The Dissertation Committee for Mark Brandon Hinga
certifies that this is the approved version of the following dissertation:

Using Parallel Computation to apply the Singular Value Decomposition (SVD) in solving for large Earth Gravity Fields based on Satellite Data

Committee:

Byron D. Tapley, Supervisor

Wallace T. Fowler

Glenn Lightsey

Robert van de Geijn

Steven Poole

Using Parallel Computation to apply the Singular Value Decomposition (SVD) in solving for large Earth Gravity Fields based on Satellite Data

by

Mark Brandon Hinga, B.S.,B.A.,M.S.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2004

Dedicated to my Mother

Acknowledgments

I would like to thank the people who supported me during my six years as a graduate student at the University of Texas at Austin, Center for Space Research. To Professor Tapley, I am grateful for the financial support, advising, class room instruction, use of facilities and the privilege to participate in the scientific investigations afforded by his successful GRACE project. To Professor Fowler, I am grateful for the opportunity to have taken part in the launch and orbit insertion activities of the GRACE satellites as well as benefitting from his advice and class room instruction. To Prof. van de Geijn, I am grateful for his innovative parallel computing infrastructure PLAPACK, without which, this dissertation would have simply been impossible. Furthermore, I would like to thank him for the many sessions of linear algebra instruction and parallel computing techniques, both in and out of the classroom. To Dr. Poole, I am very thankful for all of his guidance, expertise and support, starting from the selection of the dissertation topic to its successful completion. I would also like to thank Dr. Ries for assisting me with various scientific questions I posed throughout the years and for the opportunity to work with PRARE data to produce ERS-2 satellite orbits. Also, I would like to convey my sincere appreciation to the staff at CSR, especially Karen Burns and Shauna Pendleton for making everything run so smoothly for graduate students, both on and off campus.

Last, but definitely not least, I give great thanks to my Mother, who has

given me wonderful moral and spiritual support, not just during my time at Texas, but for my whole life. Without her love, dedication and sacrifice, I would not have made it this far.

Using Parallel Computation to apply the Singular Value Decomposition (SVD) in solving for large Earth Gravity Fields based on Satellite Data

Publication No. _____

Mark Brandon Hinga, Ph.D.
The University of Texas at Austin, 2004

Supervisor: Byron D. Tapley

Using satellite data only to estimate for an Earth gravity field introduces the problem of an ill-conditioned system of equations. This mathematical difficulty amplifies as the number of unknown gravity field parameters increases, requiring a stabilization of the inversion for solution. But the number of parameters to be estimated can also be too large to allow inversion using a sequential algorithm (one computer processor). Therefore the challenge is two-fold. A stabilized inversion must be performed with a parallel (multi-processor) algorithm.

Thus, new code was developed in the parallel computing infrastructure of Parallel Linear Algebra Package (PLAPACK) to achieve the task of applying the Singular Value Decomposition (SVD) to invert for (and stabilize) very large gravity fields of well over 25,000 unknown parameters. This new code is given the name (**P**arallel **LA**rge **S**vd **S**olver) **PLASS**.

The choice of the SVD was made because it offers multiple opportunities of stabilization techniques. Poorly observed parameter corrections are removed from the culpable eigenspace of the normal matrix of CHAMP or the singular vector space of the upper R triangular matrix of GRACE. Solutions were stabilized based on the removal of either eigenvalues or singular values using four different standard optimization criteria: Inspection, Relative Error, Norm Norm minimization, trace of the Mean Square Error (MSE) matrix, and with a fifth method, independently introduced for this investigation, that optimizes removal of eigenvalues or singular values based on Kaula's power rule of thumb. This method is given the name "Kaula Eigenvalue (KEV) or Kaula Singular Value (KSV) relation". For the gravity fields of this investigation, orbital fits, geodetic evaluations and error propagations of the best of the resulting SVD gravity fields were performed, and shown to be comparable to the CHAMP solution obtained by the GeoForschungsZentrum (GFZ) and to the full rank GRACE solution obtained by the Center for Space Research (CSR).

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Gravity Field - Historical Background

The problem of estimating the Earth's gravity field by tracking artificial satellites began almost a half century ago with the launch of the Soviet Satellite "Sputnik" in the fall of 1957. Since then, there have been steady advancements in the techniques of gravity field determination and a steady increase in their solution size. Because the Earth is non-spherical and has a heterogenous mass distribution, its gravity field is non-uniform, thereby causing the motion of an orbiting satellite to be perturbed away from that described by relative two-body orbital motion (Vallado [1]), expressed in the following equation of motion as

$$\overrightarrow{r}'' = -\frac{\mu}{r^2}\frac{\overrightarrow{r}}{r}, \tag{1.1}$$

where μ is the gravitational constant and r is the relative barycentric distance between the two bodies of point mass or uniform spherical mass distribution. An estimated gravity solution is commonly a synonym for a set of parameters that are the coefficients of the orthogonal basis functions in a model that approximates the shape of the Earth's gravity field. These coefficients are often those of the Legendre polynomial and trigonometric functions in a spherical harmonic expansion,

an equation that describes a three dimensional gravitational surface potential in the free space (zero density) above the Earth.

Data from which a gravity field is derived comes from two main sources, terrestrial measurements (taken on the Earth's surface) and that from the observed motion of satellites. The number of satellites launched into orbit for gravity field developments stood at more than 50 in the year 1964 (Vetter, et al. [2]). Gravity models of note produced in that time period are those of the Johns Hopkins University/Applied Physics Laboratory (JHU/APL) and the Smithsonian Astrophysical Observatory (SAO). Gravity Field JHU/APL 1.0, developed in 1963, was a spherical harmonic model (seen in Equation 1.13) which contained terms up to degree and order eight (Vetter, et al. [2]). Later, model JHU/APL 4.5 was complete up to degree and order 15 by 1967. In 1966, the SAO produced Standard Earth 1, using Baker-Nunn data, out to degree and order eight.

Since the mid 1960's, the sizes of estimated gravity fields have increased due to improvements in observation technologies, increase in computational power, and the demanding accuracy requirements of scientific satellites. For example, the required radial orbital error budget due to gravity model error of the Topex/Poseidon mission (Stewart et al. [3]) was specified to be no more than ± 10 cm.

In 1972, using only Baker-Nunn data, the NASA/GSFC (Goddard Space Flight Center) developed its first gravity model with the name Goddard Earth Model (GEM). It is complete out to degree and order 12. Later models, GEM 1 through GEM 10, were published and were derived from the sources of laser range, Doppler range rate, satellite radar altimetry and surface gravity data. The GEM 10, finished

in 1982, is complete to degree and order 36, followed ten years later by the model GEM-T3, complete to degree and order 50. In preparation for the Topex/Poseidon mission, this model was then followed up in a joint effort between NASA/GSFC and the University of Texas at Austin/Center for Space Research UT/CSR, and labelled the Joint Gravity Model (JGM), which models the gravity field up to degree and order 70.

The Center for Space Research (CSR) has independently developed a number of gravity field models with the designation Texas Earth Gravity (TEG), using a variety of different observations (i.e. satellite laser range (SLR), radar altimetry, Doppler range rate, surface gravity). TEG-1 (degree and order 36) and TEG-2 (degree and order 50) were published in 1988 and 1990, respectively. TEG-3 (degree and order 70) contains data of similar nature but also includes that of Global Positioning System (GPS) data from the onboard GPS receiver of Topex/Poseidon and surface gravity data from the Ohio State University (OSU) gravity model OSU91A. Currently, CSR is conducting the Gravity Recovery and Climate Experiment (GRACE) twin satellite mission to develop a gravity field complete to degree and order 360. The full set of data types may or may not encompass all those aforementioned, but will certainly contain GPS observations and the relative microwave range and range rate measurements recorded between the spacecraft pair. The next section explains the difficulties caused by using only satellite tracking data to obtain a gravity field of such a high degree and order.

1.2 Observability Problem

Because the attenuation of the high degree and order harmonics with altitude is rapid, these parameters have but a very slight effect on a satellite's orbit. Thus the satellite tracking data will observe only very small effects which are vulnerable to unwanted signals from noise or other undesirable sources. According to Newton's law for gravitational force, gravity fields decay with the inverse of the square of the distance from their sources. This decay is dependant on wavelength, so that for any given distance from the Earth surface, short-wavelength (high degree and order harmonics) anomalies are attenuated more strongly than long-wavelength (low degree and order harmonics) anomalies (NRC [4]). The inability of the observations to capture the short-wavelength anomalies is referred to as an observability problem. To alleviate this situation, the satellite must fly closer to the Earth. However at lower altitudes there are larger surface force effects, such as atmospheric drag which is non-conservative and is very difficult to model precisely. Devices such as the accelerometers inside the GRACE satellites are used to sense the non-gravitational forces at low-altitude, so they can be removed from the observations to reveal weaker gravity signals. Furthermore, surface gravity data, which are more sensitive to short-wavelength anomalies, may also be added into the observations to help allay both the problems of remaining gravity signal obfuscation and orbit insensitivities. To combine satellite tracking data with surface gravity data, UT/CSR uses the serial computer code entitled Large Linear System Solver (LLISS), or the parallel code called Aesop (Gunter [5]), both of which employ a technique that optimally assigns weights to observation data sets (Yuan [6]). If these methods do

not completely avoid or eliminate observability problems, one is forced to solve an ill-conditioned system of equations. The next section discusses what is meant by "ill-conditioned" and describes some common techniques of solving such systems.

1.2.1 The Ill-Conditioned System

A system of linear equations is considered to be ill-conditioned when it does not contain enough information to "observe" all of the parameters that one is attempting to measure. In other words, if such a system is not observing all of the intended parameters, then it can not resolve every one of them into an unambiguous value during the final inversion for solution. Indeed a solution can exist for an ill-conditioned system, however it may be nonsensical.

A system of linear equations can be represented by the matrix vector relation $A\mathbf{x} = \mathbf{b}$, where A is a matrix and **x** and **b** are vectors of appropriate dimension. If A does not contain enough information to support a solution of all its unknowns (parameters in the solution vector **x**), matrix A is regarded as ill-conditioned. It may also be classified as *singular* or *non-invertible*. This means that the solution,

$$\mathbf{x} = A^{-1}\mathbf{b}, \tag{1.2}$$

either does not exist (i.e. ∞ appears) or there are infinitely many solutions. However, if matrix A is *non-singular*, then the unique solution does exist and is given by Equation 1.2. But suppose we add a small perturbation to the system in the form of $A\mathbf{x} = \mathbf{b} + \delta\mathbf{b}$. This system also has a unique solution $\tilde{\mathbf{x}}$, that is hopefully not too far from **x**, meaning that $\tilde{\mathbf{x}} - \mathbf{x} = \delta\mathbf{x} =$ small. A small perturbation also implies

that the terms $\delta\mathbf{b}$ and $\delta\mathbf{x}$ are also small compared to $\mathbf{b}$ and $\mathbf{x}$, respectively. To define "small" and to quantify the size of vectors, the vector norm $\|\cdot\|$ is used. Thus the size of $\delta\mathbf{x}$ relative to $\mathbf{x}$ is given by $\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|}$ and the size of $\delta\mathbf{b}$ relative to $\mathbf{b}$ is given by $\frac{\|\delta\mathbf{b}\|}{\|\mathbf{b}\|}$. It would be fortunate that when $\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|}$ is small, $\frac{\|\delta\mathbf{b}\|}{\|\mathbf{b}\|}$ is also small. Substituting the perturbations $\mathbf{x}$ and $\mathbf{b}$ into the equation $A\mathbf{x} = \mathbf{b}$ gives $A(\mathbf{x} + \delta\mathbf{x}) = (\mathbf{b} + \delta\mathbf{b})$, which suggests that $A\delta\mathbf{x} = \delta\mathbf{b}$ and yields $\delta\mathbf{x} = A^{-1}\delta\mathbf{b}$.

Postulate 1. *Suppose, using any vector norm definition and invoking the induced matrix norm, this equation implies that*

$$\|\delta\mathbf{x}\| \leq \|A^{-1}\|\|\delta\mathbf{b}\|. \tag{1.3}$$

By writing the equation $A\boldsymbol{x} = \boldsymbol{b}$ *as* $\boldsymbol{b} = A\boldsymbol{x}$*, we can get the inequality* $\|\mathbf{b}\| \leq \|A\|\|\mathbf{x}\|$ *to obtain*

$$\frac{1}{\|\mathbf{x}\|} \leq \|A\|\frac{1}{\|\mathbf{b}\|}. \tag{1.4}$$

If we multiply Equations 1.3 *and* 1.4 *together, we arrive at the important inequality*

$$\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \|A\|\|A^{-1}\|\frac{\|\delta\mathbf{b}\|}{\|\mathbf{b}\|}, \tag{1.5}$$

which provides a bound for $\|\delta\mathbf{x}\|/\|\mathbf{x}\|$ *in terms of* $\|\delta\mathbf{b}\|/\|\mathbf{b}\|$*. The factor* $\|A\|\|A^{-1}\|$ *is called the condition number* κ (A). (Watkins [7], Demmel [8])

Looking at Equation 1.5 it can be seen that if κ (A) is not too "large", then small values of $\|\delta\mathbf{b}\|/\|\mathbf{b}\|$ lead to small values in $\|\delta\mathbf{x}\|/\|\mathbf{x}\|$. In other words if the system is not excessively sensitive to perturbations in $\mathbf{b}$ then matrix A is *well-conditioned*. If, on the other hand κ (A) is large, a small value of $\|\delta\mathbf{b}\|/\|\mathbf{b}\|$ might cause a large change in $\|\delta\mathbf{x}\|/\|\mathbf{x}\|$. Therefore the system and solution could be very

sensitive to small perturbations in **b** and we say that matrix A, or the system of linear equations, is *ill-conditioned* or has a *singularity* problem.

1.2.1.1 Common Methods of Solution

In such situations we must resort to using special methods of solving for the solution vector **x** of Equation 1.2. A common technique is one which uses the *L-curve criterion* to optimize a regularization procedure, that assumes error exists in both matrix A and in vector **b**. This approach tries to find the solution that achieves a balance between the error introduced by the regularization and the error caused by the data itself (Hansen [9][10] and Hansen and O'Leary [11]). Regularization is a process by which an ill-conditioned system of equations is approximated, or replaced, by another system that is less ill-conditioned. The following equations illustrate the regularization (or stabilization) concept. For our ill-conditioned system $A\mathbf{x} = \mathbf{b}$, let

$$A \in \Re^{mxn}, \mathbf{x} \in \Re^n, \mathbf{b} \in \Re^m \quad \text{and } m > n, \tag{1.6}$$

where m and n are the number of observations and parameters, respectively. The regularization arises from the introduction of the operator L, a diagonal matrix, and the replacement of the linear system with the minimization problem,

$$\min_{\mathbf{x} \in \Re^n} \left\{ \| A\mathbf{x} - \mathbf{b} \|^2 + \alpha \| L\mathbf{x} \|^2 \right\}. \tag{1.7}$$

(where α is the so called *regularization parameter*) that ensures a search within a subspace that locates a nearby acceptable solution. When Equation 1.7 is expressed in the form,

$$(A^T A - \alpha L^T L) x_\alpha = A^T b \tag{1.8}$$

it is known as the Tikhonov-Regularization (Ehrig and Nowak [12]) and provides a best fit solution. For gravity field solutions, choice of the elements in the diagonal matrix L can be related to Kaula's power rule, yielding a so-called Kaula-Matrix (Reigber [13]). This rule is based on empirical (historical) gravity field measurements and provides an acceptable limit, at each harmonic degree, for the total "power" of the estimated geo-potential coefficients in Equation 1.13. A particular power at an harmonic degree is calculated explicitly by the degree variance Equation 4.45. Kaula's power rule, seen in Equation 4.42, gives the expected limit.

The method of Generalized cross-validation (GCV) (Golub, Heath and Wahba [14]) introduces a functional G of α, namely

$$G(\alpha) = \frac{\|b - Ax_\alpha\|^2}{\left(\text{tr}\left[I_m - A(A^TA + \alpha L^TL)^{-1}A^T\right]\right)^2}, \tag{1.9}$$

to find the optimum regularization parameter α that produces the best solution. The method of *Quasioptimality* is based on the assumption that the function $\|x_\alpha - x^*\|$, where x^* is the exact solution of the unperturbed problem, contains a minimum at the optimum α (Ehrig [12]), and is found by minimizing the following equation,

$$\Psi(\alpha) = \|\alpha \frac{d}{d\alpha} x_\alpha\|^2. \tag{1.10}$$

Hanke and Raus [15] developed heuristic rules for the determination of α for different regularization techniques. Specific to the Tikhonov-Regularization approach, the function of α to be minimized is given as,

$$\varphi(\alpha) = \sqrt{1 + \frac{1}{\alpha}}\sqrt{(b - Ax_{\alpha_1})^T(b - Ax_{\alpha_2})}, \tag{1.11}$$

where

$$x_{\alpha_2} = x_{\alpha_1} - \alpha \frac{d}{d\alpha} x_\alpha \Big|_{\alpha=\alpha_1}. \tag{1.12}$$

The method of Generalized-maximum-Likelihood, a specific case of the GCV technique mentioned above (Equation 1.9), is discussed by Neumaier [16] and presented also as a functional in terms of α and is recommended to the curious reader.

The method of Mean Square Error Analysis (MSE) as a means of finding the optimum α is examined by Bouman et al. [17],[18],[19],[20], [21],[22]. The MSE is implemented in this investigation, but in such a manner as to find the optimum number of singular values or eigenvalues which produces the best gravity field solution. This technique is discussed in Chapter 4.

Another option for determining the geopotential model when the system of equations is ill-conditioned involves the application of Householder QR factorizations with column pivoting to matrix A to produce an orthonormal basis (Golub [23]) of lesser rank r, necessary for solution (where r = rank(A) $<$ n). This algorithm also determines the numerical rank of A.

1.2.2 Solutions of Ill-Conditioned Systems from the SVD

The Singular Value Decomposition (SVD) can be used to find the rank of a given matrix to determine if it is invertible in the ordinary sense. If the matrix is rank deficient (ill-conditioned), its inversion is possible by using fewer singular values than the number of parameters that are being estimated. Removing singular values to allow for a stable pseudo-inversion (SVD inversion) does not affect the parameter space. That is to say, a full set of parameters comes out of the solution

process regardless of the number of removed singular values. Thus, the problem of determining how many singular values to keep in an SVD inversion is important, as it effects solution quality.

Several methods have been investigated for finding the best number of singular values for ill-conditioned or near ill-conditioned problems. Lawson [24] suggested a method that chooses the number of singular values that corresponds to the minimum of the sum of two quantities. Those quantities are the square root of the sum of the squares of the elements in the solution vector, the norm of the solution vector, and likewise for the norm of the residual vector. If both norms are plotted in a graph of the norms of the residuals versus the norms of the solutions, this minimum occurs at the point on the curve with the smallest distance to the origin. This is referred to as a norm-norm plot and is described in Section 4.1.4.3. This technique was used by McCord [25] in the solution of linearized spacecraft navigation problems.

An example of a realtime satellite SVD application is that of cycle ambiguity resolution necessary for spacecraft attitude determination. The SVD can be used to measure the observability of a given system of available measurements by computing in real time the condition number of an observation matrix (Lightsey [26]). As the observability improves, the condition number lowers to a value determined by the Global Positioning System (GPS) signal line of sight and measurement geometry over the complete data collection. In other words, the condition number may be used as a logical input for a feedback loop to select the best sample period and collection time for the Quasi-Static integer resolution algorithm (Lightsey

[26]).

1.2.2.1 SVD Gravity Fields

Cicci [27] developed a method of inverting ill-conditioned matrices by using a Ridge-regression approach, whose equation formulation is very similar to that seen in the Tikhonov technique, Equation 1.8. This method adds *a priori* values to the diagonal of the system's matrix to avoid dividing by zero during inversion. Cicci demonstrated his technique in the simulation of a degree and order 10 gravity field solution and in 1994 [28] used it to determine the gravity field of Venus to a degree and order of 22. In 1993 Lerch et al. [29] used Eigenvalue Decomposition (EVD), which is a subset of the SVD method, to address the problems of singularity in determination of gravity fields. Using only satellite tracking data, the ill-conditioned system of equations were inverted via the EVD to solve for a gravity solution of degree and order 36. Very similar to the SVD technique, the EVD method decomposes the normal matrix to find the eigenvalues which produce the best gravity solution. Determining exactly which eigenvalues to remove was based on the minimum trace of the Mean Square Error (MSE) matrix. This trace is the summation of two terms. One term, referred to as the noise, is the sum of the squares of the reciprocals of the eigenvalues, the other is the sum of the squares of the variances, each approximated by Kaula's rule, and represents the bias. As eigenvalues are removed, the noise term gets smaller, while the the bias term gets larger. Choosing the number of eigenvalues which minimizes the sum of the two, yields the best gravity solution. The total number of parameters estimated by Lerch

in his investigation was less than 2000 and yielded comparable orbit fit residuals with those of gravity field GEM-T2. In 1996 Ahn [30] used the SVD to estimate larger gravity fields of degree and order 70 (which contain about 5000 parameters) and performed similar solution quality tests.

