoid, is found when a node on level a_1 is connected with a node on level a_2, which is connected with a node on level a_3.

The search starts with ambiguity a_3. The interval is centered at $\hat{a}_3 = 2.97$ and ranges from 0.462 to 5.478. Valid integers for a_3 are thus [1,2,3,4,5]. This interval is scanned from left to right. Ambiguity a_3 is conditioned on $a_3 = 1$. Equation (4.14) applied for ambiguity a_2, yields the integers [1,2,3]. After conditioning ambiguity $a_2 = 1$, it is not possible with (4.14) applied to ambiguity a_1, to find valid integers for ambiguity a_1. Conditioning a_1 to an integer will result in a grid point that is outside the ambiguity search ellipsoid, no matter which integer is taken. We proceed with the second candidate for ambiguity $a_2 = 2$. For this candidate we find the integer $a_1 = 5$. As we have reached level 1, we have found a full integer vector, that satisfies (4.1), i.e. a grid point that is inside the ambiguity search ellipsoid. The vector is $(a_1 = 5, a_2 = 2, a_3 = 1)$, see also figure 4.5. The process is continued until no integers are left on any level. The search is terminated and six candidate vectors have been found. The volume of this ambiguity search ellipsoid is $E_3 = 7.3$ cycles^3.

The (real valued) estimates for the ambiguities z are:

$$\begin{aligned} \hat{z}_1 &= 2.35 \\ \hat{z}_2 &= -4.57 \\ \hat{z}_3 &= 10.02 \end{aligned} \tag{4.32}$$

The search tree for the transformed problem is depicted in figure 4.4 at right, the ambiguity search ellipsoid in figure 4.5 at right. In comparing the tree at left with the tree at right in figure 4.4, it can be seen that the search in the transformed problem can be performed much more efficiently. In the tree at left, there are 13 so-called 'dead ends', in the tree at right only 3. For a dead end, computations have to be carried out (the bounds), that do not result in a full candidate vector. It is the discontinuity in the spectrum of conditional variances that causes this so-called 'halting' of the search. This discontinuity is analyzed in [6] and the implications for the search are described in [4]: the first few bounds (4.14) are rather loose, the remaining ones are very tight.

The solution, the integer least-squares estimate, is $\check{a} = (5, 3, 4)$. In the transformed problem we obtain $\check{z} = (2, -5, 10)$. This solution has to be transformed by Z^{-*} and yields exactly the same solution. The squared norm is $t(\check{a}) = 0.218$.

The differences between original and transformed ambiguities are even much larger for real GPS examples. In figure 4.6 we give the number of valid integers per ambiguity, encountered during the full search. They are given for both the original and transformed ambiguities. The data is from a 2.2 km baseline with dual frequency data to seven satellites. Two epochs of data were taken, with one second in between.

When a baseline (three coordinate unknowns) is observed for a short time span, three conditional variances, of the original ambiguities, are very large and the remaining nine ones are very small. From figure 4.6 we see that at level $i = 10$ (the third ambiguity in the search) there are over 3.10^8 candidates. After having proceeded to ambiguity a_1 there are only 2 full candidate vectors left, which implies that there are very many dead ends. The volume of the ambiguity search ellipsoid is $E_{12} = 2.8$