As the size of the gravity field to be estimated increases (as the number of geo-potential coefficients to be estimated increases) naïvely implementing the SVD quickly becomes an impossibility. In gravity field solutions, the number of observations incorporated into the so-called information matrix (regress-file) can number into the millions, and direct application of the SVD to the raw observations, would require prohibitive amounts of computer time and memory. Commonly, the SVD is carried out in such a way that requires three times the memory of the given input matrix (IMSL [31] and Dongarra [32]) that is needed for the left and right singular vectors and the original input matrix. In this study, memory requirement obstacles are eliminated because of three reasons. First, in the cases of the square CHAMP Normal matrix (dimension of 11216) and of the rectangular GRACE information matrix (dimension of 8309302 by 25917), the left singular vectors are never accumulated. Second, the GRACE information matrix is upper-triangularized (orthogonalized) into square matrix R using the software entitled "aesop" (Gunter [5], [33]) (dimension 25917), see Section 4.1.2, before the SVD is applied. Third, if additional computer core memory is needed, the (**P**arallel **LA**rge **S**vd **S**olver) **PLASS** tool can use more processors. These techniques are standard.

As the degree and order of the harmonics in an estimated gravity field model increases, the chance of unobservability in these higher coefficients gets larger.

Thus, the estimated values of these parameters will become inflated due to small diagonal elements in either the normal matrix or in the R matrix, all of which are created by the observation partials that deliver little information. The singular values obtained from the SVD of such a matrix provide insight to these diagonals and the matrix's singularity. If there are some singular values which are zero (near machine zero) or they are much smaller than other singular values, then the system (matrix) is considered ill-conditioned (singular), because the inversion is impossible in an ordinary sense. In other words, to obtain solution, the reciprocal of these singular values must be taken. If any of them are zero or near zero, then infinity or near infinity values are inserted into the projection of the singular vector subspaces onto the solution space. By simply **not** taking these reciprocals and setting them to zero during the solution process, these singular values are removed along with their associated vectors in U and V^T (the matrices containing the left and right singular vector spaces, respectively). Thus, the singularity of any given ill-conditioned system (information matrix) can be removed. In fact, no matter the severity of the ill-condition, a solution is always possible when employing the SVD (Press et al. [34]). In the case of gravity field solutions, the SVD enables the removal of harmful influences from unobserved parameters, which manifest themselves as unacceptable or inflated values in an estimation.

The solution obtained from the pseudo inverse using the SVD is therefore unique, because each case of either retained or removed singular values, produces a different estimate. The number of candidate SVD solutions can be as large as the number of singular values to remove or keep. Often the distribution of singular

values is smooth with no distinctive gap between any of them. This may be discouraging as it would seem difficult to choose a "correct" solution. However, this non-uniqueness can be an advantage. A good candidate solution can still be found among the many, whereas in the case of the full rank solution of an ill-conditioned system, where all singular values are retained, only one meaningless solution is available. Thus the question ensues, how does one choose the singular values? Section 4.1.4 discusses the techniques used to select singular values in an SVD solution.

1.2.3 Parallel Implementations of the SVD

Because the SVD is one of the most fundamental and important matrix decompositions of linear algebra (Reilly [35]), which is used in a wide range of applications to solve ill-conditioned systems of equations, there are a fair number of parallel SVD algorithms that have been developed to be applied in a multi-processor environment. Since the most time consuming phase of the SVD is the convergence from the bilinear to the diagonal form (if singular vectors are accumulated), many parallel SVD methods are focused on this stage. Transforming the bidiagonal matrix to a diagonal matrix (the elements of which are the singular values of the original system of equations), involves the application of a series of two-dimensional plane rotations (discussed in Section 5.2.2) also known as Givens plane rotations.

Because of the inherent parallelism in Givens rotations (discussed in Section 5.2.2.1) and their high cost during the diagonalization stage of the SVD, there has been a renewed interest in finding an efficient parallel SVD algorithm (Dem-

mel [8]). Givens rotations can be applied in parallel to disjoint pairs of rows and/or columns of a target matrix, so a matrix with n rows and/or columns can have [n/2] Givens rotations applied simultaneously (Luk [36]). In this investigation, through the use of the **multi-vector object** of the PLAPACK (Parallel Linear Algebra Package) infrastructure (van de Geijn [37]), the rotations are applied to **all** columns of a matrix so that $\frac{\frac{n^2}{2}}{\text{number of processors}}$ of them can be performed concurrently. The other characteristic of the multi-vector object, specific in the application of the Givens rotations, is that **no** communication or synchronization among processors is required during the convergence to singular values. These facts enable the benefits of parallel processing without the cost of communication between processors during convergence to singular values, see Section 5.3.2.

The approach by Braun et al. [38], presents a different method in which one or two columns of a matrix are updated on a single processor, requiring a high level of frequency and complexity of inter-processor communication. That technique was applied to very small problem sizes, i.e. 6 x 7 matrix, and would be problematic if applied to systems containing thousands of unknowns. A technique which assumes that the processors are arranged in a ring, is given by Berry et al. [39] where multiple columns of the matrix are distributed in circular fashion. Pan and Hamdi [40] discuss the SVD on a two-dimensional array of processors of a pipelined bus system, in which messages can be transmitted in a pipelined fashion where each processor is limited to the sending and receiving of one message per bus cycle. Both methods of Berry and Pan require communication among processors during the application of a Givens rotation. Jessup and Sorensen [41] present a conquer

and divide approach on an Alliant FX/8 machine (2-dim array), where the bidiagonalized matrix is split recursively into submatrices of order 8 and uses a serial SVD method on each processor. This recursive matrix splitting leads to a hierarchy of subproblems with a data dependency graph in the form of a binary tree of height called "h" (Jessup [41]). Somewhere inside this tree of dependencies, parallelism is achieved by dynamically assigning root-finding and singular vector computation tasks to processors (Jessup [42]). It is only necessary to understand this scheme to the extent that it is clear, that this method of converging to singular values during this Givens phase, requires communication and synchronization among processors. Since this process is of a rather fine grain algorithm (Demmel [8]), operating on pairs of columns, it cannot exploit higher level BLAS (Basic Linear Algebra Subprograms). Therefore, it is argued by Bischof and Shroff [43], [44], that one should employ block Givens algorithms, which work on more than two columns of the target matrix, allowing for efficient matrix-matrix operations. However, this argument quickly breaks down in CHAMP and GRACE size problems, when the block sizes of Givens rotations get larger than about 12 columns. The time spent accumulating block Givens rotation matrices is not recovered during their subsequent application to the target matrix during the convergence phase. It is the opinion of this author, that Givens rotation matrix sizes should be kept at the size of 2x2. The physical arrangement of the computer processors in this study are linear, but they are logically viewed in a two dimensional form. The next section describes the purpose of the parallel SVD algorithm developed for this investigation.

1.3 Purpose of this Investigation

The Singular Value Decomposition (SVD) investigation, this study, is conducted to calculate gravity fields complete to degree and order 100 and 160, derived from satellite data only, namely GPS and/or microwave measurements. The primary motive for applying the SVD in this examination is to calculate gravity fields that are comparable to or an improvement to full rank solutions, despite observability problems that can exist from using satellite observations only. The second objective, is to demonstrate the practicality of using the **PLASS** SVD tool to solve for very large gravity fields, showing that it is a feasible method of inversion and stabilization. The next section summarizes how the goals of this SVD project were met.

1.3.1 Goals of this Study - New SVD Tool "PLASS"

The goal of this study was to develop new parallel code in the infrastructure of PLAPACK to apply the SVD as a parallel algorithm to solve for very large gravity fields. The software product is given the name "**P**arallel **LA**rge **S**vd **S**olver" and is abbreviated **PLASS**. (Other PLAPACK code, "aesop", was already developed to solve for very large gravity fields implementing the **QR method** (Gunter) [5]. The "aesop" code generates the upper triangular matrix R which is used as the input for PLASS) **PLASS** is required also to determine the singular values which produce the best gravity field for a given input. Four standard techniques of choosing singular values to invert a system of equations, were implemented. Also, the Kaula-Singular Value (KSV) method, is introduced independently and included into this study to

relate Kaula's rule, specific to Earth gravity solutions, to a gravity field produced by the SVD.

To define what is meant by "very large gravity field" the following explanation is given. In this investigation, a gravity field is modelled using the spherical harmonic expansion of Equation 1.13. This expression describes a three dimensional gravitational potential, U, in the free space (zero density) above the Earth (Tapley, Born, Schutz [45]),

$$\begin{aligned} U &= \frac{GM}{r} + U' \\ U' &= -\frac{GM^*}{r}\sum_{l=1}^{\infty}\left(\frac{a_e}{r}\right)^l P_l(\sin\phi)J_l \\ &+ \frac{GM^*}{r}\sum_{l=1}^{\infty}\sum_{m=1}^{l}\left(\frac{a_e}{r}\right)^l P_{l,m}(\sin\phi)[C_{l,m}\cos m\lambda + S_{l,m}\sin m\lambda], \end{aligned} \quad (1.13)$$

where mass distribution is expressed in the spherical coordinates (r, ϕ, λ), with ϕ and λ representing geocentric latitude and longitude, respectively. The scale factors $M^\star$ and reference distance a_e nondimensionalize the mass property coefficients $C_{l,m}$ and $S_{l,m}$. The term $P_{l,m}$ is Legendre's Associated Function of degree ℓ and order m. If the values of ℓ and m are assigned the integer 100, we say that the gravity field is of degree and order 100. Expanding the double summation in this equation with ℓ =m=100 and using it to derive the partials seen in Equation 3.3 of Section 3.1 (along with the observation-state partials of Equation 3.8), results in a system of equations of about 10000 unknowns. In the CHAMP case of this study, there are 11216 unknowns. In the GRACE case (very large gravity field) of this investigation, ℓ=m=160 (degree and order 160) there are exactly 25917 parameters

to be determined. Problem sizes of 5000 or less parameters are considered “small” gravity fields for the purposes of this analysis.

1.3.2 Outline of this Study

At this point of the document, the reader has finished an introduction to the problem of gravity field estimation and a brief historical background. Common solution approaches, their computational obstacles and what motivates the development of parallel SVD algorithms were also discussed.

The next chapter, Chapter 2, provides a brief outline of some of the “basics” of parallel computation regarding machine architecture, parallel performance theory and communication. The concept of Physically Based Matrix Distribution (PBMD) [37] and a simple example thereof is also provided to give the reader an idea of the philosophy behind the underlying algorithms of this dissertation.

Chapter 3 discusses the theory of precision orbit determination. Chapter 4 presents the Singular Value Decomposition (SVD) and how a solution is stabilized for an optimum estimate along with a discussion on the SVD bias. The subsequent chapter, Chapter 5, is included into this document to explain the steps and various techniques necessary to implement the parallel SVD algorithm of this study, but is only necessary for the curious reader. Chapter 6 contains the results of the experiments. Conclusions from this study and suggestions for further research are discussed in Chapter 7. Appendix A presents the algorithm of the validation test and how it proves that the algorithm inside PLASS is correct with each execution. Appendix B presents the scalability of PLASS and numerical issues of the SVD.

Appendix C compares the SVD results of this investigation to the standard gravity field Texas Earth Gravity 4 (TEG4).

Chapter 2

Parallel Computation

The use of parallel processing has become important in fields of numerical analysis where problem sizes are too large for sequential machines. Even though the technology of individual computer processors are such that their computation speed is constantly becoming faster, approximately doubling every 18 months, it is the limitation of memory sizes near a processor and speed of memory operations, which prevent them from handling massively large problems, or prohibit the application of certain mathematical algorithms. For example, applying the Singular Value Decomposition (SVD) to solve for large gravity fields on one computer processor is simply impossible, due to the algorithm's very intensive memory operation and excessive space requirements. Thus, new code was developed, based in the PLAPACK infrastructure (van de Geijn [37]) for this investigation. There are many special complications and issues, that make parallel programming very different from programming on one computer. The following sections outline the elements of parallel processing and briefly discuss the important aspects that give rise to the above issues and the solutions to deal with them.

2.1 Machine Architecture

Parallel computers have evolved from experimental contrivances to reliable computational resources. Through the years, a variety of systems have been developed to meet the ever increasing demand for more processing power. However, problems of portability in software, both for algorithms and processor communications, across architectures, has hindered the ability of parallel computation to compete with conventional serial programming. Fortunately, there are only two classes of systems that appear to be surviving the competition, specifically, shared and distributed memory architectures. The driving force behind these system philosophies is that processors must have the ability to communicate with each other in order to cooperatively complete a task.

2.1.1 Shared Memory

A shared memory system is one in which its processors (or process elements, PE's) can access a common memory. That is, any PE shares the same memory at the usual level of load and store operations (Gropp [46]). A schematic diagram of this system is shown in Figure 2.1. This configuration allows processors to communicate with each other through variables stored in a shared address space, however it must be assured that processors do not simultaneously access regions of memory in such a way that errors would occur. This makes referencing data stored in memory similar to traditional single-processor programs. Usually, building a system with a large number of PE's is difficult and expensive and one must allow some memory references to take longer than others (Gropp [46]). Examples

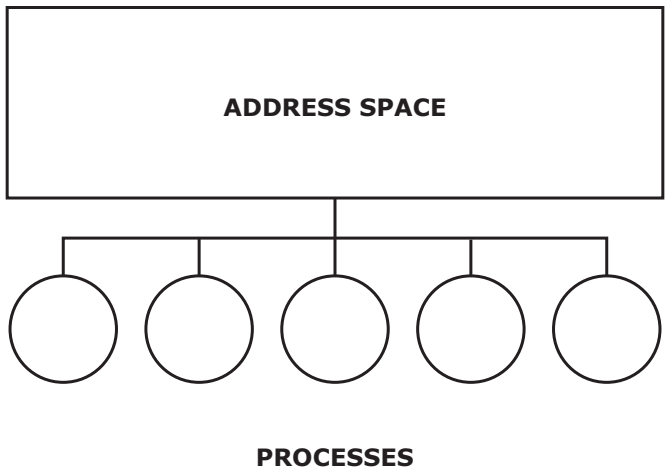


Figure 2.1: The shared-memory model

of such architectures are the SGI Origin 3000 series of computers, the SGI Power Challenge and Origin2000.

2.1.2 Distributed Memory

In distributed memory machines, each processor has its own independent memory. If one processor requires data contained in another processor's memory, messages must be passed between them, typically using a function library such as Message-Passing Interface (MPI), via a high-speed communications network. A schematic diagram of this system is shown in Figure 2.2. This system introduces the problem of how to distribute a computational task to multiple processors, each with distinct memory spaces, and to reassemble the results from each processor into one solution. Simple physical network arrangements such as rings, meshes or the torus, called topologies, are used for direct communication between processors to avoid the excess complexity that would arise if each processor were directly connected

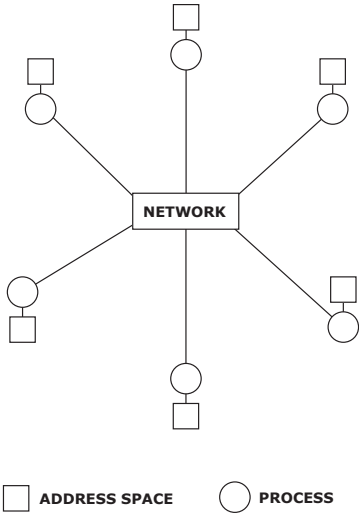


Figure 2.2: The distributed-memory model

to all other processors. Since each processor has its own memory, the problem of memory access conflicts is avoided and an arbitrarily large number of them may be employed. However problems of network delays and network congestion/conflicts must be considered in programming for a distributed memory system.

2.1.3 Classes of Parallel Machines

The classes of parallel machines may be characterized by the way instructions are carried out on the processors and the manner in which data is distributed. Parallel or concurrent operation can take on different forms within a computer system. By understanding the concept of the "stream" in the computation process, different kinds of parallelism may be illustrated. A "stream" is a sequence of objects, such as data, or of actions, such as instructions (Flynn [47]). Each stream is independent of the other and an element in a stream may consist of one or more

objects of data or instruction. Thus there are four combinations, which describe the most familiar parallel architectures (Flynn [47]):

1. SISD—single instruction, single data stream. This is the traditional uniprocessor (Figure2.3(a)).

2. SIMD—single instruction, multiple data stream. Single instruction is carried out on more than one piece of data. Also refers to a set of operations for efficiently handling large quantities of data in parallel. This includes vector processors as well as massively parallel processors. (Figure2.3(b)).

3. MISD—multiple instruction, single data stream. These are typically systolic arrays. (Or ”data parallel”) Many processing elements (functional units) perform the same operations on different data. There is often a central controller which broadcasts the instruction stream to all the processing elements. They operate with a global “heartbeat” and are no longer in use.(Figure2.3(c)).

4. MIMD—multiple instruction, multiple data stream. These computers are characterized by each processor working independently on separate data (typically executing the same program). This includes traditional multiprocessors as well as the newer work of networks of workstations. (Figure2.3(d)).

5. SPMD—single program multiple data stream (subset of MIMD). Separate processors running the same program synchronously or asynchronously using different sets of data.

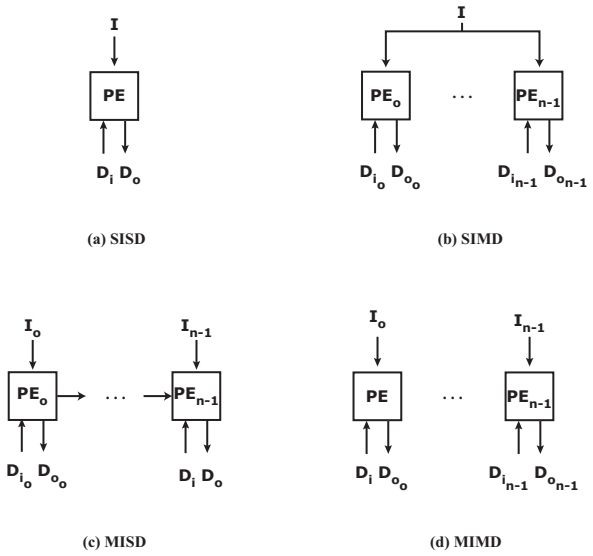


Figure 2.3: The stream model

2.2 Parallel Performance Theory

Using more processors and more memory in parallel to solve a computational task does not necessarily ensure better performance over the use of a single computer. Adding more horses to pull a heavy wagon may be helpful, but the work-efficiency on a per horse basis might decrease due to coordination problems or human-equine communication difficulties. Performance of parallel computing systems can be diminished due to memory access conflicts, notable in shared memory machines, or by network conflicts of distributed memory machines. To evaluate the advantage a parallel computing system might have over a sequential machine, objective performance metrics are needed. Some of these measures are known as “speed-up”, “efficiency” and “scalability”.

2.2.1 SpeedUp

SpeedUp is a ratio of the time needed for the completion of a sequential algorithm to that time needed for the completion of a parallel algorithm run on P processors.

$$\text{SpeedUp} = \frac{Time_{sequential}}{Time(P)_{parallel}}, \tag{2.1}$$

where $Time(P)$ is the notation for parallel wall clock time as a function of the number of processors P. To get the best, and most widely acceptable ratio, it is important to evaluate this proportion using the fastest serial algorithm available.

2.2.2 Concurrent Efficiency

Concurrent or parallel efficiency, E, can be thought of as SpeedUp per processor (van de Geijn [48]). It is defined as the ratio of SpeedUp to the number of processors, P, on which the parallel algorithm is run

$$E = \frac{SpeedUp}{P}. \tag{2.2}$$

In dense linear algebra algorithms containing direct methods, such as LU factorization, the concurrent efficiency depends on problem size and the number of processors, so on a given parallel computer and for a fixed number of processors the running time should not vary greatly for problems of the same size (van de Geijn [48]). Thus, E may be expressed also as a function of matrix problem size n (where n is the number of columns of a square matrix),

$$E(n, P) = \frac{1}{P}\frac{Time(n)_{sequential}}{Time(n, P)_{parallel}}. \tag{2.3}$$

2.2.3 Scalability

Scalability defines the change in efficiency of a parallel algorithm as the number of processors and/or problem size varies, and is commonly based on the performance per processor at fixed memory per processor. An important presumption in a scalability analysis is that the chosen algorithm and problem size can be carried out on one processor. If a problem is too large for a sequential machine, then an adjusted scalability analysis should be considered. In any case, scalability issues should be examined during the development of parallel code, to assure efficiency as the number of available processors changes.