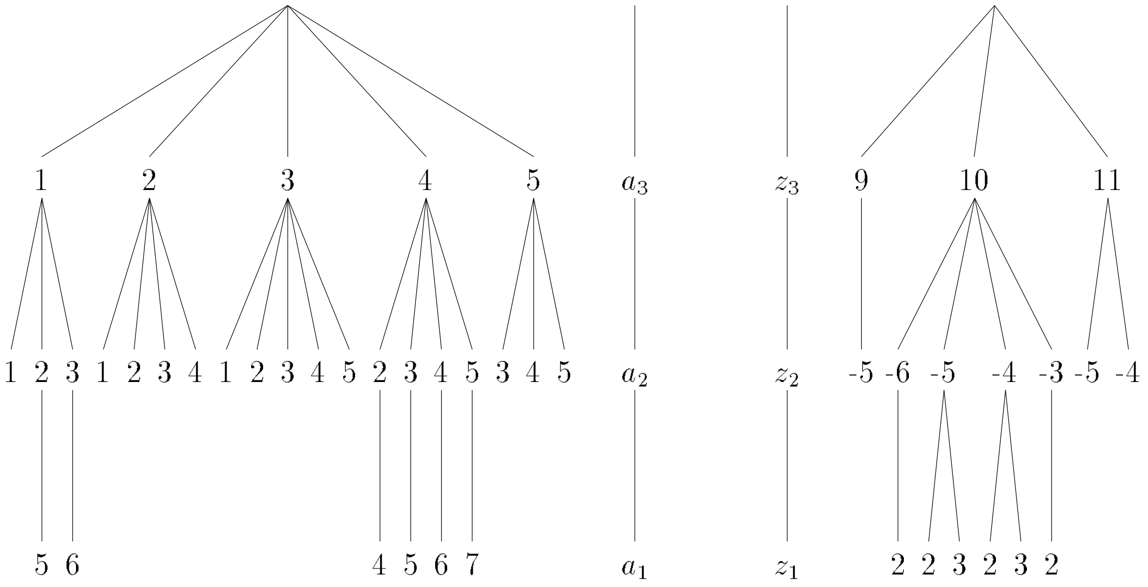

Figure 4.4: The search tree for the original problem (left) and for the transformed problem (right).

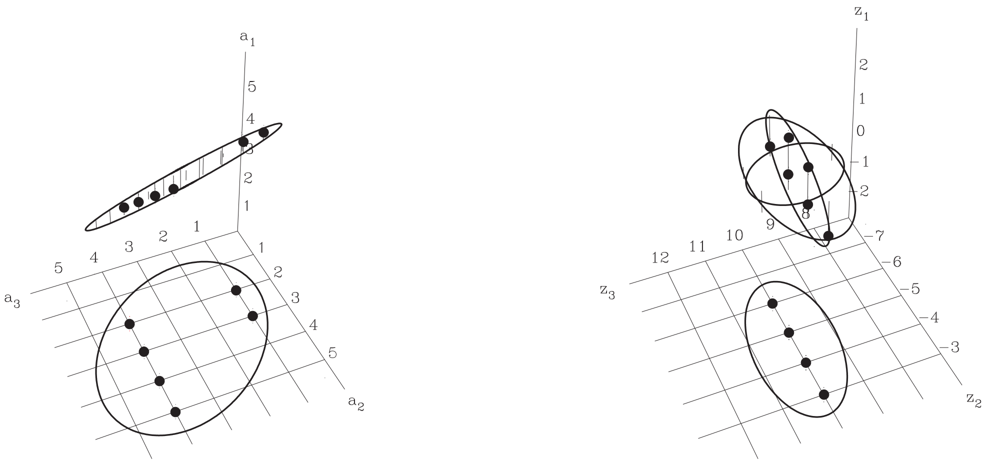

Figure 4.5: The ellipsoid in 3-D and its perpendicular projection onto the 2-3 plane for the original problem (left) and for the transformed problem (right).

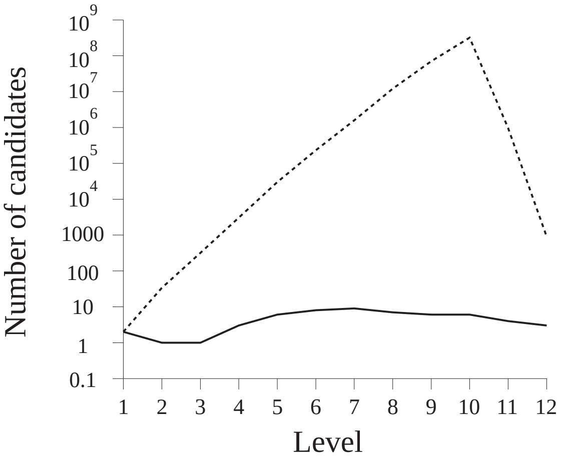

Figure 4.6: Number of candidates per level; dashed line: before transformation, solid line: after transformation.

cycles[12]. By the transformation, the spectrum of conditional variances is flattened, and so is the graph of the number of candidates per level: the search on the transformed ambiguities can be performed very efficiently, there are only a few dead ends left.

4.13 The back transformation

Once the transformed integer minimization problem (3.6) has been solved, we have to back transform the integer estimate $\check{z}$, in order to obtain the integer least-squares estimate $\check{a}$. The relation reads

$$Z^* \check{a} = \check{z}$$

The inverse of the matrix Z^* does not need to be computed explicitly. Matrix Z^* has only integer elements, has full rank, with $|Z^*| = \pm 1$, and is square and usually dense, as the transformation is truly a multi-channel transformation [3]. Therefore an *LU*-factorization can be made by Gaussian elimination with partial pivoting, see chapter 3 of [7] and chapter 1 of [11] (the LINPACK routines DGEFA and DGESL). Then the integer least-squares estimate $\check{a}$ is obtained via for- and backward substitution. Note that the intermediate result – after the forward substitution – is in general *not* an integer vector. The for- and backward substitution can be repeated for any other candidate, e.g. the second best $\check{a}'$.

Alternatives to the back transformation, computing matrix Z^{-*} directly or computing matrix $Q_{\hat{b}\hat{z}}$, are suggested in section 3.9.

4.14 Concluding remarks

The integer ambiguities are estimated in a sequential conditional least-squares adjustment. The ambiguities are constrained to integers, in a sequential manner from ambiguity a_n to a_1. In case the search is performed on the original ambiguities a, they should be ordered according to their conditional precision [1]:

$$\sigma^2_{\hat{a}_{1|2,\ldots,n}} \geq, \ldots, \geq \sigma^2_{\hat{a}_n} \tag{4.33}$$

The sequential adjustment should start with the most precise ambiguity. The decorrelating Z-transformation typically flattens the spectrum of conditional variances: the variances are equalized. This makes that the sequential conditional adjustment can be carried out very efficiently.

5
Flow diagram of the parameter estimation

Procedure for solving the linear system (2.2)

Float solution:

- normal equations $Nx = r$ (equation 2.3)
- Cholesky factorization of normal matrix, $N = CC^*$, e.g. algorithm 4.2.2. [7]
- forward substitution $C(C^*x) = r$, algorithm 3.1.1. [7]
- backward substitution $C^*x = C^{-1}r$, algorithm 3.1.2. [7]

Integer ambiguity estimation:

- split Cholesky factor $G, G = L\sqrt{D} \rightarrow L$ and D (equation 3.9)
- inversion of $D \rightarrow D^{-1}$
- inversion of $L \rightarrow L^{-1}$, LTINV
- construction of matrix Z, transformation of $D^{-1} \rightarrow \tilde{D}^{-1}$, $L^{-1} \rightarrow \tilde{L}^{-1}$ and $\hat{a} \rightarrow \hat{z}$, SRC1
- inversion of $\tilde{D}^{-1} \rightarrow \tilde{D}$
- inversion of $\tilde{L}^{-1} \rightarrow \tilde{L}$, LTINV
- integer minimization, FI71 $\rightarrow \check{z}$ (equation 3.6)
- transposition of $Z \rightarrow Z^*$
- LU-factorization of Z^*, algorithm 3.4.1. [7]
- forward substitution, algorithm 3.1.1. [7]
- backward substitution, algorithm 3.1.2. [7] $\rightarrow \check{a}$

Fixed solution:

- forward substitution, algorithm 3.1.1. [7]
- backward substitution, algorithm 3.1.2. [7] $\rightarrow \check{b}$

6
Concluding remarks

The LAMBDA method for integer estimation of the GPS double difference ambiguities consists of firstly a decorrelation of the ambiguities and secondly a sequential conditional adjustment of the ambiguities. In this report the implementation aspects have been discussed. The integer minimization problem is attacked by a discrete search over an ellipsoidal region, the ambiguity search ellipsoid.

The shape and orientation of the ellipsoid are governed by the variance covariance matrix of the ambiguities. The decorrelation realizes an ellipsoid that is very much sphere-like. It can be searched through very efficiently. The size of the ellipsoid can be controlled prior to the search using the volume function, using one of the two strategies presented. The volume gives an indication of the number of candidates contained in the ellipsoid. Making a request for a few candidates and carrying out the search on the transformed ambiguities, enables a straightforward implementation of the search. A limited number of candidates will be gathered of which one is the integer least-squares estimate for the vector of ambiguities.