There are many theories that can be used to asses scalability and performance of a parallel algorithm. One of them is Amdahl's Law, which for a given algorithm/problem, is based upon the concept of comparing parallel performance to that of sequential performance through ratios similar to those seen above. Letting W_{par} = work that can be parallelized, W_{seq} = work that is inherently sequential, and P = the number of processors used, then the SpeedUp, S(P), may be expressed as

$$S(P) = \frac{W_{par} + W_{seq}}{\frac{W_{par}}{P} + W_{seq}}. \tag{2.4}$$

If we let $W = W_{seq} + W_{par}$, the total work in completion of a parallel algorithm, then Equation 2.4, can be rewritten to represent an expected bound in the value of SpeedUp as,

$$S(P) \leq \frac{W}{\frac{W_{par}}{P} + W_{seq}} \leq \frac{W}{W_{seq}} = \frac{1}{\frac{W_{seq}}{W}}. \tag{2.5}$$

This equation implies that the maximum SpeedUp of a parallel algorithm is bounded by its sequential portion of work. However this law assumes that the problem size

(i.e. square matrix dimension "n" in a linear algebra problem) is fixed as the number of processors increases. But if we assume that the problem size per processor remains constant, as suggested by Gustafson et al. [49], then we can improve this ratio by adding more processors as problem sizes get larger. Thus, Equation 2.5 may be rewritten as,

$$S(P) \leq \frac{W(P)}{\frac{W(P)_{par}}{P} + W(P)_{seq}} \leq \frac{W(P)}{W(P)_{seq}} = \frac{1}{\frac{W(P)_{seq}}{W(P)}}, \tag{2.6}$$

where $W(P)_{seq}$ is the amount of sequential work done as a function of the number of processors. Equation 2.6 shows that if the number of processors is allowed to grow along with the size of the problem, the ratio $\frac{W(P)}{W(P)_{seq}}$ will get larger and SpeedUp will not be limited by the serial fraction of work, but will instead be a linear function of the number of processors. Thus scalability for large parallel algorithms is achievable.

2.2.4 Communication

Inherent in the performance of a parallel algorithm is the transmission of data and instructions among processors. Introducing a model to understand the contribution of communication to overall cost and wall clock time is discussed in this section. The simplest example of communication is the sending of a message by one processor and its reception by another.

2.2.4.1 Point to Point Communication

The cost, or time, needed to send a message of length n bytes long between two directly connected processors, without conflicts, can be modelled as:

$$\alpha + n\beta \tag{2.7}$$

where α is a fixed startup cost, needed to get the attention of the other processor, independent of the message size, and β is the transmission cost per unit of data (Schultz [50]). The term α is independent of message length and is also referred to as the latency of communication and the reciprocal of β is called the communication bandwidth. The processor interface through which this fundamental message is transmitted, is described in the next section.

2.2.4.2 Message-Passing Interface (MPI)

Communication using message-passing between processors enables parallel computation to take place. Models of message-passing for coordination of data movement and instructions among processors are numerous and may not be the same for machines of different vendors. The attempt among a wide class of vendors and users to collect, improve and standardize the best features of the many message-passing systems, which have been developed over the years, resulted in the Message-Passing Interface (MPI) standard. MPI is not a revolutionary new way of programming parallel computers, but only the result of this consolidated effort. MPI is a library not a language, that is used to support the basic computational model, which is a collection of *processors* communicating with *messages* (Gropp [46]).

The most simple form of communication between two processors, is when one performs a *send* operation and the other performs a *receive* operation. For the *send* operation, the usual things to be specified are the data, with a starting address and length (in bytes for that data type) and its destination processor, commonly indicated with an integer. For the receiving processor, the minimal describing arguments will be those which designate where in local memory the data will be placed, address and length, along with an integer which identifies the origin of the incoming message. Screening parameters which enable a processor to control which messages it receives based on data type and origin are necessary to ensure *matching* of sent and received messages. Finally, a parameter to specify a size limit on a particular message is common. The following is then a minimal message interface (Gropp [46]):

send(address, length, destination, tag)

receive(address,length, source, tag, actlen)

where “tag” identifies data type (allowing matching or the control of which messages a processor is to receive), and “actlen” specifies the message length. The latter is commonly considered violated only if the message length is too long, but not if it is too short.

However, there are complications of describing the message buffer, which are not addressed by the above simple model. Data structures may be stored contiguously or non-contiguously, i.e. a matrix row stored in a column-wise format is not of stride one. Differing formats for data types between machines, such as in the case of parallel computing using workstation networks or in other instances of

heterogenous systems, where the floating point formats and lengths are different, calls for the need to specify messages more completely (Gropp [46]). MPI offers an approach to solve these problems, by specifying the parameter "count" and the versatile value "datatype", to further describe the message buffer. Thus with the triplet, (address, count, datatype), the number occurrences "count" of whatever defined "datatype" starting at "address", a message buffer can be described flexibly. For example (Gropp [46]), (A, 300, MPI_REAL) describes a vector of 300 real numbers, regardless of the length or format of a floating-point number. In an MPI implementation, the term MPI_REAL, will guarantee in a heterogenous network, that the same 300 reals will be received, even though the receiving machine may have a different floating-point format (Gropp [46]). Furthermore, the user may define his own "datatype" using MPI routines, to describe noncontiguous data. MPI also offers other advanced features such as "collective communication".

2.2.4.3 Collective Communication

Collective communication is a concept stemming from early message-passing libraries, that manifests itself in the notion of a *collective operation* performed by all processes in a computation (Schultz [50]). These communications are needed because most parallel algorithms require more complicated transmissions that cannot be efficiently constructed using necessary combinations of send and receive messages, represented by the point to point communication cost of Equation 2.7 in the minimal message interface.

There are two kinds of collective operations: data movement operations,

used to rearrange data among the processes, and collective computation operations, which include minimum, maximum, sum, logical OR, etc., as well as user defined operations (Gropp [46]). The cost of either collective operation depends on how much data is to be transmitted or the type of computation performed. These operations are implemented by the MPI libraries, and are listed below. Their lower bound costs is given in Table 2.1 (van de Geijn [51]), where α is the latency or communication startup cost, β is the cost to send one element of data (usually in units of bytes), γ is the cost of a floating-point operation (multiply or add) and P is the number of processors. Of course the linear sum of any row in this table would represent the total cost of an operation.

- **Broadcast**: Prior to this operation, one processor contains a piece of data, i.e. a vector. After the Broadcast, all processors contain a copy of the data.

- **Reduce to One**: Prior to this operation, each processor contains a vector. The operation calculates the sum of all vectors, element-wise, then sends the result to one processor.

- **Scatter**: Prior to this operation, one processor contains a vector of length n elements. After the Scatter, each processor owns a piece of the vector of length of length n/P, where P is the number of processors.

- **Gather**: Prior to this operation, each processor contains a piece of data, i.e. a piece of a vector. After the Gather, one processor contains the entire vector of data.

Operation	Latency	Bandwidth	Computation	MPI Call
Broadcast	$log_2(P)\alpha$	n β		MPI_Bcast
Reduce to One	$log_2(P)\alpha$	n β	$\frac{p-1}{p}n\gamma$	MPI_Reduce
Scatter	$log_2(P)\alpha$	$\frac{p-1}{p}n\beta$		MPI_Scatter
Gather	$log_2(P)\alpha$	$\frac{p-1}{p}n\beta$		MPI_Gather
Collect	$log_2(P)\alpha$	$\frac{p-1}{p}n\beta$		MPI_Allgather
Reduce to All	$log_2(P)\alpha$	$2\frac{p-1}{p}n\beta$	$\frac{p-1}{p}n\gamma$	MPI_Allreduce
Distributed Reduce	$log_2(P)\alpha$	$\frac{p-1}{p}n\beta$	$\frac{p-1}{p}n\gamma$	MPI_Reduce_scatter
All Scatter	$log_2(P)\alpha$	$\frac{p-1}{p}n\beta$		MPI_Alltoall

Table 2.1: Costs(time) of Collective Communication Operations

- **Collect**: Prior to this operation, each processor contains a piece of data, i.e. a piece of a vector. After the Collect, all processors contain a copy of the entire vector of data.

- **Reduce to All**: This operation is the same as **Reduce to One**, except that the resulting sum (vector) is duplicated to every processor.

- **Distributed Reduce**: This operation is the same as **Reduce to One**, except that afterwards, each processor will contain a portion of the resulting sum (vector).

- **All Scatter**: Equivalent to performing a **Scatter** on every processor.

2.3 Parallel Linear Algebra Package (PLAPACK)

Common approaches to the implementation and creation of high performance parallel dense linear algebra libraries have been seriously hindered by the

seemingly endless complexities of translating sequential algorithms to parallel code. These complications arise from the required manipulation of indices and parameters needed to describe data, its distribution to processors, and all supporting communication (ScaLAPACK [52]). Such intricacies lead inevitably to error-prone and bug-filled parallel code, exactly the condition which prevents the parallel implementation of sophisticated algorithms.

The Parallel Linear Algebra Package (PLAPACK) (pronounced PLAY-pack) is an infrastructure developed at the University of Texas at Austin that overcomes these obstacles. PLAPACK provides coding interfaces which can mirror natural descriptions of sequential dense linear algebra algorithms (van de Geijn et al. [37]), by employing an "object based" approach to programming. Such an approach is also utilized by the Message-Passing Interface (MPI), which is used in PLAPACK as the mechanism for the communication between processors. Descriptions of linear algebra objects such as the matrix or vector, are stored in *linear algebra objects*. The use of *view objects*, which are references into distributed matrices or distributed vectors, enables the addressing of sub-blocks of matrices and sub-segments of vectors. Therefore employing this philosophy/infrastructure, individual elements of a blocked linear algebra algorithm are not explicitly indexed, and a parallel algorithm can be written without explicitly referencing processor indices or communication. Through use of *views*, a PLAPACK implementation becomes a line-by-line translation of a given blocked algorithm.

2.3.1 Physically Based Matrix Distribution

The most distinguished fundamental principle of the PLAPACK infrastructure is its alternative approach to matrix distribution among processors in a parallel system. Traditional approaches to distributing a matrix among processors for a parallel dense linear algebra algorithm are primarily concerned with partitioning and distributing the matrices, not the accompanying vectors. This would seem to make sense based on the fact that a dense n x n matrix represents $O(n^2)$ memory and multiplications with one another requires $O(n^3)$ operations. Vectors on the other hand require much less memory and any computations involving them would only reach a maximum of $O(n^2)$ operations. However this reasoning fails to consider the needs of the application. A matrix is merely a linear operator which provides a transformation between vectors. It is the vectors in linear systems that naturally dictate the partitioning and distribution of work associated with (most) applications that lead to linear systems (Patra et al. [53]). Therefore by distributing the vectors, (the objects which contain information of physical significance, representing the physical problem) among processors, i.e. partitioning the vectors into sub-vectors and assigning them to processors, we distribute the underlying problem itself. This in turn, will induce matrix distribution among processors that is natural to the problem and which greatly simplifies the interfaces between PLAPACK applications and MPI libraries. This concept is known as *Physically Based Matrix Distribution* PBMD (van de Geijn [37], Patra et al. [53]). The following example illustrates this approach.

2.3.2 Simple Example: Matrix-Vector Multiply

The matrix-vector multiplication operation, $Ax = y$, is implemented by multiplying the matrix by vector x and overwriting the result into vector y, written as $Ax \longrightarrow y$, where the rows of A correspond to elements of y and elements of x correspond to columns of A. Both vectors are identically distributed among P processors in either a row-major or column-major order by logically viewing the processors as a two dimensional r x c mesh, where r and c are the number of rows and columns, respectively. In this example x and y are distributed in column-wise processor order, leading to the vector distribution where processor(i,j) holds the following subvectors:

$$\begin{bmatrix} x_{j*r} \\ \hline x_{j*r+1} \\ \hline \vdots \\ \hline x_{(j+1)*r-1} \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} y_{j*r} \\ \hline y_{j*r+1} \\ \hline \vdots \\ \hline y_{(j+1)*r-1} \end{bmatrix} \tag{2.8}$$

The distribution of matrix A (Patra et al. [53]) is such that processor (i, j) holds the sub-matrix $\tilde{A}_{i,j}$, written as

$$\tilde{A}_{i,j} = \begin{bmatrix} A_{i,j*r} & A_{i,j*r+1} & \cdots & A_{i,(j+1)*r-1} \\ \hline A_{i+r,j*r} & A_{i+r,j*r+1} & \cdots & A_{i+r,(j+1)*r-1} \\ \hline \vdots & \vdots & & \vdots \\ \hline A_{i+(c-1)*r,j*r} & A_{i+(c-1)*r,j*r+1} & \cdots & A_{i+(c-1)*r,(j+1)*r-1} \end{bmatrix} \tag{2.9}$$

With this distribution of matrix A and vectors x and y, each processor can locally execute its own matrix vector multiply, that leads to a local contribution of vector y from processor(i,j). After a summation within rows of processors, the resulting global vector y is completed and distributed among processors in column-major

format. The following steps to accomplish this operation are enumerated below and illustrated in Figure 2.4 (van de Geijn [37]).

1. Distribute subvectors of x among processors in column-major order, which dictates the distribution of matrix A as described above.

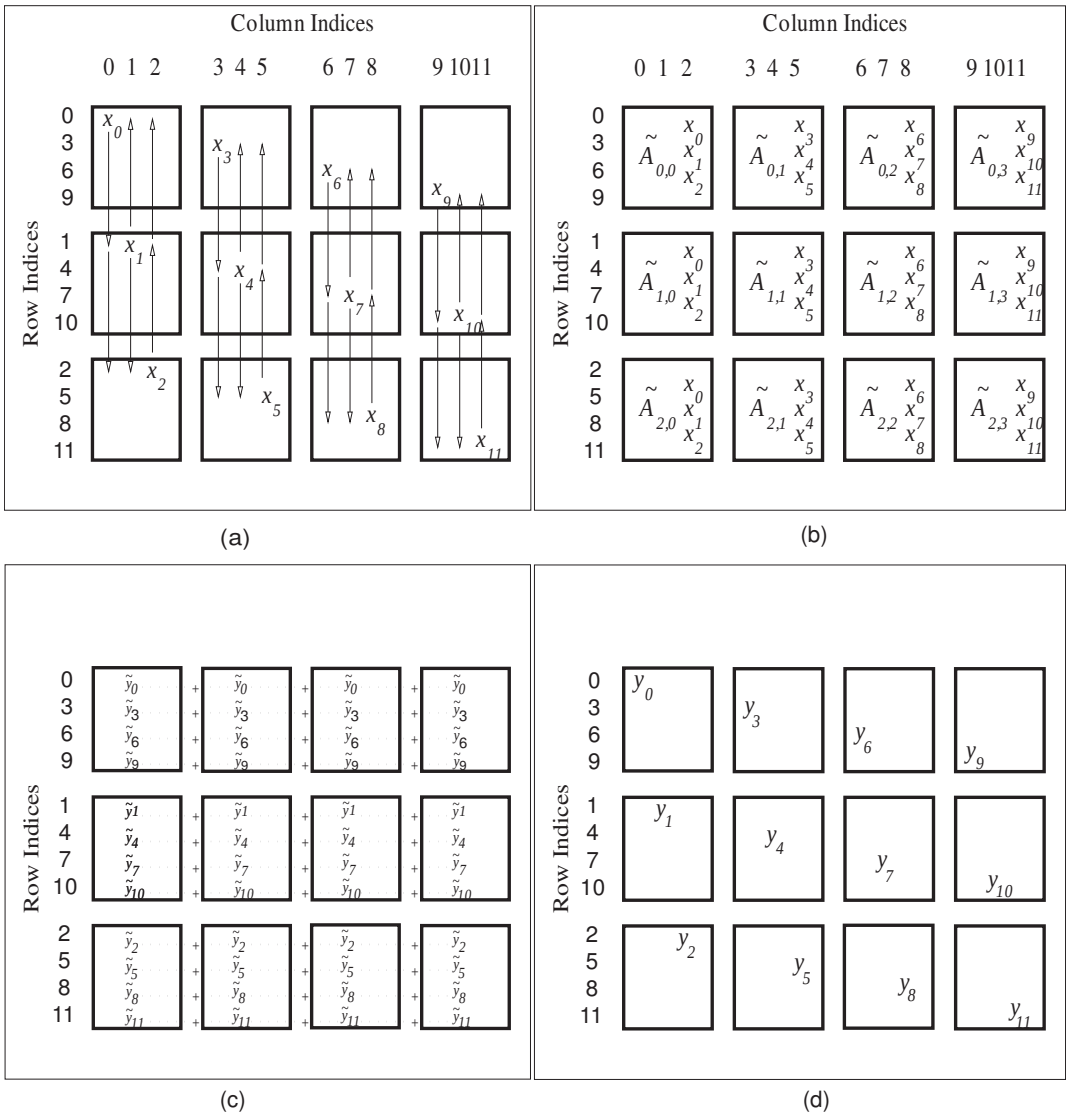
2. Broadcast (MPI_Allgather) the elements of x on each processor within each column of processors, thus duplicating the first r subsections of x on each processor in the first column of processors, the second r subsections of x on each processor in the second column of processors and so on, for all columns of processors.

3. Perform the local matrix vector multiply on every processor.

4. Perform summation within rows of processors of the local partial results to yield the desired vector y. This is accomplished with the Distributed Reduce (sum), (MPI_Reduce_scatter), operation, and leaves a portion of the resulting (sum) vector on every processor in a particular processor-row, a distribution format which is identical to that of vector x.

2.3.2.1 Theoretical Performance

Using the concepts of SpeedUp and concurrent efficiency, introduced by Equations 2.1 and 2.2, respectively, the derivation of performance equations for the matrix-vector-multiply is straightforward. The number of operations required for A times x is exactly $2n^2$, where n is the dimension of vector x. Therefore the



Parallel matrix-vector multiplication. (a) Sub-vectors of x start distributed amoung nodes in column-major order. (b) After spreading x within columns, local matrix-vector multiplication can commence. (c) Each node holds a partial contribution to the results that need to be summed within rows of nodes. (d) Sub-vectors of result y finish distributed among nodes in column-major order.

Figure 2.4: PBMD: Parallel matrix-vector multiplication.

sequential cost of this operation would be

$$Time(n)_{sequential} = 2n^2\gamma, \tag{2.10}$$

where γ is the time required for an arithmetic operation. The parallel cost on an $r \times c$ mesh of processors is roughly given by:

$$Time(n,r,c)_{parallel} = \frac{2n^2\gamma}{P} + T(n,r)_{broadcast} + T(n,r)_{dist.reduce}. \tag{2.11}$$

Substituting the cost expressions (van de Geijn [51])

$$\begin{aligned} T(n,r,c)_{broadcast} &= [log(P) + r + c - 2]\,\alpha + 2\frac{P-1}{P}n\beta \\ T(n,r,c)_{dist.reduce} &= log_2(P)\alpha + \frac{(c-1)^2 + c(r-1)^2}{P}n\beta + \frac{P-1}{P}n\gamma \end{aligned} \tag{2.12}$$

into Equation 2.11, yields,

$$SpeedUp(n,r,c) = \frac{2n^2\gamma}{\frac{2n^2\gamma}{P} + [log(P)+r+c-2]\alpha + 2\frac{P-1}{P}n\beta + log_2(P)\alpha + \frac{(c-1)^2+c(r-1)^2}{P}n\beta + \frac{P-1}{P}n\gamma}. \tag{2.13}$$

Dividing by P and simplifying terms, the expression for concurrent efficiency, becomes,

$$E(n,r,c) = \frac{1}{1 + \frac{[log(P)+r+c-2]\alpha P}{2n^2\gamma} + \frac{(P-1)\beta}{n\gamma} + \frac{\alpha P log_2(P)}{2n^2\gamma} + \frac{(c-1)^2+c(r-1)^2}{2n\gamma}\beta + \frac{P-1}{2n}}. \tag{2.14}$$

From equation 2.14, it can be seen that for a fixed number of processors in an $r \times c$ mesh, as the problem size n increases, the value of E(n,r,c) $\longrightarrow$ 1. E = 1 indicates full processor utilization. If n were held constant, while r and c increased, then E $\longrightarrow$ 0. E = 0 tells us that there is too little data and far too many processors. Introducing Gustafson's principle into Equation 2.14 by keeping the problem size per processor constant (Gustafson [49]) as the number of processors grows, the ratio of processors to matrix size will be given by

$$\frac{P}{n^2} = constant. \tag{2.15}$$

Furthermore, if r is kept equal to c, or their ratio $\frac{r}{c}$ is held constant, then it can be seen again that E(n,r,c) $\longrightarrow$ 1.

2.3.2.2 Actual Performance

To evaluate the parallel performance of the matrix-vector-multiply, $Ax \longrightarrow y$ operation, five experiments were conducted using various sizes of two dimensional logical meshes. Using the PLAPACK infrastructure, the tests were carried out at the National Aeronautics and Space Administration (NASA) Ames computational research facility on the Silicon Graphics Inc. (SGI) Origin 3000 machines and at the GeoForschungsZentrum (GFZ) in Potsdam, Germany on a Hewlett Packard (HP) machine.

The performance results of PLAPACK were excellent and clearly demonstrate the value of parallel computation. The theoretical predictions of the previous section can be seen in all of the figures illustrating the timing, SpeedUp,

and efficiency curves resulting from the experiments. Figure 2.5 shows that for a small number (four) of processors, the performance of the sequential matrix-vector-multiply is equal to or better than that of the parallel implementation. This is because the communication costs of the parallel algorithm dominate and outweigh the benefits of distributing the computational burden among four processors. However, as the matrix size increases, it can be seen that the advantages of parallel computation quickly begin to prevail over the communication costs, resulting in a performance superior to that produced by a single processor. Figures 2.6, 2.7, and 2.8 illustrate very similar behavior in performance as the number of processors and mesh size increases. It can be seen that the parallel times continue to decrease, resulting in an ever improving performance of the parallel matrix-vector-multiply.

Figure 2.9 shows that for an increasing problem size n, a larger number of processors (8x8) will deliver more speed up than the smaller mesh sizes. This is due to the fact that more processors can contain more data, perform more floating point operations for increasing problem sizes while doing less memory operations in comparison to the fewer processors in the smaller meshes.

This example illustrated that the theoretical concepts of SpeedUp and efficiency can be used in helping predict actual performance of parallel algorithms. It was seen that communication costs can degrade performance when the problem size is too small for a given number of processors. There are other obstacles that complicate the programming for a parallel algorithm. An important one is the issue of load balancing, i.e. the equal distribution of computation and communication burden among processors. By using PLAPACK and the library of MPI, all of these

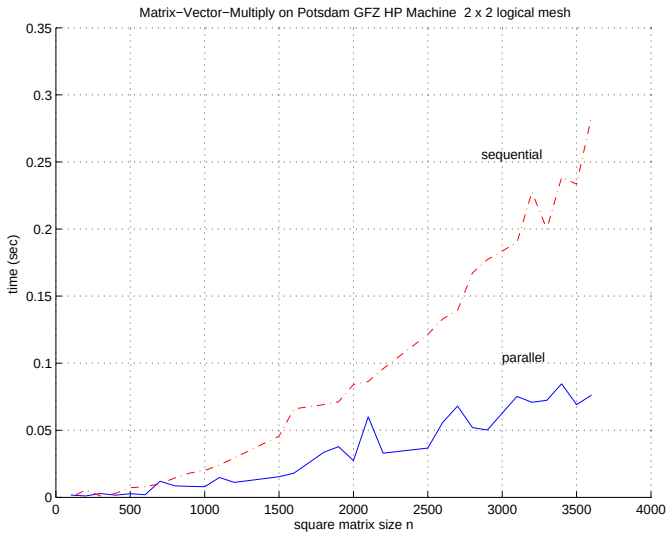


Figure 2.5: Parallel and Sequential Times for the Matrix-Vector-Multiply using a 2x2 mesh of the GFZ HP Machine.

concerns can be adequately addressed, allowing scalability to be achieved for ever increasing problem sizes.

2.4 Summary

A brief outline of parallel computation, regarding machine architecture, parallel performance theory and communication has been given. The next chapter presents a discussion of precise orbit determination and some of the standard methods used to estimate a gravity field.

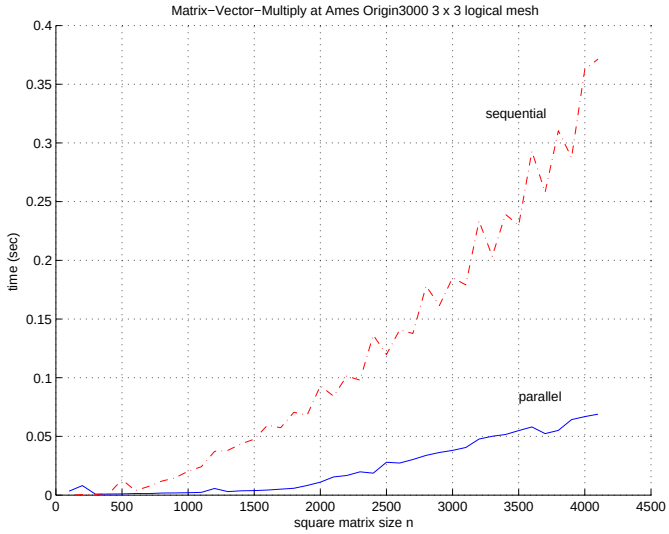


Figure 2.6: Parallel and Sequential Times for the Matrix-Vector-Multiply using a 3x3 mesh of the the NASA SGI Machine.

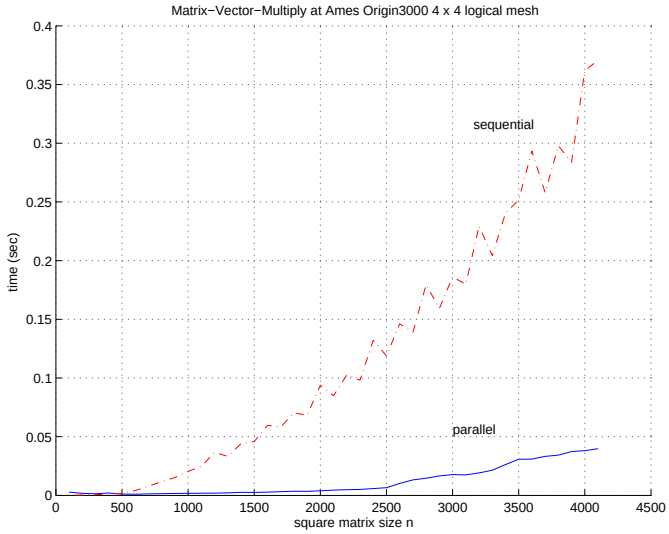


Figure 2.7: Parallel and Sequential Times for the Matrix-Vector-Multiply using a 4x4 mesh of the the NASA SGI Machine.

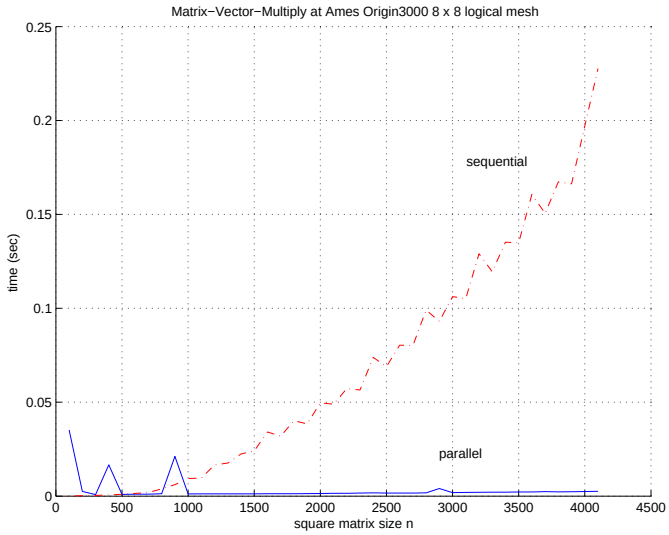


Figure 2.8: Parallel and Sequential Times for the Matrix-Vector-Multiply using a 8x8 mesh of the the NASA SGI Machine.

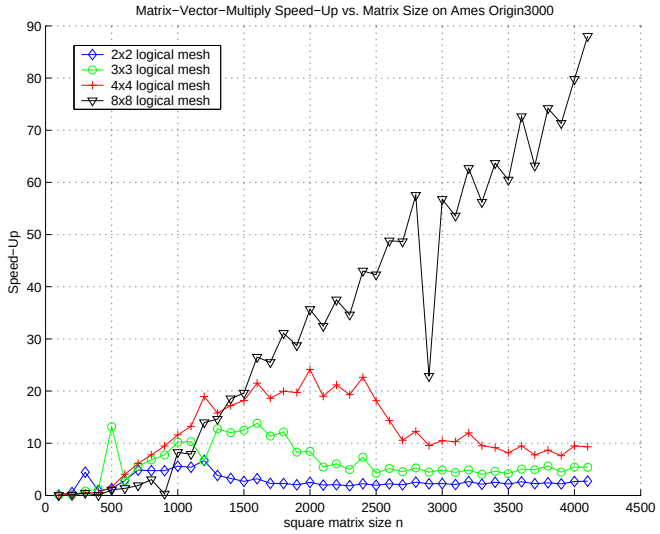


Figure 2.9: Parallel Matrix-Vector-Multiply SpeedUp for various mesh sizes.

Chapter 3

Precision Orbit Determination

With the start of the year 1600, the evolution of orbit determination began in middle Europe with the fruitful cooperation between two very significant historical figures, Johannes Kepler and Tycho Brahe. These two men concerned themselves with solving the perplexing problem of the erratic behavior of planet Mars in its heliocentric trajectory. After Kepler abandoned his assumption of perfect circular motion, he was able to match Brahe's exquisite orbital observations to an elliptical path. Thus, Kepler was able to determine the true shape of Mars' orbit.

Much later, in 1795, Karl Friedrich Gauss invented the process of least squares, providing a firm computational basis of orbit prediction (Vallado [1]). Gauss' next remarkable achievement was to accurately predict the reappearance of the asteroid cluster Ceres from behind the Sun on New Year's Day in 1802 (Bate, Mueller, White [54]). The goal of orbit prediction and determination is to obtain accurate ephemeris (positions and velocities) of an orbiting satellite, using temporal sequences of observations. By integrating the equations of motion of a satellite from a reference epoch to the time of a true observation in relation to the model of how the satellite is observed, a *predicted* observation is produced. The difference between a predicted and a true observation is called a residual. Minimizing

the residuals for all observations in a Gauss least squares sense, is an estimation process that determines the kinematic and dynamic parameters which describe the satellite's ephemeris and those which designate the participating models.

3.1 Batch Filter Estimation

The equations of motion of a satellite in orbit are represented in vector form, as a system of linear first order ordinary differential equations, with time t as the independent variable,

$$\dot{X}(t) = F[X(t), t] \quad \text{and} \quad \dot{X}^*(t) = F[X^*(t)], \tag{3.1}$$

both for the true and nominal state, respectively, where

$$\begin{aligned} X &= \begin{bmatrix} \overrightarrow{r} \\ \overrightarrow{v} \\ \overrightarrow{\alpha} \end{bmatrix} \text{ (n x 1 vector)} \\ \overrightarrow{r} &= \text{satellite position (3 x 1 vector)} \\ \overrightarrow{v} &= \text{satellite velocity (3 x 1 vector)} \\ \overrightarrow{\alpha} &= \text{vector of model parameters} \\ F &= \text{derivatives of the state (n x 1 vector)} \end{aligned}$$

The term "nominal" refers to the state of the satellite that is computed, using mathematical and physical models, and is given the "*" notation. The initial conditions are $X(t_0)$ and $X^*(t_0)$. Knowing that the true state $X(t)$ is the combination of the nominal and some deviation, $x(t)$

$$X(t) = X^*(t) + x(t), \tag{3.2}$$

where $x(t)$ is an (nx1) vector of deviations away from the computed nominal value of the state $X^*(t)$, a weak variation equation of state may be constructed. Rearranging terms in Equation 3.2, making one substitution, and taking the derivative with respect to time, we get

$$\begin{aligned}\dot{x}(t) &= \dot{X}(t) - \dot{X}^*(t) \\ &= F[X(t),t] - F[X^*(t),t] \\ &= F[X^*(t)+x(t),t] - F[X^*(t),t]\end{aligned}$$

Expanding this equation in a Taylor series about the nominal trajectory, and ignoring higher order terms, leads to,

$$\begin{aligned}\dot{x}(t) &= F[X^*(t),t] + \left.\frac{\partial F[X^*(t),t]}{\partial X^*(t)}x(t)\right|_{\text{eval. on nominal}} + \text{higher order terms} - F[X^*(t),t] \\ &= A(t)x(t),\end{aligned}$$

where

$$A(t) = \frac{\partial F[X^*(t),t]}{\partial X^*(t)} = \begin{bmatrix} \frac{\partial F_1}{\partial X_1} & \frac{\partial F_1}{\partial X_2} & \cdots & \frac{\partial F_1}{\partial X_n} \\ \frac{\partial F_2}{\partial X_1} & \frac{\partial F_2}{\partial X_2} & \cdots & \frac{\partial F_2}{\partial X_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial F_n}{\partial X_1} & \frac{\partial F_n}{\partial X_2} & \cdots & \frac{\partial F_n}{\partial X_n} \end{bmatrix} \quad \text{(nxn matrix)} \tag{3.3}$$

The linearized differential Equation 3.3 has the solution

$$x(t) = \Phi(t,t_0)x(t_0), \tag{3.4}$$

where $x(t_0)$ is the value of x(t) at epoch t_0 and $\Phi(t,t_0)$ is the state transition matrix, which relates a deviation to the state at some time t to the state at t_0 (Gelb [55]). The matrix $\Phi(t,t_0)$ satisfies the differential equation

$$\dot{\Phi}(t,t_0) = A(t)\Phi(t,t_0), \tag{3.5}$$

with the initial condition of $\Phi(t_0, t_0) = I$, the identity matrix. By numerically integrating Equation 3.5, the Equation 3.4 may be obtained. This deviation is then related to the satellite observation residuals through the linearized observation-state relation. With this relation, observation residuals are used to estimate corrections to the nominal state vector. By using the state transition matrix $\Phi(t, t_0)$ to map all observations to some common epoch $x(t_0)$, the corrections are applied at time $t = t_0$. If there are p actual observations taken at time t, they can be represented by the (p x 1) vector $Y(t)$. The observation-state equation is assumed to have the following form

$$Y(t) = G[X(t), t] + \epsilon(t) \tag{3.6}$$

where $G[X(t), t]$ is a (p x 1) vector representing the mathematical model of satellite observations. The (p x 1) vector ϵ represents the errors of commission and omission in the mathematical models of motion. Using Equation 3.2 and expanding Equation 3.6 in a Taylor series and dropping terms higher than first order, we obtain the following equation which relates an observation residual y(t) to x(t),

$$\begin{aligned} Y(t) &= G[X^*(t) + x(t), t] + \epsilon(t) \\ Y(t) &= G[X^*(t)] + \frac{\partial G[X^*(t), t]}{\partial X^*(t)} x(t) + \epsilon(t) \\ Y(t) - G[X^*(t)] &= \frac{\partial G[X^*(t), t]}{\partial X^*(t)} x(t) + \epsilon(t) \\ y(t) &= \tilde{H}(t) x(t) + \epsilon(t), \end{aligned} \tag{3.7}$$

where $\tilde{H}(t)$ is defined by ,

$$\tilde{H}(t) = \frac{\partial G[X^*(t), t]}{\partial X^*(t)} \tag{3.8}$$

and $\epsilon(t)$ now contains errors due to linearization in the observation and motion models. Inserting Equation 3.4 into Equation 3.7 we find that

$$y(t_i) = \tilde{H}(t_i)\Phi(t_i, t_0)x(t_0) + \epsilon(t_i). \tag{3.9}$$

If we let $H(t_i) = \tilde{H}(t_i)\Phi(t_i, t_0)$,we then have the following expression which maps an observation taken at time t_i, to the initial time t = t_0

$$y(t_i) = H(t_i)x(t_0) + \epsilon(t_i), \tag{3.10}$$

where y(t_i) and $\epsilon(t_i)$ are (p x 1) vectors, $x(t_0)$ is (n x 1) and $H(t_i)$ is a (p x n) matrix. If a set of observations, termed a "batch", is taken at times $[t_1, t_2, \cdots t_k]$, all can be represented by one equation in the following matrix form.

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_k \end{bmatrix} = \begin{bmatrix} H(t_1) \\ H(t_2) \\ \vdots \\ H(t_k) \end{bmatrix} x_0 + \begin{bmatrix} \epsilon(t_1) \\ \epsilon(t_2) \\ \vdots \\ \epsilon(t_k) \end{bmatrix} \tag{3.11}$$

Compactly written as

$$y = Hx_0 + \epsilon, \tag{3.12}$$

where y has the dimensions (kp x 1 = m x 1), H is (kp x n = m x n), x_0 is (n x 1) and ϵ is (kp x 1 = m x 1). (Note: H often has the name "information matrix") Usually the number of observations is much greater than the number of parameters to be solved (for), m $\gg$ n and the observations are assigned weights. Thus the solution for x_0 in Equation 3.12 can be obtained by using the weighted least squares estimation technique. This can be done by either forming the normal equation to solve for x_0 or by directly performing an orthogonal factorization on the information matrix

H. Another approach is to employ the linear unbiased minimum variance estimate method. These methods are discussed in the following sections.

3.1.1 Forming the Normal Equation

The solution to Equation 3.12 is an estimated correction vector $\hat{x}$ that is added to the nominal state vector $X^*(t)$ at the initial epoch, t_0, namely

$$X^*(t_0) = X^*(t_0) + \hat{x}(t_0). \tag{3.13}$$

This correction vector can by obtained by minimizing the weighted sum of the square of the observation residuals as defined by the performance index J,

$$J = \epsilon^T W \epsilon. \tag{3.14}$$

Rearranging Equation 3.10, and dropping the indices i and 0 for simplicity, we find that at time t,

$$\epsilon(t) = y(t) - H(t)x(t), \tag{3.15}$$

leads to

$$J = [y(t) - H(t)x(t)]^T W [y(t) - H(t)x(t)], \tag{3.16}$$

where W is a diagonal matrix containing assigned observations weights, a topic which is discussed in a later section. Setting the first variation of Equation 3.16 equal to zero, results in the following normal equation of the linear system,

$$(H^T W H)\hat{x} = H^T W y. \tag{3.17}$$

Of course when inversion of the matrix $(H^T W H)$ is possible, either by direct or indirect means, the solution is written as

$$\hat{x} = (H^T W H)^{-1} H^T W y. \tag{3.18}$$

(The notation $\hat{x}$ implies that the correction vector which satisfies the first variation condition, occurs at an extremum.) The variance and covariance of the unbiased estimated correction vector $\hat{x}$ is given by (Tapley [56]) as

$$\begin{aligned} P &= E[(\hat{x} - E[\hat{x}])(\hat{x} - E[\hat{x}])^T] \\ &= E[(\hat{x} - x)(\hat{x} - x)^T] \\ &= (H^T W H)^{-1} H^T W E[\epsilon\epsilon^T] W H (H^T W H)^{-1}. \end{aligned} \tag{3.19}$$

When the weighting matrix W is chosen to equal the inverse of *observation* covariance, i.e. $W = \left\{E[\epsilon\epsilon^T]\right\}^{-1}$, Equation 3.19 reduces to the following simple form

$$P = (H^T W H)^{-1}. \tag{3.20}$$

To include a previous estimate $\overline{x}$, known as an *a priori* estimate and its corresponding error covariance $\overline{P}$ into a current estimation, after the proper mapping has been carried out, the performance index must first be redefined at time t as

$$J = [y(t) - H(t)x(t)]^T W [y(t) - H(t)x(t)] + [\overline{x}(t) - x(t)][\overline{P}(t)^{-1}][\overline{x}(t) - x(t)]. \tag{3.21}$$

Setting the first variation equation equal to zero, and dropping the index t for ease of presentation, gives the adjusted weighted least squares estimate, for the inclusion

of *a priori* information, as

$$\hat{x} = [(H^T W H) + \overline{P}^{-1}]^{-1}[H^T W y + \overline{P}^{-1}\overline{x}], \tag{3.22}$$

Again, by letting $W = \left\{E[\epsilon\epsilon^T]\right\}^{-1}$, the resultant error covariance is

$$P = \left[(H^T W H) + \overline{P}^{-1}\right]^{-1}. \tag{3.23}$$

3.1.2 Orthogonal Factorization

Another name for orthogonal factorization is QR decomposition. It is an important solution technique, because the disadvantage of the normal-equation approach is that it is sometimes less accurate than the QR approach. In fact, critical information can be lost when $H^T H$ is formed (Watkins [7], Tapley [45]). A simple example taken from (Watkins [7]), illustrates an issue that can occur for any size problem. Let

$$H = \begin{bmatrix} 1 & 1 \\ \tau & 0 \\ 0 & \tau \end{bmatrix}, \tag{3.24}$$

where $\tau > 0$, is small. Clearly H has full rank, and

$$H^T H = \begin{bmatrix} 1+\tau^2 & 1 \\ 1 & 1+\tau^2 \end{bmatrix} \tag{3.25}$$

which is positive definite. However if τ is small enough such that τ^2 is below machine precision, i.e. numerically zero, then the computed $H^T H$ will be $\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$, which is singular. Of course using double precision arithmetic will, in many cases of the least square approach, be an adequate remedy. Despite this numerical property and because H is often sufficiently well conditioned, (and despite the fact that

the condition number of $H^T H$ is the square of the condition number of matrix H) the normal equation approach is frequently used. For further discussion on the sensitivity of the least-squares problem, the text by Watkins [7] is suggested to the curious reader. However, by using the QR approach, the matrix $H^T H$ is not formed, rather the matrix H is used directly in the solution process. The matrix Q is discussed in Chapter Five, however let us introduce its definition here. The concept of orthogonality between two arbitrary vectors $u, v \in \Re^3$, is commonly defined by the dot product $\langle u, v \rangle$ = 0, namely that the angle between them is $\frac{\pi}{2}$. Extending the orthogonality concept to matrices, a matrix $Q \in \Re^{nxn}$ is said to be orthogonal if $QQ^T = I$. This equation also says that Q has an inverse, and $Q^{-1} = Q^T$ (Watkins [7]). The following theorem and proof (Watkins [7]), justify why/how orthogonal factorization can be used as a solution technique.