For future information concerning the LAMBDA method, we refer to our WWW page:

http://www.geo.tudelft.nl/mgp/lambda

A
Inversion of a triangular matrix

The construction of the Z-transformation matrix is based on the variance-covariance matrix $Q_{\hat{a}}$. With the first option of (3.8) the inverse of the unit lower triangular matrix L has to be computed. The inversion of diagonal matrix D is trivial.

Algorithm *LTINV*: Given the full rank lower triangular matrix L, this algorithm computes the inverse $X = L^{-1}$. The relation between matrix L and its inverse X reads $LX = I$. For element (i, j) of I we have

$$\sum_{k=j}^{i} L(i,k)X(k,j) = \begin{cases} 0 \ i > j \\ 1 \ i = j \end{cases}$$

From this follows that for $i > j$

$$X(i,j) = \frac{\sum_{k=j}^{i-1} L(i,k)X(k,j)}{L(i,i)}$$

and for $i = j$

$$X(i,i) = \frac{1}{L(i,i)}$$

The algorithm reads

```
for $i = 1 : n$
    for $j = 1 : i - 1$
        $X(i,j) = -dot(L(i, j : i-1), X(j : i-1, j))/L(i,i)$
    end
    $X(i,i) = 1/L(i,i)$
end
```

The inverse is computed row wise and in-place; the inverse may overwrite matrix L. Function *dot* computes the dot or inner product of two vectors.

References

[1] Teunissen, P.J.G. (1993). *Least-squares estimation of the integer GPS ambiguities.* Invited lecture, Section IV Theory and Methodology, IAG General Meeting, Beijing, China, August, also in *Delft Geodetic Computing Centre LGR series,* No. 6, 16 pp.

[2] Teunissen, P.J.G. (1995). The least-squares ambiguity decorrelation adjustment: A method for fast GPS integer ambiguity estimation. *Journal of Geodesy,* Vol. 70, No. 1-2, pp. 65-82.

[3] Teunissen, P.J.G. (1995). The invertible GPS ambiguity transformations. *Manuscripta Geodaetica.* Vol. 20, No. 6, pp. 489-497.

[4] Teunissen, P.J.G. (1995). On the GPS double difference ambiguities and their partial search spaces. *Proceedings IAG Symposium No. 114, Geodetic Theory Today,* Springer Verlag, pp. 39-48, III Hotine-Marussi symposium on Mathematical Geodesy, L'Aquila, Italy, May 29-June 3, 1994

[5] Teunissen, P.J.G., and C.C.J.M. Tiberius (1994). Integer least-squares estimation of the GPS phase ambiguities. *Proceedings of International Symposium on Kinematic Systems in Geodesy, Geomatics and Navigation KIS'94,* Banff, Canada, August 30-September 2, pp. 221-231.

[6] Teunissen, P.J.G., P.J. de Jonge, and C.C.J.M. Tiberius (1994). On the spectrum of the GPS DD-Ambiguities. *Proceedings of ION GPS-94, 7th International Technical Meeting of the Satellite Division of the Institute of Navigation,* Salt Lake City, UT, September 20-23, pp. 115-124.

[7] Golub, G.H., and C.F. van Loan (1989). *Matrix computations.* Second edition. The Johns Hopkins University Press, Baltimore, Maryland, USA.

[8] Teunissen, P.J.G. (1994). A new method for fast carrier phase ambiguity estimation. *Proceedings IEEE Position, Location and Navigation Symposium PLANS'94.* Las Vegas, NV, April 11-15, pp. 562-573.

[9] Jonge de, P.J., and C.C.J.M. Tiberius (1994). A new GPS ambiguity estimation method based on integer least squares. *Proceedings of the 3rd International Symposium on Differential Satellite Navigation Systems DSNS'94.* London, UK, April 18-22, Paper No. 73, 9 pp.

[10] Tiberius, C.C.J.M., and P.J. de Jonge (1995). Fast positioning using the LAMBDA-method. *Proceedings of the 4th International Symposium on Differential Satellite Navigation Systems DSNS'95.* Bergen, Norway. April 24-28, Paper No. 30, 8 pp.

[11] Dongarra, J.J., J.R. Bunch, C.B. Moler, and G.W. Stewart (1979). *LINPACK Users' Guide.* SIAM Publications, Philadelphia, USA.