Theorem 3.1.1. *If $Q \in \Re^{nxn}$ is orthogonal, then for all x,y $\in \Re^n$,*

$$\begin{aligned} (a) \quad \langle Qx, Qy \rangle &= \langle x, y \rangle \\ (b) \quad \|Qx\|_2 &= \|x\|_2 \end{aligned}$$

Proof.

$$\begin{aligned} (a) \quad \langle Qx, Qy \rangle &= (Qy)^T Qx = y^T Q^T Qx = y^T x = \langle x, y \rangle \\ (b) \quad \| \langle Qx, Qx \rangle \|_2 &= (Qx)^T Qx = x^T Q^T Qx = x^T x = \| \langle x, x \rangle \|_2 \end{aligned}$$

□

Because part (b) of the Theorem 3.1.1 says that Qx and x have the same length, *orthogonal transformations preserve lengths*. By combining parts (a) and

(b), and using the Law of Cosines (which permits the computation of the cosine of an angle by the dot product of two vectors) it is clear that

$$\arccos\frac{\langle Qx, Qy\rangle}{\|Qx\|_2\|Qy\|_2} = \arccos\frac{\langle x, y\rangle}{\|x\|_2\|y\|_2}. \tag{3.26}$$

Because the angle between Qx and Qy is the same as the angle between x and y, *orthogonal transformations preserve angles*. Therefore the application of Q in the solution of a least squares problem is permitted because when A and b are replaced by QA and Qb, respectively, all lengths and angles are preserved. Rewriting the performance index seen in Equation 3.16 in the form of the Euclidean norm, to illustrate how Q is directly applied to H, and dropping the index t for simplicity,

$$\begin{aligned} J &= [y - Hx]^T W^{\frac{T}{2}} W^{\frac{1}{2}} [y - Hx] \\ &= \left\{W^{\frac{1}{2}}[y - Hx]\right\}^T \left\{W^{\frac{1}{2}}[y - Hx]\right\} = \|W^{\frac{1}{2}}[y - Hx]\|^2 \\ &= \|QW^{\frac{1}{2}}[y - Hx]\|^2. \end{aligned} \tag{3.27}$$

By choosing the orthogonal transformation matrix Q such that

$$QW^{\frac{1}{2}}H = \begin{bmatrix} R \\ 0 \end{bmatrix}, \tag{3.28}$$

where R is an upper triangular n x n matrix in the top portion of the m x n transformed H. Applying the same Q to the right hand side of $Hx = y$, where y is an (m x 1) vector, we get

$$QW^{\frac{1}{2}}y = \begin{bmatrix} b \\ e \end{bmatrix}, \tag{3.29}$$

where b and e are vectors of dimension n and (m-n), respectively. Because the transformation produces the upper triangular matrix R, this orthogonal factorization

is commonly called the QR decomposition. The performance index is now written as

$$\begin{aligned} J &= \left\| \begin{bmatrix} R \\ 0 \end{bmatrix} x \quad - \quad \begin{bmatrix} b \\ e \end{bmatrix} \right\|^2 \\ J &= \|Rx - b\|^2 + \|e\|^2. \end{aligned} \tag{3.30}$$

By inspection, the solution which minimizes J is $R\hat{x} = b$, which is easily obtained by simple backwards substitution, a less costly endeavor than the inversion of the normal matrix $H^T H$. Thus, J will equal the scalar $\|e\|^2$ and represents its minimum value. Since $\|e\|^2$ is an approximation of the root mean square (RMS) of the observation residuals for the estimated solution, it is also termed the linear predicted RMS. By letting $W = \left\{E[\epsilon\epsilon^T]\right\}^{-1}$, as was the case in the formation of the normal equation, and assuming that the errors in ϵ are random only, i.e. E[ϵ] = 0, the covariance matrix can be written as

$$P = (H^T W H)^{-1} = (R^T R)^{-1} \tag{3.31}$$

The calculation technique, seen in Hinga et al. [57], for the covariance after orthogonalization has been completed, has a great advantage over the standard method used to compute covariance of the normal equation solution. By using the matrix R produced by the QR decomposition for example, covariance computation can be performed "in place", meaning that no memory beyond that required to store the (n x n) matrix R is needed. Also, in cases where R is ill-conditioned, the Singular Value Decomposition (SVD) may be applied and used to generate stabilized solutions, by manipulating the singular values. Such an application is a topic of this investigation.

3.1.3 Linear Unbiased Minimum Variance Estimate (LUMVE)

Another methodology that can be applied in the estimation of a gravity field is called the Linear Unbiased Minimum Variance Estimate (LUMVE). The equations for state and data, given by 3.4 and 3.12 respectively, repeated here are,

$$x(t) = \Phi(t, t_0)x(t_0), \tag{3.32}$$

and

$$y = Hx_0 + \epsilon, \tag{3.33}$$

are used in the LUMVE method. Although the state is considered a constant variable, the errors ϵ and the data y are modelled as random variables. Using the breve notation to denote random variables, we rewrite 3.33 as

$$\breve{y} = Hx + \breve{\epsilon}. \tag{3.34}$$

It is assumed that observation error has a zero mean, $E[\breve{\epsilon}] = 0$ and some known observation covariance R (not to be confused with the orthogonalized R matrix in the previous section), defined here as

$$E[\epsilon\epsilon^T] = R, \tag{3.35}$$

where E represents the expected value. This estimate requires $\hat{x}$ to have a linear relation to $\breve{y}$ in the form

$$\hat{x} = M\breve{y}, \tag{3.36}$$

where M is an (m x n) matrix, with $m > n$. Because we have the requirement that the estimate be unbiased,

$$E[\hat{x}] = x \text{ and } E[\breve{\epsilon}] = 0. \tag{3.37}$$

By substituting Equations 3.36 and 3.34 into 3.37

$$E[\hat{x}] = E[M\breve{y}] = E[M(Hx + \breve{\epsilon})] = MHE[x] + E[\breve{\epsilon}] = MHx, \tag{3.38}$$

leads to the constraint that gives and unbiased estimate, $MH = I$. The error covariance matrix is defined as

$$P = E[(\hat{x} - E[\hat{x}])(\hat{x} - E[\hat{x}])^T]. \tag{3.39}$$

By substituting 3.34, and Equations 3.35 through 3.37 and imposing constraint $MH = I$, one obtains

$$P = MRM^T. \tag{3.40}$$

To find the minimum variance, the trace of P is minimized by adding the constraint $MH = I$ and its transpose with a Lagrange multiplier, Λ, in the following form,

$$P = \text{trace}\left\{MRM^T + \Lambda[MH - I]^T + [MH - I]\Lambda^T\right\}. \tag{3.41}$$

By setting small delta changes ΔM and $\Delta\Lambda$ to zero and ignoring all second order terms and higher, the first variation condition can be satisfied, yielding

$$M = (H^T R^{-1} H)^{-1} H^T R^{-1}. \tag{3.42}$$

Substituting this M into Equation 3.36 and 3.40 gives

$$\hat{x} = (H^T R^{-1} H)^{-1} H^T R^{-1} \breve{y} \tag{3.43}$$

and

$$P = (H^T R^{-1} H)^{-1}. \tag{3.44}$$

The immediate question is whether we have a minimum and not a maximum, produced by this first variation condition. Given that the covariance matrix of observations, R, is full rank, and the fact that it is usually positive definite under usual circumstances, i.e. the spectral norm of R, $\|\mathrm{R}\|_2 > 0$, is an assurance that we have a minimum. For the skeptic, the second variation derivation and the proof that Equation 3.43 corresponds to a minimum, the text by Poole [58] is suggested.

Often it is desirable to use an *a priori* estimated correction to the state x, and the corresponding covariance P as the initial condition for the current time. After properly mapping them to the current epoch, denoted by $\overline{x}$ and $\overline{P}$, they can then be propagated to obtain a future state. For the previous and current data equations, we have

$$\text{old} \quad \overline{x} = Ix + \eta \qquad \text{current} \quad y = Hx + \epsilon \tag{3.45}$$

where η is the previous observation error. For both old and current, the observation errors and error covariance are defined as

$$E[\epsilon] = 0, \quad E[\epsilon\epsilon^T] = R, \quad E[\eta] = 0, \quad E[\eta\eta^T] = \overline{P}. \tag{3.46}$$

To combine these equations together, it is assumed that previous and current observation errors are not correlated, $E[\epsilon\eta^T] = 0$ and $E[\eta\epsilon^T] = 0$. This leads to

$$\acute{y} = \begin{bmatrix} y \\ \overline{x} \end{bmatrix}, \quad \acute{H} = \begin{bmatrix} H \\ I \end{bmatrix}, \quad \acute{\epsilon} = \begin{bmatrix} \epsilon \\ \eta \end{bmatrix}, \quad \text{and} \quad \acute{R} = \begin{bmatrix} R & 0 \\ 0 & \overline{P} \end{bmatrix}. \tag{3.47}$$

Substituting $\acute{R}, \acute{H},$ and $\acute{y}$ into Equation 3.43, the linear unbiased minimum variance estimate, that includes *a priori* previous state, yields

$$\hat{x} = [(H^T R^{-1} H) + \overline{P}^{-1}]^{-1} [H^T R^{-1} y + \overline{P}^{-1} \overline{x}]. \tag{3.48}$$

Finally, the error covariance that includes previous covariance is expressed as

$$P = [(H^T R^{-1} H) + \overline{P}^{-1}]^{-1}. \tag{3.49}$$

Notice when $W = R^{-1}$, the estimates given by the normal equation in weighted least squares and LUMVE are equivalent, Equations 3.22 and 3.48, respectively.

3.1.4 Description of the H Matrix

The observation partials matrix H, often called the *sensitivity* or *information* matrix, relates how observations are affected by changes in state. As seen in Equation 3.9, H is the product of $\tilde{H}(t_i)$ and the state transition matrix $\Phi(t, t_0)$, where the former is derived from G, the observation model. A simple example of this model is that of the range between an observing terrestrial station and a satellite, shown in Figure 3.1 where $\overline{r}$ equals the inertial coordinates of the satellite, $\overline{R}_{s.i.}$ is the relative range vector from the ith station to the satellite in the inertial coordinate frame, and $\overline{r}_{s.i.}$ is the inertial position vector of the ith station. G = $G(X(t), t) = \left\|\overline{R}_{s.i.}\right\|$, is the magnitude of the relative position vector between the satellite and the ith observing station. Thus the scalar range G is expressed as,

$$\begin{aligned} G &= [(x - x_{ECEF}\cos\theta + y_{ECEF}\sin\theta)^2 + (y - x_{ECEF}\sin\theta - y_{ECEF}\cos\theta)^2 \\ &\quad +(z - z_{ECEF})^2]^{\frac{1}{2}}, \end{aligned} \tag{3.50}$$

where the Earth Centered Earth Fixed (ECEF) (x,y,z) coordinates of the ith station have been transformed through the Greenwich hour angle θ into the Earth centered inertial frame and subtracted from the inertial position vector $\overline{r}$(x,y,z) of the satellite. Utilizing analytical or numerical techniques, G is used to evaluate the partials

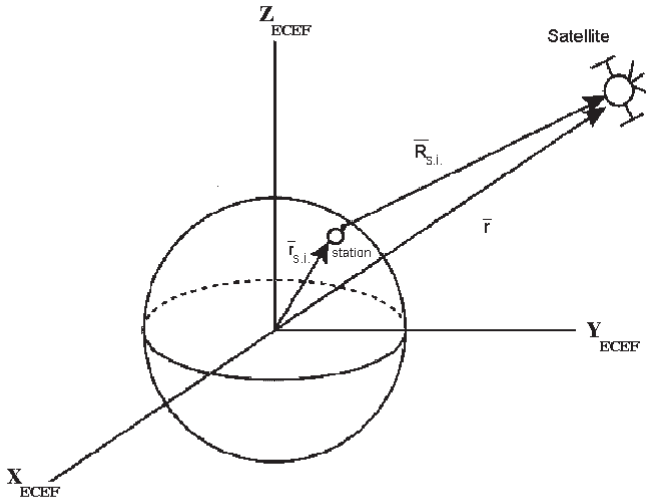


Figure 3.1: Observation Model.

for row vector $\tilde{H}(t)$,

$$\tilde{H}(t) = \frac{\partial G[X^*(t)]}{\partial X^*(t)} = \left[\begin{array}{cccc} \frac{\partial G[X^*(t)]}{\partial X^*(t)_1} & \frac{\partial G[X^*(t)]}{\partial X^*(t)_2} & \cdots & \frac{\partial G[X^*(t)]}{\partial X^*(t)_n} \end{array} \right], \tag{3.51}$$

where n is the number of unknown parameters to be estimated. Multiplication by the state transition matrix is then carried out to map dynamic parameters to the required epoch, resulting in the matrix H. It can been seen here, and in Equation 3.9, that each row of H contains the relation between one observation and a change in state, mapped to the time of epoch. Therefore all rows of H relate how a change in the initial state (at epoch) affect all measurements in a given batch of observations. Of course more than one observation model may be included into a batch of observations which make up all of the rows of the sensitivity matrix H.

3.2 Weighting of Observation Data

This section discusses the issue of assigning the proper amount of "influence" or weight that an observation is allowed to have during the estimation process of a gravity field. Because not all observations are created equal or come from identical sensor equipment, the accuracy of each individual observation or class of observations must be taken into account. Assigning proper weights to satellite observation data to combine them into a single information equation is the most difficult step in the gravity solution process and is often performed in an iterative manner, that is empirical in nature. Of course the chosen observation weights affect both the quality of the gravity solution and its error covariance, so great care is needed in their selection. If, the observation errors were exactly known, then the square of their inverses, $\frac{1}{[\sigma(i)_{obs}{}^2]}$ would provide ideal data weights, where $\sigma(i)$ is the ith standard deviation error of the ith satellite observation. However, these observations errors are never exactly known, so they are approximated by the given accuracy of the corresponding measurement system. Furthermore, the observation model error for a class of observations is also considered in the selection of observation data weights. An iterative algorithm, which automatically determines observation data weights was developed by Yuan [6]. For an additional ith data set in the solution, W_i is defined as the ratio of two scalars, namely

$$W_i = \frac{m_i}{(y_i - H_i\hat{x})^T {R_i}^{-1}(y_i - H_i\hat{x})}, \tag{3.52}$$

where m_i is the number of observations in the ith additional data set, ${R_i}^{-1}$ is the initial weight based on the accuracies and models of the observation system class

when the information equation for that data set was created, $\hat{x}$ is the corresponding estimate of the state deviation from the initial gravity solution, and y_i and H_i are the observation residual vector and observation partials matrix, respectively, of the ith data set. Remember that the calculated scalar values of Equation 3.52 are located along the diagonal of the matrix W_i. The new solution that uses these weighted observation data sets is expected to return smaller residuals for a better "fit" to the observations, than was seen in the initial solution. Continuing this scheme, the corresponding variables, y_i, H_i and ${W_i}^{-1}$ of the new solution are then utilized to calculate new weights for the observation data sets, for a better solution. This procedure is repeated until a defined criterion of convergence has been satisfied. The Center for Space Research at the University of Texas at Austin uses this algorithm in most gravity field solutions.

Another technique for determining weights, compares the changes in solution and error covariance as the result of introducing particular data sets into the solution. This technique is called independent and subset solutions (Nerem [59]). The distinction between two different estimated gravity solutions $\hat{X}_1$ and $\hat{X}_2$, that have been solved by using the same reference gravity field, is equal to the difference between their state deviation estimates, $\hat{x}_1$ and $\hat{x}_2$, namely,

$$\begin{aligned} \hat{X}_1 - \hat{X}_2 &= (X_1{}^* + \hat{x}_1) - (X_2{}^* + \hat{x}_2) \\ &= \hat{x}_1 - \hat{x}_2 \quad \text{if} \quad \hat{X_1}^* = \hat{X_2}^*. \end{aligned} \tag{3.53}$$

The covariances of $\hat{X}_1$ and $\hat{X}_2$ are of the same form seen in Equation 3.19, rewritten

here as

$$P_1 = E[(\hat{x}_1 - x)(\hat{x}_1 - x)^T] = E[\epsilon_1 {\epsilon_1}^T] \tag{3.54}$$

$$P_2 = E[(\hat{x}_2 - x)(\hat{x}_2 - x)^T] = E[\epsilon_2 {\epsilon_2}^T] \tag{3.55}$$

where ϵ_1 and ϵ_2 are the estimation error vectors in solution 1 and 2, respectively. The difference in the covariance between the two solutions is given by Lerch [60] as

$$\begin{aligned} P_{diff} &= E[((\hat{x}_1 - x) - (\hat{x}_2 - x))((\hat{x}_1 - x) - (\hat{x}_2 - x))^T] \\ &= E[(\epsilon_1 - \epsilon_2)(\epsilon_1 - \epsilon_2)^T] \\ &= E[(\epsilon_1 {\epsilon_1}^T - 2\epsilon_1 {\epsilon_2}^T + \epsilon_2 {\epsilon_2}^T)] \\ &= P_1 - 2P_{12} + P_2, \end{aligned} \tag{3.56}$$

where P_{12} represents the "cross-covariance" of the two solutions (and x is the true value of the state deviation vector). However, if the two solutions are each computed with completely different sets of observation data, the errors in the solutions are uncorrelated, simplifying Equation 3.56 to

$$P_{diff} = P_1 + P_2 \tag{3.57}$$

If the two solutions are not computed with completely different sets of observation data, i.e. solution 2 is a subset of solution 1, then $P_{12} = P_1$ and Equation 3.56 becomes

$$P_{diff} = P_2 - P_1. \tag{3.58}$$

Therefore, in either case, the differences in the computed geopotential coefficients of both solutions may be compared to the differences of their covariances. A scale

factor can be introduced to form a relationship to assist in the comparison of the solution difference to the covariance difference, and is written as

$$(\hat{x}_1 - \hat{x}_2)^T(\hat{x}_1 - \hat{x}_2) = \overline{k} \cdot \text{diag}(P_{diff}), \tag{3.59}$$

where $\overline{k}$ is a row vector containing the scale factors k_i, which correspond to each computed geopotential coefficient. In the event that all of these scale factors are close to unity, $k_i \approx 1$, then the observation data weights, that produced these solutions were appropriate. If this is not the case, then the weights are in need of adjustment.

3.3 Summary

This chapter presented three basic approaches for the gravity estimation process and discussed the issue of the weighting of observation data. The next chapter addresses the question of how the SVD is applied in the solution of an estimation problem and what techniques are used in this study to find the best gravity field. Then, a simple numerical example of the SVD is given to illustrate one simple approach.

Chapter 4

The Singular Value Decomposition (SVD)

The origin of the SVD can be traced back to the year 1873, when the Italian mathematician, Eugene Beltrami (who studied at the University of Pavia, in Italy, from 1853 to 1856) first published a paper containing a derivation of the singular value decomposition (SVD). He used the SVD in his attempt to explain bilinear forms. In 1874, Camile Jordan, a French engineer and author of 120 papers in mathematics, independently derived the SVD in an effort to reduce a bilinear form to a diagonal form by performing orthogonal substitutions. Later in 1907, the German Erhard Schmidt generalized the SVD concept to function space and discovered the approximation theorem using singular values. In 1913, Autonne extended the SVD to complex matrices. In 1936, Americans Eckart and Young extended the SVD to apply to rectangular matrices. Then in 1965, Golub and Kahan introduced the SVD into the field of numerical analysis with their proposed computational algorithm (Golub and W. Kahan [61]). Since the inception of the numerical computational SVD algorithm, it has been widely applied in many different fields of engineering science.

The SVD is not only useful in engineering but also in the areas of computer science, image processing, control, signal processing, as well in the fields of natural

science for example chemistry, geology, statistics, and medicine.

4.1 Formal Mathematical Definition of the SVD

When a set of linear equations inadequately describes a system, the matrix that is used to solve for a solution, will be singular or numerically close to singularity. Standard techniques such as Gaussian elimination or LU Decomposition will fail to give satisfactory results and the solution will be nonsensical. However, through the use of the SVD it is possible to discover where the problems are, eliminate them and arrive at a solution. It is with the help of the singular values of a matrix, that one can "diagnose" difficulties preventing solution and remove them. The SVD is based on the following theorem, where n is the number of unknown parameters and m is the number of observations.

Theorem 4.1.1. *If* $\mathbf{H}$ *is a real m x n matrix, where* $m \geq n$*, then there exist orthogonal matrices* $\mathbf{U} = [\mathbf{u_1}, ..., \mathbf{u_m}] \in \Re^{mxm}$ *and* $\mathbf{V} = [\mathbf{v_1}, ..., \mathbf{v_n}] \in \Re^{nxn}$ *such that* $\mathbf{U^T H V} = \mathbf{diag}(\sigma_\mathbf{1}, ..., \sigma_\mathbf{p}) \in \Re^{\mathbf{mxn}}$*, p = min(m,n), where* $\sigma_1 \geq \sigma_2 \geq \ldots \sigma_p \geq 0$.

The σ_i's are the *singular values* of $\mathbf{H}$ and the column vectors $\mathbf{u_i}$ and $\mathbf{v_i}$ are the *i*th *left singular vector* and the *i*th *right singular vector*, respectively. $\mathbf{V}$ is both row and column orthogonal such that $\mathbf{V^T V} = \mathbf{V V^T} = \mathbf{I_n}$ and $\mathbf{U}$ is column orthogonal such that $\mathbf{U^T U} = \mathbf{I_m}$ (Golub [23]). The singular values of matrix H represent the lengths of a semi-axes of the hyper ellipsoid defined by $\{\mathbf{Hx} : \|\mathbf{x}\| = 1\}$ (Belsley [62]). The SVD of H may be expressed, where the rank(H) = n (that is H is full

rank), as

$$\mathbf{H} = \mathbf{U_m S_n V_n}^{\mathbf{T}} = \sum_{\mathbf{i=1}}^{\mathbf{n}} \sigma_{\mathbf{i}} \mathbf{u_i v_i}^{\mathbf{T}}. \tag{4.1}$$

Notice that each σ_i in Equation 4.1 is a scalar, allowing them to be moved out from between the vectors **u** and **v** in the summation. Portraying this decomposition in another way, one can see the dimensions or shapes of the matrices clearly. The square m-by-m matrix U, consists of the left singular $\mathbf{u}_i$ vectors, rectangular m-by-n matrix Σ, contains the singular values in the top square n-by-n matrix, where all other matrix entries are zero, and the square n-by-n matrix V contains the right singular vectors $\mathbf{v}_i$,

$$H = \begin{bmatrix} & & & \\ u_1 & \cdots & \cdots & u_m \\ & & & \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \\ & 0 & \end{bmatrix} \begin{bmatrix} & & \\ v_1 & \cdots & v_n \\ & & \end{bmatrix}. \tag{4.2}$$

If matrix H were square to begin with, then all matrix dimension in Equation 4.2 would be equal. It is important to mention this matrix geometry, as it is helpful to understand how the SVD is applied to the normal matrix in the normal least squares problem or in the orthogonal transformation.