[12] Teunissen, P.J.G., P.J. de Jonge, and C.C.J.M. Tiberius (1995). A new way to fix carrier-phase ambiguities. *GPS World*, Vol. 6, No. 4, April 1995, pp. 58-61.

[13] Teunissen, P.J.G. (1996). GPS carrier phase ambiguity fixing concepts. Chapter 8 of Lecture Notes in Earth Sciences, No. 60, *GPS for Geodesy,* International School, Delft, The Netherlands, March 26-April 1, 1995, A. Kleusberg and P.J.G. Teunissen, (Eds.) Springer Verlag.

[14] Apostol, T.M. (1969). *Multi-variable calculus and linear algebra, with applications to differential equations and probability.* Calculus Vol. 2, Second edition, Wiley, New York.

[15] Jonge de, P.J., and C.C.J.M. Tiberius (1996). Integer ambiguity estimation with the LAMBDA method. *Proceedings IAG Symposium No. 115, GPS trends in precise terrestrial, airborne and spaceborne applications,* XXI General Assembly of IUGG, July 2-14, Boulder, CO, July 2-14, 1995, G. Beutler et.al., (Eds.), Springer Verlag, pp. 280-284.

[16] Teunissen, P.J.G., P.J. de Jonge, and C.C.J.M Tiberius (1995). The least-squares ambiguity decorrelation adjustment: its performance on short GPS baselines, submitted to *Journal of Geodesy.*

[17] Marel, van der, H. (1989). Algorithmic aspects of solving large systems of equations (1989). *Proceedings of ISPRS intercommission working group III/IV, Tutorial on mathematical aspects of data analysis,* Pisa, Italy, June 1-2, pp. 49-103.

[18] Jonge de P.J., C.C.J.M. Tiberius, and P.J.G. Teunissen (1996). Computational aspects of the LAMBDA method for GPS ambiguity resolution. Poster to be presented at *ION GPS-96, 9th International Technical Meeting of the Satellite Division of the Institute of Navigation,* Kansas City, Missouri, Sept. 17-20.

[19] Teunissen, P.J.G., P.J. de Jonge, and C.C.J.M. Tiberius (1996). The volume of the GPS ambiguity search space and its relevance for integer ambiguity resolution. Paper to be presented at *ION GPS-96, 9th International Technical Meeting of the Satellite Division of the Institute of Navigation,* Kansas City, Missouri, Sept. 17-20.

LAMBDA bibliography

Jonge de, P.J., and C.C.J.M. Tiberius (1994). A new GPS ambiguity estimation method based on integer least squares. *Proceedings of the 3rd International Symposium on Differential Satellite Navigation Systems DSNS'94,* London, UK, April 18-22, Paper No. 73, 9 pp.

Jonge de, P.J., and C.C.J.M. Tiberius (1996). Integer ambiguity estimation with the LAMBDA method. *Proceedings IAG Symposium No. 115, GPS trends in precise terrestrial, airborne and spaceborne applications,* XXI General Assembly of IUGG, July 2-14, Boulder, CO, July 2-14, 1995, G. Beutler et.al., (Eds.), Springer Verlag, pp. 280-284.

Jonge de, P.J., and C.C.J.M. Tiberius (1996). The LAMBDA method for integer ambiguity estimation: implementation aspects. *Delft Geodetic Computing Centre LGR series,* No. 12, (this report)

Jonge de P.J., C.C.J.M. Tiberius, and P.J.G. Teunissen (1996). Computational aspects of the LAMBDA method for GPS ambiguity resolution. Paper to be presented at *ION GPS-96, 9th International Technical Meeting of the Satellite Division of the Institute of Navigation,* Kansas City, Missouri, Sept. 17-20.

Teunissen, P.J.G. (1993). *Least-squares estimation of the integer GPS ambiguities.* Invited lecture, Section IV Theory and Methodology, IAG General Meeting, Beijing, China, August, also in *Delft Geodetic Computing Centre LGR series,* No. 6, 16 pp.

Teunissen, P.J.G. (1994). A new method for fast carrier phase ambiguity estimation. *Proceedings IEEE Position, Location and Navigation Symposium PLANS'94,* Las Vegas, NV, April 11-15, pp. 562-573.

Teunissen, P.J.G. (1995). On the GPS double difference ambiguities and their partial search spaces. *Proceedings IAG Symposium No. 114, Geodetic Theory Today,* Springer Verlag, pp. 39-48, III Hotine-Marussi symposium on Mathematical Geodesy, L'Aquila, Italy, May 29-June 3, 1994.

Teunissen, P.J.G. (1995). The invertible GPS ambiguity transformations. *Manuscripta Geodaetica,* Vol. 20, No. 6, pp. 489-497.

Teunissen, P.J.G. (1995). The least-squares ambiguity decorrelation adjustment: a method for fast GPS integer ambiguity estimation. *Journal of Geodesy,* Vol. 70, No. 1-2, pp. 65-82.

Teunissen, P.J.G. (1996). GPS carrier phase ambiguity fixing concepts. Chapter 8 of Lecture Notes in Earth Sciences, No. 60, *GPS for Geodesy,* International School, Delft, The Netherlands, March 26-April 1, 1995, A. Kleusberg and P.J.G. Teunissen, (Eds.) Springer Verlag.

Teunissen, P.J.G. (1996). Size and shape of L1/L2 ambiguity search space. *Proceedings IAG Symposium No. 115, GPS trends in precise terrestrial, airborne and spaceborne applications,* XXI General Assembly of IUGG, Boulder, CO, July 2-14, 1995, G. Beutler et.el., (Eds.), Springer Verlag, pp. 275-279.

Teunissen, P.J.G. (1996). Rank defect integer least-squares with applications to GPS. *Bolletino di Geodesia e Scienze Affini,* Vol. 55, No. 3, pp. 225-238.

Teunissen, P.J.G. (1996). An analytical study of ambiguity decorrelation using dual frequency code and carrier phase. *Journal of Geodesy,* Vol. 70, No. 8, pp. 515-528.

Teunissen, P.J.G. (1996). On the geometry of the ambiguity search space with and without ionosphere. *Zeitschrift für Vermessungswesen,* Vol. 121, No. 7, pp. 332-341.

Teunissen, P.J.G., P.J. de Jonge, and C.C.J.M. Tiberius (1994). On the spectrum of the GPS DD-ambiguities. *Proceedings of ION GPS-94, 7th International Technical Meeting of the Satellite Division of the Institute of Navigation,* Salt Lake City, UT, September 20-23, pp. 115-124.

Teunissen, P.J.G., P.J. de Jonge, and C.C.J.M. Tiberius (1995). A new way to fix carrier-phase ambiguities. *GPS World,* Vol. 6, No. 4, pp. 58-61.

Teunissen, P.J.G., P.J. de Jonge, and C.C.J.M. Tiberius (1995). The LAMBDA method for fast GPS surveying. *Proceedings of International Symposium 'GPS technology applications',* Bucharest, Rumania, September 26-29, pp. 203-210.

Teunissen, P.J.G., P.J. de Jonge, and C.C.J.M Tiberius (1995). The least-squares ambiguity decorrelation adjustment: its performance on short GPS baselines, submitted to *Journal of Geodesy.*

Teunissen, P.J.G., P.J. de Jonge, and C.CJ.M. Tiberius (1996). The volume of the GPS ambiguity search space and its relevance for integer ambiguity resolution. Paper to be presented at *ION GPS-96, 9th International Technical Meeting of the Satellite Division of the Institute of Navigation,* Kansas City, Missouri, Sept. 17-20.

Teunissen, P.J.G., and C.C.J.M. Tiberius (1994). Integer least-squares estimation of the GPS phase ambiguities. *Proceedings of International Symposium on Kinematic Systems in Geodesy, Geomatics and Navigation KIS'94,* Banff, Canada, August 30-September 2, pp. 221-231.

Tiberius, C.C.J.M., and P.J. de Jonge (1995). Fast positioning using the LAMBDA-method. *Proceedings of the 4th International Symposium on Differential Satellite Navigation Systems DSNS'95,* Bergen, Norway, April 24-28, Paper 30, 8 pp.

Tiberius, C.C.J.M., and P.J. de Jonge (1995). Introduction to GPS Surveying (Part 4). *GIM International Journal for Geomatics,* Vol. 9, No. 10, pp. 61-67.