4.1.1 The SVD and the Least Squares Problem

In the linearized least squares problem, the state deviation vector x of dimension n, is related to the observation residual vector y, of dimension m, in the following equation, through the so called information matrix H,

$$y = Hx + \epsilon. \tag{4.3}$$

H is m-by-n and is also called the observation-state matrix while ϵ is a an error vector of dimension m. Vector ϵ represents the errors of omission and commission in the mathematical models involved in an estimation problem as well as the observation errors. While it is assumed that only the observation errors are random, this error vector is considered to be random with zero mean, $E[\epsilon] = 0$, with a covariance equal to the identity matrix, $E\left[\epsilon\right]\left[\epsilon\right]^T = I$, where I is an m-by-m.

Now suppose H is rank deficient, that is rank(H)$= k < n$. This means that matrix H is *singular* or ill-conditioned. Such a situation can be caused by two scenarios. There are fewer than n independent columns or rows in matrix H, such that the attempt to solve for n parameters in the system is impossible. Or the dimensions m and n are sufficiently large, the problem is that not enough equations in the linearized system, which manifest themselves as rows in matrix H, are linearly independent enough to resolve certain "*trouble*" parameters apart from one another during the solution, or "inversion" process. In other words, one or more rows in matrix H are very similar to each other, in that they are almost a multiple of one another, namely they *point* in very similar directions, i.e. they are close to being *collinear*. Therefore, they are not observing these parameters and their information content (in a collinear row) is too weak to return separate estimation values for them.

Regardless of the situation, the SVD of H may be written as

$$H = USV^T \tag{4.4}$$

where matrix U, called the *Range Space* of matrix H, contains the left singular

vectors, and matrix V, called the *Null Space* of matrix H, contains the right singular vectors of matrix H. The matrix S contains the singular values of H in its upper square portion, along the diagonal, as seen in Equation 4.2. Before we continue with the discussion of the SVD of H, it is useful to explain what a *Range Space* and a *Null Space* is. In general, a matrix is a linear operator which transforms a vector from one vector space into another.

To illustrate this concept, let the linear operator, matrix $A \in \Re^{mxn}$, $w \in \Re^n$ and $q \in \Re^m$. If, for example, A is full rank, then it can transform the vector w, which exists in the space of dimension n, into a space of dimension m, namely the vector q,

$$Aw = q. \tag{4.5}$$

If this transformation is successful, then the range of A is "full" so that all of w is transformed in to the space of q, and the vector q is therefore of dimension m-by-1. Now if this transformation is unsuccessful, that means the range of A is not "full", and can not reach all of the space of q. Therefore after the unsuccessful transformation, the vector q is not of dimension m-by-1, rather it is something less, i.e. (m-p)-by-1. The "missing" portion of vector q, let us call it vector s, is of dimension p-by-1. Vector s is called the null space vector. Similarly, the vector q is called the range space vector. Both vectors are produced by their own linear operators and provide for the transformation of the w vector into these two distinct vector spaces.

These linear operators are called the Range Space (Matrix U) and the Null Space (Matrix V), as seen in Theorem 4.1.1, where the former is the basis space

for the transformation of vector w into q, and the latter is the basis space for the transformation of vector w into s. It can also be seen from Theorem 4.1.1, that the column vectors in U and V are what span these spaces. Now that these important concepts have been introduced, discussion of Equation 4.4 and the application of the SVD on H may be continued. The matrix S can be written as

$$S = \begin{bmatrix} S_k & 0 \\ 0 & 0 \end{bmatrix} \tag{4.6}$$

where S_k is the diagonal matrix of the non-zero portion of the singular values of H. Notice that k equals the rank of matrix H, which is less than its column dimension n. Now, in the least squares problem, the goal is to minimize the Euclidean norm of the residuals. In terms of the SVD this is written as

$$\rho = \|y - Hx\| = \|U^T y - U^T \left[USV^T\right] x\| = \|a - Sz\| \tag{4.7}$$

where $a = U^T y$ and $z = V^T x$. The vector **a** and **v** may be rewritten as

$$a = \begin{bmatrix} a_k \\ a_{m-k} \end{bmatrix} \quad \text{and} \quad z = \begin{bmatrix} z_k \\ z_{m-k} \end{bmatrix}. \tag{4.8}$$

Multiplying S by **z**, we see that

$$Sz = \begin{bmatrix} S_k & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} z_k \\ z_{m-k} \end{bmatrix} \tag{4.9}$$

leads to

$$Sz = \begin{bmatrix} S_k z_k \\ 0 \end{bmatrix}. \tag{4.10}$$

After substituting equation 4.8 and 4.10 into 4.7 we obtain

$$\begin{aligned} \rho &= \|a - Sz\| = \left\| \begin{bmatrix} a_k \\ a_{m-k} \end{bmatrix} - \begin{bmatrix} S_k z_k \\ 0 \end{bmatrix} \right\| = \left\| \begin{array}{c} a_k - S_k z_k \\ a_{m-k} \end{array} \right\| \\ &= \|a_k - S_k z_k\| + \|a_{m-k}\|. \end{aligned} \tag{4.11}$$

Thus by setting $a_k - S_k z_k = 0$, $z_k = {S_k}^{-1} a_k$, ρ is minimized. Since the value of a_{m-k} has no effect on ρ it can be set to any value. By choosing $a_{m-k} = 0$, it produces a minimum norm of all solutions for x. Now let us find the expression for this x. From Equation 4.7, and the fact that V is an orthogonal matrix, it can be seen that $\mathbf{x} = V\mathbf{z}$. This means that the norm of x is equivalent to the norm of z. Vector z can be written as

$$z = \begin{bmatrix} z_k \\ 0 \end{bmatrix} = \begin{bmatrix} {S_k}^{-1} a_k \\ 0 \end{bmatrix} = \begin{bmatrix} {S_k}^{-1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} a_k \\ a_{m-k} \end{bmatrix} = {S_k}^{-1} a_k. \tag{4.12}$$

By substituting this expression for **z** into $x = Vz$ and that for vector **a**, leads to the estimate $\hat{x}$

$$\hat{x} = V {S_k}^{-1} a_k = V {S_k}^{-1} U^T y. \tag{4.13}$$

The term ${S_k}^{-1}$ is referred to the pseudo-inverse S^+ of S (Golub [23]). The expected value of $\hat{x}$ is

$$\begin{aligned} E[\hat{x}] &= E\left[V {S_k}^{+} U^T y\right] = E\left[V {S_k}^{+} U^T (Hx + \epsilon)\right] \\ &= V {S_k}^{-1} U^T H E[x] = V S^{+} U^T U S V^T x = V S^{+} S V^T x. \end{aligned} \tag{4.14}$$

Notice that,

$$S^{+} S = \begin{bmatrix} {S_k}^{-1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} S_k & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} I_k & 0 \\ 0 & 0 \end{bmatrix}. \tag{4.15}$$

This leads us to the following claim, that the estimate obtained from the SVD is not unbiased. That is, it produces a biased estimate, if any singular values are disposed of, causing zeros to appear in the lower right quadrant of the matrix $S^{+}S$ defined in Equation 4.15. Letting

$$V = \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix} \tag{4.16}$$

and carrying out the next equation shows that one does not get the Identity matrix when $k \neq n$.

$$V \begin{bmatrix} I_k & 0 \\ 0 & 0 \end{bmatrix} V^T = \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix} \begin{bmatrix} I_k & 0 \\ 0 & 0 \end{bmatrix} V^T = \begin{bmatrix} V_{11} & 0 \\ V_{21} & 0 \end{bmatrix} V^T \neq I_n. \tag{4.17}$$

Therefore, substituting the result of 4.17 into Equation 4.14, the expectation of $\hat{x}$ is

$$E[\hat{x}] = VS^+SV^Tx = \begin{bmatrix} V_{11} & 0 \\ V_{21} & 0 \end{bmatrix} V^Tx \neq x \quad \text{for} \quad k \neq n. \tag{4.18}$$

Thus an SVD estimate is not unbiased if any singular values are set to zero. If, however, all singular values are retained, then k = n and $S^+S = I_n$ leading us to an unbiased estimate. The covariance matrix P for the SVD can be derived as follows.

$$\begin{aligned} P &= E\left[(\hat{x} - E[\hat{x}])(\hat{x} - E[\hat{x}])^T\right] \\ &= E\left[\left(VS^+U^Ty - VS^+U^THx\right)\left(VS^+U^Ty - VS^+U^THx\right)^T\right] \\ &= VS^+U^TE\left[(y - Hx)(y - Hx)^T\right] US^{+T}V^T \\ &= VS^+U^TE\left[\epsilon\epsilon^T\right] US^{+T}V^T. \end{aligned} \tag{4.19}$$

Because $E\left[\epsilon\epsilon^T\right] = I_n$, Equation 4.19 is simplified to the form

$$P = VS^+S^{+T}V^T = V \begin{bmatrix} {S_k}^{-2} & 0 \\ 0 & 0 \end{bmatrix} V^T. \tag{4.20}$$

The covariance matrix of Equation 4.20 represents only the random errors in the estimation process. The bias, arising from the disposal of singular values (caused by setting them to zero), is not included. Therefore, this covariance matrix is not a complete representation of the error (Lerch [29]). Something very important must also be mentioned about Equation 4.20. In the case of the normal equations

in the least squares estimation process, it is the eigenvalues that appear along the diagonal of matrix $S^{+}S^{+^T}$ in Equation 4.20, because each singular value ends up being squared along this diagonal. An eigenvalue λ_i is the square of a singular value σ_i, i.e. $\lambda_i = {\sigma_i}^2$. Thus, the exponent on S_k should be set to minus one in Equation 4.20, when actually calculating the covariance matrix P of the normal equation.

4.1.2 The SVD and the Orthogonal Transformation

The performance index is the same as seen in the previous chapter and matrix R comes from the orthogonalization of matrix H.

$$\begin{aligned} J &= \|QW^{\frac{1}{2}}(y - Hx)\| \\ &= \|b - Rx\| + \|e\| = \|b - USV^T x\| + \|e\| \\ &= \|U^T\left(b - USV^T x\right)\| + \|e\| = \|\tilde{b} - SV^T x\| + \|e\| \\ &= \|\tilde{b} - Sz\| + \|e\| \end{aligned} \tag{4.21}$$

where $R = USV^T$, $\tilde{b} = U^T b$ and $z = V^T x$. For a full rank, well conditioned matrix R, the performance index is minimized when $\|b - Rx\| = 0$ or $Rx = b$. Then the state deviation vector $x = R^{-1}b$ is obtained by backwards substitution. It can also be obtained by using the SVD, where $S = S^{+} = S^{k}$, and k = n as seen in Equation 4.13. If R is not full rank (ill-conditioned), then the performance index is minimized when $\|\tilde{b} - Sz\| = 0$ or $Sz = \tilde{b}$ and zero (or near zero) singular values on the diagonal of matrix S are discarded to obtain a non-unique solution, $x = VS^{+}U^T b$. The number of discarded singular values is n-k. During solution, the reciprocals of each singular value is required. If one of them is zero, then its

inverse will be infinity causing a machine overflow. To avoid this problem, a zero singular value is discarded by setting it to zero after inversion. In other words we set infinity to zero! (Important note: In actual implementation, the inverse of zero, or near zero, is simply avoided and its "inverse value" is simply assigned the value of zero.) The performance index in this case is

$$J = \|\tilde{b}_k - S_k z_k\| + \|\tilde{b}_{n-k}\| + \|e\|. \tag{4.22}$$

(In comparison to the numerical rank solution, which includes all singular values, better gravity solutions may be obtained by discarding non-zero singular values.) Equation 4.22 represents the performance index in which k singular values are *retained* and n-k are thrown away. Notice the introduction of the bias term $\|\tilde{b}_{n-k}\|$. It represents the information which is harmful or at best useless to the estimated solution. The disposal of the n-k singular values results in a biased estimate. The size of this bias and how it compares to the random only portion of the solution is a topic of a later section.

Often some singular values appearing in matrix S are very close to zero and there is a smooth transition from them to very large values. If a value is "very small", during the inversion process, the reciprocal of this number might cause the inflation of the correction to an estimated parameter. If the singular value is "large" then it is not apt to do this. Exactly which non-zero singular values in matrix S should be discarded to achieve the best solution, is the subject of discussion after the next section.

4.1.3 The SVD Bias

Bias has been almost completely ignored in the literature of geophysical and geodetic inversions (Rummel [63]). In the previous section, the bias resulting from an SVD (inversion) solution was identified as $\|\tilde{b}_{n-k}\|$. For simplicity, let us rename this vector $\tilde{b}$. To understand better what the bias vector and a biased estimate is, let us review the meaning of an unbiased estimate.

If multiple unbiased estimations are performed using different data, the scatter of estimates (each one from a different data set) would be centered about the true state X. Using the expected value operator E[ ], the mean of the unbiased updated state is,

$$\begin{aligned} E[\hat{X}] &= E[X^* + \hat{x}] \\ &= X^* + E[\hat{x}] \\ &= X^* + x \\ &= X \end{aligned} \tag{4.23}$$

where X and x are the true state and true deviation, respectively, and $\hat{x}$ is the estimated correction. Thus, the covariance of this solution is

$$\begin{aligned} P_{\hat{X}} &= E[(\hat{X} - E[\hat{X}])(\hat{X} - E[\hat{X}])^T] \\ &= E[(\hat{X} - X)(\hat{X} - X)^T], \end{aligned} \tag{4.24}$$

where $E[\hat{X}] = X$ (the true state) and

$$\begin{aligned} \hat{X} - X &= X^* + \hat{x} - X \\ &= X^* + \hat{x} - (X^* + x) \\ &= \hat{x} - x. \end{aligned} \tag{4.25}$$

This allows us to conclude that,

$$\begin{aligned} P_{\hat{X}} &= E[(\hat{x} - x)(\hat{x} - x)^T] \\ &= E[(\hat{x} - E[\hat{x}])(\hat{x} - E[\hat{x}])^T] \\ &= P_{\hat{x}}. \end{aligned} \tag{4.26}$$

In other words, the covariance of the updated state is the same as the covariance of the correction update vector $\hat{x}$. Stated in another way, the scatter of multiple estimates (each using different data) of $\hat{X}$ about a mean of X (true state) has the same scatter as multiple estimates of $\hat{x}$ about a mean of $\hat{x}$ = x (true deviation). If we were to draw error hyper- ellipsoids for $\hat{X}$, they would all be exactly of the same shape and orientation as those error hyper-ellipsoids for $\hat{x}$. (Note: a hyper-ellipsoid is an error ellipsoid for more than 3 dimensions.)

If the error in $\hat{x}$ is defined as $\tilde{e} \equiv \hat{x} - x$, the scatter of the error has a mean

of zero.

$$\begin{aligned} E[\tilde{e}] &= E[\hat{x} - x] \\ &= E[\hat{x}] - E[x] \\ &= x - x \\ &= 0 \end{aligned} \tag{4.27}$$

Therefore the covariance matrix of $\tilde{e}$ is the same as the covariance matrix of $\hat{x}$, namely

$$\begin{aligned} P_{\tilde{e}} &= E[(\tilde{e} - E[\tilde{e}])(\tilde{e} - E[\tilde{e}])^T] \\ &= E[(\tilde{e})(\tilde{e})^T] \\ &= E[(\hat{x} - x)(\hat{x} - x)^T] \\ &= E[(\hat{x} - E[\hat{x}])(\hat{x} - E[\hat{x}])^T] = P_{\hat{x}}. \end{aligned} \tag{4.28}$$

Thus, for an unbiased estimate, $P_{\tilde{e}} = P_{\hat{x}} = P_{\hat{X}}$. Figure 4.1 illustrates this concept. A two dimensional (2x1) vector estimate, has the error ellipses for $\hat{x}$ centered on x, the true deviation. It has the same error ellipses for $\tilde{e}$, centered on zero.

Now we are in a position to better understand what the bias vector and a biased estimate is. Usually the term "true bias" $\tilde{b}$ is an unknown and represents the difference between the expected value of the estimated state $\hat{X}$ and the true state vector X,

$$\tilde{b} = E[\hat{X}_{biased}] - X. \tag{4.29}$$

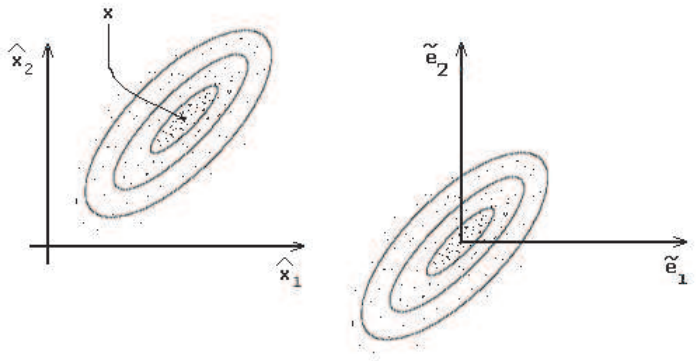


Figure 4.1: 2-Dim. Error Ellipses for Deviation and Error

Figure 4.2 displays this scenario.

The term $\tilde{b}$ is also equal to the difference between the expected value of $\hat{x}$ and the true *deviation* x. Equation 4.30 expresses this relation as,

$$\tilde{b} = E[\hat{x}_{biased}] - x. \tag{4.30}$$

Whether the estimate is biased or not, the state estimate is defined as

$$\hat{X} = X^* + \hat{x}, \tag{4.31}$$

where X^* is the nominal and $\hat{x}$ is the correction.

Notice that Figure 4.2 also illustrates the fact that (in two dimensions) the error ellipses for a biased estimate are not only centered about a different expected value, but that the ellipses themselves are of a different shape, size and orientation. For the case where n $>$ 3, the resulting biased error hyper-ellipsoids would also be centered about a different location (given by $E[\hat{X}_{biased}]$) and have different shapes and orientation in comparison to those of the solutions from an unbiased estimate.

An expression for the biased estimate is given (analogously to the unbiased case) as

$$\begin{aligned}
\hat{X}_{biased} - E[\hat{X}_{biased}] &= (X^* + \hat{x}_{biased}) - (X + \tilde{b}) \\
&= (X^* + \hat{x}_{biased}) - (X^* + x + \tilde{b}) \\
&= \hat{x}_{biased} - (x + \tilde{b}) \\
&= \hat{x}_{biased} - E[\hat{x}_{biased}],
\end{aligned} \tag{4.32}$$

where $E[\hat{x}_{biased}]$ is also seen in Equation 4.30. The covariance of this biased estimate is then written as

$$\begin{aligned}
P_{\hat{X}_{biased}} &= E[(\hat{X}_{biased} - E[\hat{X}_{biased}])(\hat{X}_{biased} - E[\hat{X}_{biased}])^T] \\
&= E[(\hat{x}_{biased} - E[\hat{x}_{biased}])(\hat{x}_{biased} - E[\hat{x}_{biased}])^T] \\
&= P_{\hat{x}_{biased}}.
\end{aligned} \tag{4.33}$$

Equation 4.33 shows that the covariance of the updated biased state is the same as the covariance of the biased correction update vector.

If the error in $\hat{x}_{biased}$ is defined as $\tilde{e}_{biased} \equiv \hat{x}_{biased} - x$, the scatter of the biased error has its mean centered on the bias (not zero), where $x = x_{true}$.

$$\begin{aligned}
E[\tilde{e}_{biased}] &= E[\hat{x}_{biased} - x] \\
&= E[\hat{x}_{biased}] - E[x] \\
&= x_{biased} - x \\
&= \tilde{b},
\end{aligned} \tag{4.34}$$

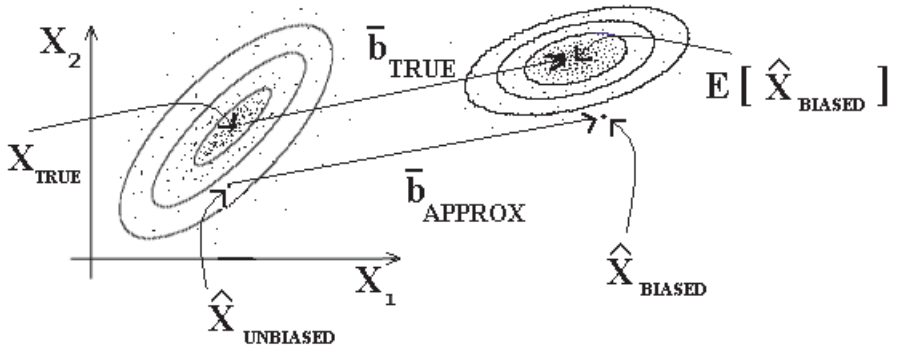


Figure 4.2: 2-Dim. Error Ellipses for the SVD Biased Estimate

where $\tilde{b}$ is also defined in Equation 4.30. (Important note: None of the Equations 4.23 through 4.34 can/will in practice provide no evidence *whatsoever* of a bias in the estimate. These equations convey absolutely no knowledge of the existence of $\tilde{b}$. Implementing/Calculating Equation 4.18 provides the proof that a bias exists. Calculating that bias vector is explained below.)

Unfortunately, because the "true bias" $\tilde{b}$ ($\bar{b}_{TRUE}$ in Figure 4.2) is an unknown (because the true state is unknown), one is forced to approximate $\bar{b}_{TRUE}$. This is done by taking the difference between one value of $\hat{X}_{UNBIASED}$ and one value of the SVD estimated $\hat{X}_{BIASED}$, to give the approximate bias vector $\bar{b}_{APPROX}$. The term $\hat{X}_{UNBIASED}$ is assigned to be an accepted standard estimate, such as the Texas Earth Gravity 4 (TEG4) solution. Figure 4.2 illustrates this concept in two dimensions. The true bias vector is the exact difference between the expected value of $\hat{X}_{BIASED}$ (the SVD estimate) and X_{TRUE}. Figure 4.2 also shows that if both the

biased and unbiased solutions are some "small" distance away from their expected values (the center of their respective scatters), then the approximated bias vector $\overline{b}_{APPROX}$ will be approximately equal to the true bias.

One method of quantifying/measuring the proximity of a biased solution to that of an unbiased, is to use this approximated bias vector,

$$\overline{b}_{APPROX} = \hat{X}_{BIASED} - \hat{X}_{UNBIASED}, \tag{4.35}$$

in comparing the trace of the biased covariance matrix $P_{\hat{x}_{biased}}$ (which contains random error/uncertainty only) to the trace of $\overline{b}_{APPROX}\overline{b}_{APPROX}^{T}$ (Bouman [64]). If the ratio of the latter to the former is much less than unity, then the biased solution may be considered as unbiased in an approximate sense. Since this metric is used to assess the bias due to ridge regression or a Tikhonov regularization with signal constraint (Bouman [64]), it will be used to evaluate the bias in the SVD solutions of this investigation. To compute the bias *vector* $\overline{b}_{APPROX}$, true coefficients are needed. For this purpose, solutions from EIGEN1S (Reigber [65]), TEG4 (Tapley [66]) and EGM96 [67] are used for the cases of CHAMP and GRACE.

4.1.4 Stabilization and Optimization of Solution

The reason why singular values in an ill-conditioned system of linear equations are discarded, is because keeping all of them results in a solution in which some poorly observed parameters (elements of the state deviation correction vector), become "inflated", meaning that they become too large or nonsensical. This is an indication that there is not enough information in the observations to resolve the

parameters apart from one another, specifically those which are suffering the inflation. To “deflate” these estimated parameters or to stabilize their values, singular values, which contribute to their instability during the estimation process, must be discarded, and they may be thrown away based upon subjective or objective criteria. For a large gravity field problem, high degree and order coefficients may be poorly observed, leading to an inflation in their estimated values. By discarding singular values it is possible to stabilize or to “deflate” these parameters so that reasonable solutions can be obtained. Because this leads to many candidate solutions, it is desirable to search for the optimal solution. The next section describes some of these optimization techniques.

4.1.4.1 Inspection of Singular Values

The rank of a matrix determined by performing the SVD, or any other numerical method, is referred to as the numerical rank. The rank of a matrix is the number of singular values that are larger than zero. The most simplest method to find the zeros is to plot all singular values and inspect them. Any located zero value is then discarded during the solution process. However, it is often the case that many singular values are very “close” to zero and the other computed singular values are smoothly dispersed up to some maximum value. Unfortunately, numerical roundoff may cause an otherwise zero value to result in a very small value, or visa versa. Therefore a choice must be made as to which singular values will be kept or discarded, and this choice is often not obvious. The subjective method of inspection may be used a good starting point in the search for the optimal pseudo inverse.

4.1.4.2 Relative Error

Using a smaller rank matrix with k singular values to approximate a matrix with n singular values, represents a forfeiture of n-k singular values and the information they carry. One way to describe this loss of data is termed "relative error" [30], and is represented in the following equation.

$$e\left(k\right) = 1 - \sqrt{\frac{\sum_{i=1}^{k} s_i^2}{\sum_{i=1}^{n} s_i^2}}, \tag{4.36}$$

where each s_i is a singular value. Notice that e(k) ranges from zero to one as a function of the integer k. As k approaches n, then more and more information is retained. Thus, when k = n, no matrix information is discarded, giving a value of 0 relative error. As k approaches zero, the opposite trend is observed. Thus, when k = 0, all matrix information is gone, yielding a relative error of unity. Of course the magnitude of singular values has a profound effect on relative error as to their inclusion or exclusion. Consequently, dropping the very small singular values of a near singular matrix does not significantly affect the relative error. However, dropping the very large singular values, delivers considerable effect. Therefore if a value of relative error is chosen as an objective, then the corresponding numerical rank may be determined using Equation 4.36.

4.1.4.3 Norm-Norm Plot

Since the Norm-Norm Plot is the combination of two different graphs of the "norm of the solution" and the "norm of the residual", each should be described separately. Although both are dependent on the matrix-vector equation $\widetilde{A}\hat{x} = \overline{b}$,

each exhibits distinct behaviors as a function of the number of singular values used in an SVD solution. The "norm of the solution" $\|\hat{x}\|_2$ is defined as

$$\|\hat{x}\|_2 = \|\widetilde{A}^{-1}\overline{b}\|_2, \tag{4.37}$$

and the "norm of the residual" $\|\overline{\rho}\|_2$, is known as

$$\|\overline{\rho}\|_2 = \| \widetilde{A}\hat{x} - \overline{b} \|_2. \tag{4.38}$$

Every point in the plots generated by Equations 4.37 and 4.38, represents a different solution to the linear equation of $\widetilde{A}\hat{x} = \overline{b}$. Thus, it is possible to search for and find an optimal SVD solution among the many candidates afforded by these equations. The norm of the solution vector is directly proportional to the number of included singular values used in an SVD solution, while there is an indirect proportion between the norm of the residual vector and the singular values. These two norms can be merged together in a so called "norm-norm" plot, by plotting the norm of the residual vector versus the norm of the solution vector, see Figure 6.38. Such norm-norm plots can be seen in Ehrig and Nowak [12] and in the problem of spacecraft navigation at the Jet Propulsion Laboratory (JPL) McCord [25]. In the left region of the norm-norm plot, fewer singular values are retained in the SVD solution. The residuals will be larger due to the smaller contributions of the correction vector elements to the a priori state vector. As more and more singular values are retained, the correction vector becomes larger while its elements contribute more and more to the state vector, leading to an overall decrease in residuals. This trend continues until all singular values are retained, causing the norm of the solution to

be large compared to the norm of the residual. It is the point on this norm-norm curve, that is closest to the origin where we find the optimal balance between these two norms. It is the point where the square root of the sum of the squares of the elements of the residual vector, ρ and solution vector $\hat{x}$ are a minimum.

Before the parallel implementation of the SVD in this investigation, the main disadvantage and difficulty of generating these norm norm plots was the amount of time needed to compute each SVD solution and their norms (Ahn [30]). However, implementing newly developed code ((**P**arallel **LA**rge **S**vd **S**olver) **PLASS**), whose methods are discussed in Chapter 5, this obstacle has been eliminated through the use of multiprocessing. Using 32 processors, for example, **PLASS**2 can generate very easily 3,000 different solutions of a GRACE gravity field of over 25,000 parameters in less than 8 hours on the current computer system at NASA Ames. Such computation power allows for a fine grained sweep of all singular values to be performed so that interesting regions in the norm-norm plot may be investigated further. Since a norm-norm plot involves only the norms of the correction vector of a given system, its optimal solution may not correspond to an optimal *gravity* solution. The correction vector must be added to the nominal gravity field (called the nominal vector), before an adequate evaluation can be completed. Example of such assessments are the orbital arc fit and the geoid test and are discussed in a later section.

4.1.4.4 Mean Square Error (MSE)

The MSE analysis can provide both an approximation of the bias introduced by the elimination of singular values during an SVD solution and the optimal number of them to retain. The mean square error is the sum of the bias and noise for an estimate. Using the Equations 4.7, 4.8 and remembering that $z = V^T x$, we can let the bias vector due to the removal of singular values, be defined in z space. The bias vector is then written as (Lerch [29])

$$\beta = z - E[\hat{z}] = \begin{bmatrix} z_k \\ z_{n-k} \end{bmatrix} - E\begin{bmatrix} z_k \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ z_{n-k} \end{bmatrix} \quad \text{where} \quad \hat{z} = \begin{bmatrix} z_k \\ 0 \end{bmatrix}. \tag{4.39}$$

If it is desired, the covariance matrix of the estimate z may be derived as

$$\begin{aligned} P &= E\left[(\hat{z} - E[\hat{z}])(\hat{z} - E[\hat{z}])^T\right] \\ &= E\left[\left(S^+U^Ty - S^+U^THx\right)\left(S^+U^Ty - S^+U^THx\right)^T\right] \\ &= S^+U^TE\left[(y - Hx)(y - Hx)^T\right]US^{+T} \\ &= S^+U^TE\left[\epsilon\epsilon^T\right]US^{+T} \\ &= \begin{bmatrix} {S_k}^{-2} & 0 \\ 0 & 0 \end{bmatrix} \end{aligned} \tag{4.40}$$

According to Lerch [29], the trace of the mean square error matrix in terms of z, can be derived as a function of the integer k

$$\begin{aligned} \text{trace}[MSE(\hat{X})] &= E[(X - \hat{X})^T(X - \hat{X})] \\ &= E[(z - \hat{z})^T(z - \hat{z})] \\ &= \sum_{i=1}^{k} S_i^{-2} + \sum_{i=k+1}^{n} z_i^2. \end{aligned} \tag{4.41}$$

The index k is the number of retained singular values in the SVD solution. $\hat{X}$ is the solution vector, which is a combination of the nominal and the state deviation

vectors, namely $\hat{X} = X^* + \hat{x}$, where $\hat{x} = V\hat{z}$. The last relation in Equation 4.41 contains the addition of a noise component and a bias component, respectively. It can be seen that the noise contribution increases with k while the bias decreases with k. Hence, the optimum k is one that corresponds to a noise-bias combination that minimizes the scalar trace$[MSE(\hat{X})]$. Unfortunately, the true values of z_i which depend on $\hat{x}$, are not available. However, they can be approximated by using Kaula's rule allowing a realistic definition of the bias term in Equation 4.41. The degree variances of coefficients in the spherical harmonic expansion that represent Earth's gravitational potential, have a magnitude that can be approximated by the formula:

$$\sigma^2 = \frac{10^{-10}(2\ell + 1)}{\ell^4}, \tag{4.42}$$

where ℓ is geo-potential degree.

Using the fact that $x = Vz$, z_i can be approximated as

$$z^2 \approx \text{diag} \left\{ V_T \begin{bmatrix} {\sigma_1}^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & {\sigma_m}^2 \end{bmatrix} V \right\} \tag{4.43}$$

where z^2 is a vector whose elements are each z_i and m is the maximum geo-potential degree of the gravity model. It is important to say that the dimension of all matrices seen in Equation 4.43 are n-by-n, and the order in which the spherical harmonic coefficients appear on the diagonal is determined by the so called parameter name-list vector. This is an important vector which describes the order in which the parameters of the state deviation vector appear. For a given gravity field problem, the MSE method finds a unique value for k. Although this value may

be based on an approximation, it nevertheless provides a good solution that can be compared to the others obtained by different techniques.

4.1.4.5 The Kaula-Singular Value (KSV) Relation

One goal of this investigation is to relate the disposal of singular values, in the SVD stabilization of gravity field solutions, to Kaula's rule seen in Equation 4.42. Since an SVD solution is affected by which singular values are included in gravity field calculations, there must exist a relation between singular values and the equations of gravity field estimation. A gravity field solution is an estimated parameter vector, whose elements are the scaled dimensionless coefficients, $C_{l,m}$ and $S_{l,m}$. These parameters are the constants that are multiplied against the basis functions appearing in the spherical harmonic expansion, which is used in the equation to describe a three dimensional gravitational potential, U, in the free space (zero density) above the Earth. That potential equation is given by Tapley, Born, Schutz [45] as

$$\begin{aligned} U &= \frac{GM}{r} + U' \\ U' &= -\frac{GM^*}{r}\sum_{l=1}^{\infty}\left(\frac{a_e}{r}\right)^l P_l(\sin\phi)J_l \\ &+ \frac{GM^*}{r}\sum_{l=1}^{\infty}\sum_{m=1}^{l}\left(\frac{a_e}{r}\right)^l P_{l,m}(\sin\phi)[C_{l,m}\cos m\lambda + S_{l,m}\sin m\lambda], \end{aligned} \quad (4.44)$$

The relationship between singular values and these spherical harmonic coefficients is revealed through the use of the equation called "degree variance",

$$\sigma_l^2 = \sum_{m=0}^{l}(C_{l,m}^2 + S_{l,m}^2). \quad (4.45)$$

Thus, all coefficients for a particular harmonic degree ℓ are squared and summed into a single scalar, a scalar which is closely approximated by Kaula's rule.

Two new "vectors" (actually one-dimensional arrays) can be constructed by using Kaula's rule and the estimated coefficients of the SVD solution. Let us call these vectors the "Kaula vector", $\overline{v}_{Kaula}$, and the "svd-vector", $\overline{v}_{svd(k)}$, where k is the number of singular values retained in an SVD solution. The ordering of the elements for both vectors are identical and are based upon the sequence of the estimated coefficients in the SVD solution, i.e. the ordering in which degree "l" and order "m" appear in the estimation, are repeated and synchronized between corresponding elements of these two vectors. Each element of $\overline{v}_{Kaula}$ is defined by Kaula's rule and each element of $\overline{v}_{svd(k)}$ is defined by the σ_l^2 seen in Equation 4.45, which is based on the SVD solution. By populating both vectors in this manner and comparing their elements, a relation between Kaula's rule and singular value disposal may be discovered. By taking the two-norm of the difference of these two vectors, we obtain the scalar,

$$\alpha(k) = \left\| \overline{v}_{svd(k)} - \overline{v}_{Kaula} \right\|_2. \tag{4.46}$$

This is equivalent to taking the square root of the sum of the squares of the differences among the elements of the vectors. The following equation illustrates this.

$$\alpha(k) = \left\{ \sum_{l=1}^{lmax} \left[v(l)_{svd(k)_{l,m}} - v(l)_{Kaula} \right]^2 \right\}^{\frac{1}{2}}, \tag{4.47}$$

where,

$$v(l)_{svd(k)_{l,m}} = \left[\sum_{m=0}^{l} (C_{l,m}^2 + S_{l,m}^2) \right], \tag{4.48}$$

and

$$v(l)_{Kaula} = \left\{ \frac{10^{-10}(2l+1)}{l^4} \right\}. \tag{4.49}$$

(The calculated ith or "ℓth" element of $\overline{v}_{svd(k)}$, $v(l)_{svd(k)_{l,m}}$ and the calculated "ℓth" element of $\overline{v}_{Kaula}$, $v(l)_{Kaula}$ is controlled by whatever degree and order is specified by the parameter name-list vector.) By constructing this vector pair for each new SVD solution, according to each new combination of singular values, the behavior of the dimensionless scalar $\alpha(k)$ may be plotted against k, the number of singular values retained in a solution. Thus a function relating Kaula's rule to singular value disposal may be graphed. It is the minimum of this curve that corresponds to the optimal choice of singular values, for the gravity solution that best satisfies Kaula's rule. By sweeping through many solutions, using **PLASS**2, this optimum can be quickly found. The corresponding gravity field can then be submitted for an orbital fit or geoid analysis.

4.1.4.6 Orbit Fit Computations and the Geoid Test

An orbit fit is a comparison of actual satellite observations to the simulated observations which are generated by the simulated movement of the same satellite through an estimated gravity field. The estimated gravity field from an SVD solution of either an information array or a normal matrix, can be evaluated by fitting various satellite orbits using UTOPIA (University of Texas Orbit determination Program) (Tapley et al.[68]) and calculating the RMS of the data residuals. Although this technique is expensive, it provides a good evaluation of a candidate gravity solution. The geoid test is another assessment method in which an SVD gravity

solution is used to produce an Earth geoid and is compared to that of an accepted standard. The differences between them are then inspected and quantified for objective criticism. Both solution assessment techniques are conducted only after the other, less expensive five methodologies, discussed in this chapter, have produced their optimal candidate solutions.

4.1.4.7 Simple Numerical Example of the SVD

This section is intended to show the results of the SVD method performed on a simple case of the Least Squares problem, $\widetilde{H}\overline{x} = \overline{b}$. In the first test, the information matrix $\widetilde{H}$, is kept in its original rectangular m-by-n form ($m = 12, n = 6$), whereas in the second test, the normal matrix, $\tilde{H}^T\tilde{H}$, is formed before the SVD is applied. Both sets of singular values are tabulated, and it can be seen that the nonzero singular values of $\tilde{H}^T\tilde{H}$ are the square of those of $\widetilde{H}$. After solving the equation $\widetilde{H}\overline{x} = \overline{b}$, the norm of the solution and norm of the residual are defined as $\|\overline{x}\|$ and $\|\overline{\rho}\| = \|\widetilde{H}\overline{x} - \overline{b}\|$, respectively and are listed for each case of singular value disposal, seen in Tables 4.3 and 4.4. Finally, the norm-norm plots are given. For this simple example, the elements of matrix $\tilde{H}$ and vector $\overline{b}$ were randomly generated and are shown in Equations 4.50 and 4.51.

$$\widetilde{H} = \begin{bmatrix} 837.0789 & 208.1173 & 247.5771 & 48.6663 & 585.0636 & 83.5205 \\ 505.7258 & 468.5202 & 613.7354 & 781.0929 & 561.3329 & 228.5038 \\ 431.4836 & 176.7597 & 395.1752 & 107.7964 & 98.7934 & 456.9960 \\ 898.3065 & 602.2620 & 617.2275 & 589.2601 & 527.8372 & 482.8333 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 898.3065 & 602.2620 & 617.2275 & 589.2601 & 527.8372 & 482.8333 \end{bmatrix} \tag{4.50}$$

σ_i	$\widetilde{H}$	$\tilde{H}^T\tilde{H}$
1	4.9626e+003	2.4628e+007
2	622.9068	3.8801e+005
3	478.9823	2.2942e+005
4	177.6345	3.1554e+004
5	3.2285e-014	5.0616e-010
6	2.0120e-030	1.3386e-010

Table 4.1: Singular Values for computed simple test cases

x_i	$\tilde{H}$	$\tilde{H}^T\tilde{H}$
x_1	0.0044	0.0044
x_2	0.0120	0.0120
x_3	-0.0095	-0.0095
x_4	0.0013	0.0013
x_5	-0.0057	-0.0057
x_6	0.0011	0.0011

Table 4.2: Solution of $\hat{x}$ using $\tilde{H}$ and $\tilde{H}^T\tilde{H}$

$$\overline{b} = \begin{bmatrix} 0.6977 \\ 0.1542 \\ 0.3700 \\ 3.6820 \\ \vdots \\ 3.6820 \end{bmatrix} \tag{4.51}$$

σ_i's retained	$\|\overline{x}\|$	$\|\overline{\rho}\|$
σ_1, σ_2, σ_3, σ_4, σ_5, σ_6	2.386320e+014	7.234728e+017
σ_1, σ_2, σ_3, σ_4, σ_5	6.958308e-002	4.986367e+001
σ_1, σ_2, σ_3, σ_4	1.701464e-002	1.977524e-014
σ_1, σ_2, σ_3	3.597810e-003	2.954046e+000
σ_1, σ_2	2.141795e-003	3.262464e+000
σ_1	2.132577e-003	3.264806e+000

Table 4.3: Norm of solution and residual using $\tilde{H}$

σ_i's retained	$\|\overline{x}\|$	$\|\overline{\rho}\|$
σ_1, σ_2, σ_3, σ_4, σ_5, σ_6	2.150794e-002	5.931892e+004
σ_1, σ_2, σ_3, σ_4, σ_5	2.101312e-002	1.506255e+004
σ_1, σ_2, σ_3, σ_4	1.701464e-002	4.055163e-011
σ_1, σ_2, σ_3	3.597810e-003	5.247405e+002
σ_1, σ_2	2.141795e-003	8.457092e+002
σ_1	2.132577e-003	8.492091e+002

Table 4.4: Norm of solution and residual using $\tilde{H}^T\tilde{H}$

As expected, Table 4.2 shows that the same best solution is obtained regardless of whether $\tilde{H}$ is used directly, or if the normal matrix $\tilde{H}^T\tilde{H}$ is formed. Judging by the singular values in Table 4.1, it is obvious that the rank of $\tilde{H}$ is 4, because there are four singular values which are much greater than zero. However this rank deficiency is clearly seen earlier by noting that this information matrix contains only 4 independent rows out of a possible 12. This is evidence that the final 8 rows of $\tilde{H}$ convey no further data needed to solve for the 12 parameters in $\overline{x}$. Notice that in Tables 4.3 and 4.4, the best solution occurs when the first four singular values are retained ($\sigma_1, \sigma_2, \sigma_3, \sigma_4$), discarding the rest. Also, looking at the norm-norm plots in Figures 4.3 and 4.4, one can see that the optimal solution occurs at the point where four singular values are kept in the solution, while the remaining two are discarded. (Remember that a singular value disposal is defined as the act of setting the *reciprocal* of a particular singular value to zero, not the singular value itself.) Thus, both ill-conditioned matrices can be used to obtain a non-unique solution by managing the inclusion/exclusion of singular values. Of course, for much larger problems, rank deficiency may not be readily apparent.

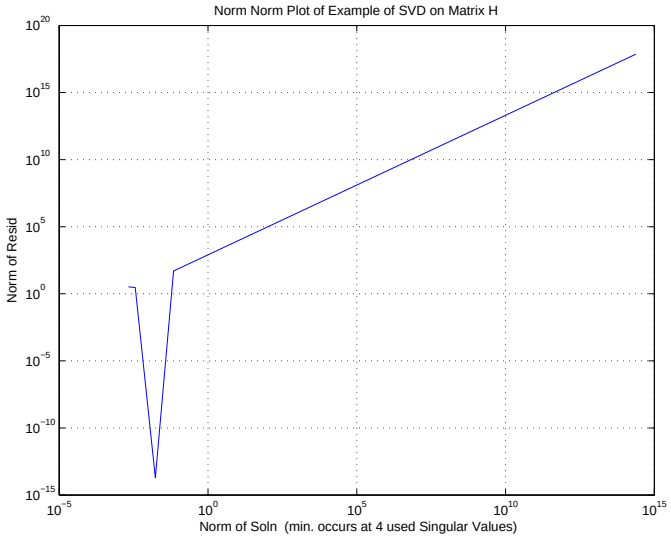


Figure 4.3: Norm of Residual vs. Norm of Solution for H

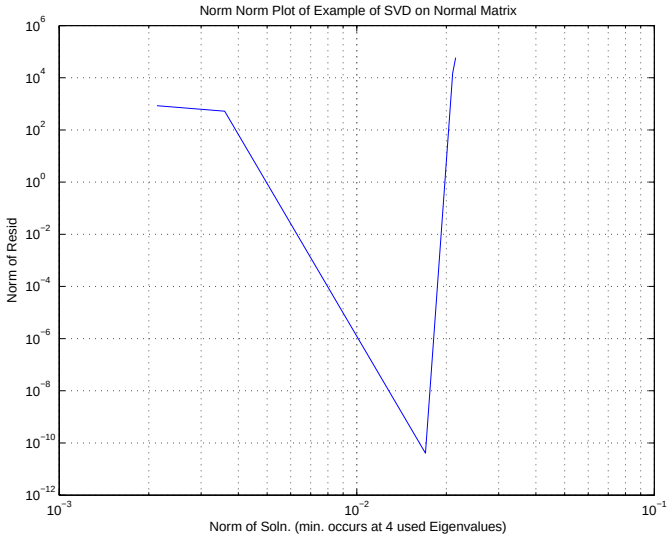


Figure 4.4: Norm of Residual vs. Norm of Solution for $H^T H$

4.2 Summary

This chapter answered the question of how the SVD is applied in the solution of an estimation problem and the five techniques that are used to determine the best method of singular value disposal. Also, a small example of the SVD was given to illustrate the inspection method. If one is curious, the next chapter explains the steps used to implement the parallel SVD algorithm. Otherwise, Chapter 5 is not necessary background for Chapters 6 and 7.

Chapter 5

PLASS Methods of Computing the SVD

In order to apply the SVD to solve for a very large gravity field, new code, written in C, and using the object based parallel infrastructure of Parallel Linear Algebra Package (PLAPACK) (van de Geijn [37]), was developed. Over 200 new parallel routines were written and integrated into two SVD tools required for this study and are given the common name (**P**arallel **LA**rge **S**vd **S**olver) **PLASS**. These two tools are distinguished as **PLASS**1 and **PLASS**2. Already existing code, for example the Fortran 90 and the Fortran 77 seen in the widely used canned routines of the Linear Algebra Package (LAPACK) is not able to easily handle estimation problems involving gravity fields much larger than about 5000 unknown parameters. Because the Fortran code of LAPACK is *sequential* in nature, using only one computer, applying it to typical problem sizes seen in either the Challenging Minisatellite Payload (CHAMP) or Gravity Recovery and Climate Experiment (GRACE) projects, would require months or years of computer wall clock time to solve for SVD gravity solutions. There are however, a few selected Fortran 90 and Fortran 77 routines that have been wrapped inside the **PLASS** SVD Tools, which are needed to perform some elementary single processor calculations. The following sections describe the SVD algorithm of **PLASS** and its methods of overcoming the numerous computational obstacles faced when implementing the SVD

technique to solve for large gravity fields. When either of the below algorithms, **PLASS**1 or **PLASS**2 are utilized, implementing the SVD becomes a viable option for solving large systems of equations.

5.1 Bidiagonalization of a Matrix

The first stage in performing the singular value decomposition on a system of linear equations is to decompose the representative matrix to a bidiagonal form, in which all matrix entries are zero except for the main diagonal and the upper diagonal. The way in which a matrix can be bidiagonalized is through the use of orthogonal *transformations* (Watkins [7], Demmel [69]). The two most common forms are the plane rotation (i.e. Givens' rotation) and the reflection (i.e. Householder reflection), the latter of which is described and defined in the next section.

5.1.1 Householder Reflection

Also called a reflector, the Householder transformation, named after A.S. Householder, who was the first to use them in matrix computations, is a transformation which zeroes out all but the first element of a particular matrix column or row. This reflector is an operator, that is multiplied against a matrix. To illustrate what this means, let us first consider its operation in a two dimensional basis space. Let ξ be any line in $\Re^2$ that passes through the origin. The operator that reflects through the line ξ is a linear transformation, so it can be represented by a matrix (Watkins [7]). To determine that matrix, let nonzero vector $\mathbf{v}$ lie along ξ and nonzero vector $\mathbf{u}$ be orthogonal to ξ. Then $\mathbf{u}, \mathbf{v}$ is a basis of $\Re^2$ so every $\mathbf{x} \in \Re^2$

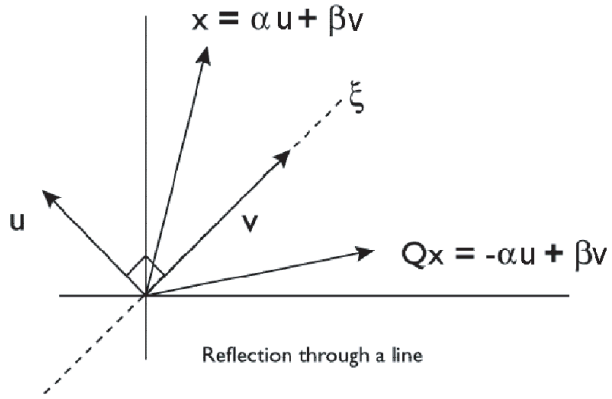


Figure 5.1: Reflection through a line.

can be expressed as a linear combination of $\mathbf{v}$ and $\mathbf{u}$: $\mathbf{x} = \alpha\mathbf{u} + \beta\mathbf{v}$. Therefore the *reflection* of $\mathbf{x}$ through ξ is $-\alpha\mathbf{u} + \beta\mathbf{v}$, see Figure 5.1.

This two dimensional orthogonal transformation seen in its matrix format can be written as a rotation or a reflection, namely

$$Q_{rotation} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \quad Q_{reflection} = \begin{pmatrix} \cos\theta & \sin\theta \\ \sin\theta & -\cos\theta \end{pmatrix} \tag{5.1}$$

while $Q\mathbf{x}$ is obtained by reflecting the vector x across the line defined by

$$S = span \begin{pmatrix} \cos\frac{\theta}{2} \\ \sin\frac{\theta}{2} \end{pmatrix}. \tag{5.2}$$

If the basis space were three dimensional, then the reflection would occur across a simple plane and the Householder matrix Q would be 3-by-3. Now let $\mathbf{v} \in \Re^n$ be nonzero. Then the following is the mathematical representation of this transformation in n-dimensions.

$$Q = I - \frac{2}{\mathbf{v}^T\mathbf{v}}\mathbf{v}\mathbf{v}^T \tag{5.3}$$

(Notice that the fraction term evaluates to a scalar, $\alpha = \frac{2}{\mathbf{v}^T\mathbf{v}}$.) The vector $\mathbf{v}$ is called a Householder vector. If a vector x is multiplied by Q, then it is reflected in the

hyperplane span$\{\mathbf{v}\}^{\perp}$ (Golub [23]). Now that we have defined what a Householder transformation is, let us describe how it is multiplied against a matrix which is to be transformed.

5.1.1.1 Reduction to Bidiagonal form

The Householder matrix is never explicitly formulated in this study. Furthermore, the effect of a transformation on a matrix is also never literally recorded, because storing zeroes is a wasteful use of computer memory. Rather, every computed Householder vector is stored over its corresponding column or row that would have contained the actual result of the transformation. To transform any matrix column, such that all elements but the first are zero, a reflection must occur on the left side of the matrix. To transform a particular matrix row, such that all elements but the first are zero, a reflection must occur on the right side of the matrix. These reflections must be performed in an alternating fashion so as not to "destroy" the zeroing of elements from a previous reflection. The following sketched procedure illustrates this scheme. Let $U_1 \in \Re^{mxm}$ be a reflector, acting on the left side of A, or acting on a column of A, such that

$$U_{H1} \begin{bmatrix} a_{11} \\ a_{12} \\ \vdots \\ a_{n1} \end{bmatrix} = \begin{bmatrix} * \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{5.4}$$

Notice the first column of $U_1 A$ consists of zeroes except for the first entry. Now choosing the first row in A, $[a_{11}, a_{12}, \ldots, a_{1n}]$, let $V_1 \in \Re^{nxn}$ be a reflector of the

form

$$V_{H_1} = \left(\begin{array}{c|c} 1 & 0 \ldots 0 \\ \hline 0 & \\ \vdots & \hat{V}_{H1} \\ 0 & \end{array}\right) \tag{5.5}$$

such that

$$[a_{12}, a_{13}, \ldots, a_{1n}]V_{H1} = [*0 \ldots 0]. \tag{5.6}$$

Remember that V_{H1} is equivalent to a rank one update of the identity matrix (Equation 5.3) corresponding to the first right side Householder reflection of matrix A.

Therefore the first row of $U_{H1}AV_{H1}$ consists of zeros except for the first two entries. Since the first column of V_{H1} is $[1 \ldots 0]^T$, the first column of $U_{H1}A$ is unaltered by right side multiplication of V_{H1}, therefore $U_{H1}AV_{H1}$ has the form

$$U_{H1}AV_{H1} = \left(\begin{array}{c|c} * & * \ 0 \ldots 0 \\ \hline 0 & \\ \vdots & \hat{A} \\ 0 & \end{array}\right) \tag{5.7}$$

Now, the next step (left side reflection) is identical to the first, except that it acts on the submatrix $\hat{A}$. Therefore after four steps, which completes two reflections on both sides, matrix A has the following appearance.

$$U_{H2}U_{H1}AV_{H1}V_{H2} = \left(\begin{array}{cc|ccc} * & * & 0 & 0 & \ldots \ 0 \\ 0 & * & * & 0 & \ldots \ 0 \\ \hline 0 & 0 & & & \\ \vdots & \vdots & & \hat{\hat{A}} & \\ 0 & 0 & & & \end{array}\right) \tag{5.8}$$

The fifth and sixth step will perform the third reflection for U_{H3} and V_{H3} respectively on submatrix $\hat{\hat{A}}$. Continuing this policy, after *m* reflections we achieve the

following bidiagonal form of A:

$$U_{Hn}\ldots U_{H2}U_{H1}AV_{H1}V_{H2}\ldots V_{Hn-2} = \left(\begin{array}{cccc} * & * & & 0 \\ & \ddots & \ddots & \\ & & \ddots & * \\ 0 & & & * \\ \hline & & 0 & \end{array}\right) = \text{B} \tag{5.9}$$

In many applications, m (the number of observations) is much larger than n (the number of unknown parameters), so it is obvious that once the number of reflections has reached n-2, the last two reflections, U_{Hn-1}, and U_{Hn} should be applied only on the left side, because all matrix rows have been zeroed above the two non-zero diagonals in the last column, signified by asterisks in Equation 5.9. If m $\gg$ n, then it makes sense to perform all of the left side Householder reflections of U_{Hi} before the bidiagonalization is commenced. This should be done in block format, by zeroing out more than one column at a time. This is called the QR decomposition and transforms the matrix A into an upper triangular form, R, where all entries are zero except those on and above the diagonal. The following expression illustrates this.

$$A = U_{Rn}\ldots U_{R2}U_{R1}A = QR = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix} \tag{5.10}$$

where $R \in \Re^{nxn}$ is upper triangular. These left side Householder reflections are then applied to the right hand side vector, $\overline{b}_{r.h.s.}$, as seen in the following equation.

$$\overline{b} = [U_{Rn}\ldots U_{R2}U_{R1}]\,\overline{b}_{r.h.s.}. \tag{5.11}$$

(note: **PLASS** does not perform block Householder reflections, only single column reflections.) In the case of GRACE data, the matrix R, provided by **Aesop** (Gunter [5]), is an input to **PLASS**. Then, in the next transformation stage, the bidiagonalization is performed in the alternating fashion, as seen in Equation 5.9, on matrix R. The advantage of using matrix R, which is n-by-n, is that it is relatively smaller than m-by-n matrix A. This is a more efficient process than implementing Equation 5.9 on the original matrix A at the beginning. Thus, R is reduced to its bidiagonal form $\hat{R} = \tilde{U}\tilde{B}\tilde{V}^T$. Since all of these matrices are *n* by *n*, it can be shown (Watkins [7]) that

$$A = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{bmatrix} \tilde{U} & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \tilde{B} \\ 0 \end{bmatrix} \tilde{V}^T. \tag{5.12}$$

If we let

$$\hat{U} = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{bmatrix} \tilde{U} & 0 \\ 0 & I \end{bmatrix} = \begin{bmatrix} Q_1\tilde{U} \\ Q_2 \end{bmatrix} \in \Re^{nxn} \tag{5.13}$$

$$B = \begin{bmatrix} \tilde{B} \\ 0 \end{bmatrix} \in \Re^{mxn} \tag{5.14}$$

and $\hat{V} = \tilde{V} \in \Re^{nxn}$, it can be said that $A = \hat{U}B\hat{V}^T$. In this less costly procedure, the right side reflections are applied to the smaller matrix $\hat{R}$ instead of to the larger matrix A. However, once the upper $\hat{R}$ is obtained, any right side reflections will destroy the zeros in the columns under the main diagonal, so the left side reflections must be repeated to re-establish zeros under the main diagonal. But if the ratio $\frac{m}{n}$ is sufficiently large, the added cost of the extra left side multiplications will be more than offset by the savings in the right side multiplications (Watkins [7]). In the case of CHAMP, the n by n normal matrix A (arising from the normal equation) is given as $A = H^T H$, where H is the so called m by n information matrix. Therefore, it

makes no sense in the CHAMP case to transform A into an upper $\hat{R}$ form. However, in the case of GRACE, the normal equations are not formed, leaving us with $A = H$, where H is m by n, and m $\gg$ n. Therefore in case of GRACE, A is transformed into $\hat{R}$ before the alternating left and right Householder reflections are applied, as seen in Equation 5.9. After the matrix A has been reduced to its bidiagonal form, one may ask the question, what do we do with all of the left and right side reflections that have been stored in the form of Householder vectors over the original matrix A? The answer is that they eventually must be applied in an appropriate manner that will lead to a solution. It is necessary to mention that the scalar α seen in Equation 5.3 is also stored for each Householder reflection, into a one dimensional storage array. There are two types of one dimensional storage arrays, one containing all of the left side reflection coefficients, called the **sL** array, and one containing all of the right side reflection coefficients, called the **sR** array. The next section discusses how all of these Householder vectors and coefficients are applied in the solution technique.

5.1.1.2 Left Side Householder Reflections

Before we reach a *fork* in the *road* as described in the next section, it is necessary first to rearrange the bidiagonalization of A into a form which solves both the minimum least squares equation and the orthogonal transformation equation. In both cases the equation $\tilde{A}\overline{x} = \overline{b}$ must be re-expressed to put $\overline{x}$ into closed form, namely $\overline{x} = \tilde{A}^{-1}\overline{b}$. After $\overline{x}$ has been calculated, it is the vector which minimizes the norm $\|\hat{A}\overline{x} - \overline{b}\|$. Of course, the term $\tilde{A}^{-1}$ is not calculated by direct means, but

in a circuitous way that involves the inverse of the bidiagonalized A. The following equation introduces the decomposition of A into the bidiagonal matrix B.

$$U_{H_n}^T \ldots U_{H_2}^T U_{H_1}^T A V_{H1} V_{H2} \ldots V_{Hn-2} = B \tag{5.15}$$

By expressing bidiagonalized A in closed form, we see that

$$A = [U_{H1} U_{H2} \ldots U_{Hn}]\, B\, [V_{H1} V_{H2} \ldots V_{Hn-2}]^T . \tag{5.16}$$

Since each U_{Hi} and V_{Hi} are orthonormal matrices, the expression for the inverse of bidiagonalized A is

$$A^{-1} = [V_{H1} V_{H2} \ldots V_{Hn-2}]\, B^{-1} [U_{H1} U_{H2} \ldots U_{Hn}]^T. \tag{5.17}$$

But the inverse of B is also not carried out directly. Obtaining the inverse of B will be discussed later. For now the steps taken in the solution up to and including the bidiagonalization of A, are described. Inserting the bidiagonalization expression of A^{-1} into the relation $\overline{x} = A^{-1}\overline{b}$, yields,

$$\overline{x} = [V_{H1} V_{H2} \ldots V_{Hn-2}]\, B^{-1} [U_{H1} U_{H2} \ldots U_{Hn}]^T \overline{b}. \tag{5.18}$$

Very fortunately, it is not necessary to explicitly formulate, in $O(m^3)$ operations, the left side reflections into the m x m matrix $U_b = [U_{H1} U_{H2} \ldots U_{Hn}]$. Because there is no future need for them, they can be applied to the right hand side vector $\overline{b}$, transforming it to $\tilde{b} = {U_b}^T \overline{b}$, in as few as $4n^2 + n$ operations. The solution therefore reduces to

$$\overline{x} = [V_{H1} V_{H2} \ldots V_{Hn-2}]\, B^{-1} \tilde{b}. \tag{5.19}$$

Now, depending upon whether we employ **PLASS**1 or **PLASS**2, described in the sections below, it may be necessary to explicitly formulate the matrix

$$V_b = [V_{H1} V_{H2} \ldots V_{H n-2}] \,. \tag{5.20}$$

If we were to naively carry out each each matrix multiplication to formulate V_b, then the number of operations would be on the order of $O(n^3)$. However, by accumulating these reflections in block format and taking advantage of the many zeroes in each reflection, the number of operations can be drastically reduced, as shown in a later section. The next section describes the choice which dictates a major path in the **PLASS** algorithm that is to be taken, the so called *fork* in the *road*. That is, whether an SVD covariance matrix is to be calculated or not. If an SVD covariance matrix is needed, then **PLASS**2 should be used, otherwise **PLASS**1 should be applied.

5.2 SVD Tool Method One: PLASS1

Employing **PLASS**1 requires twice the number of processors (in comparison to **PLASS**2), however it consumes about half of the wall clock time needed by **PLASS**2. The reason is that **PLASS**1 does not accumulate matrix V_b, instead all Householder reflections (both left and right) are applied to the right hand side in as few as $4n^2 + 6n$ operations. Therefore, no matrices are accumulated or explicitly formulated anywhere in the process.

5.2.1 Computing the SVD without Right Side Orthonormal Basis

After matrix A has been reduced to its bidiagonal form (matrix B) further transformations applied to reduce B are no longer Householder reflections, rather they are Givens' rotations, because we only need to set single particular elements in matrix B to zero, not entire rows or columns. Such a transformation will be described in detail later, after the policy of managing them has been discussed.

5.2.1.1 Managing the Givens' Rotations

Givens' rotations are applied to the left and right side of matrix B in an alternating fashion, not unlike the pattern of the Householder transformations, until matrix B has been decomposed to its diagonal values. Any left side Givens' rotation applied to matrix B, during this process, is multiplied against the right hand side vector $\tilde{b}$. However, the right side Givens' rotations (V_{Gi} described below) can be applied to $\tilde{b}$ only **after** matrix B has been fully decomposed to its singular values. Since there is no accumulated matrix V_b to which they can be multiplied against, all V_{Gi} rotations must be stored in an array until such time it is possible to multiply them against the right hand side. The following equations illustrate this *bottleneck* caused by Σ^{-1}.

$$\overline{x} = [V_{H1}V_{H2}\dots V_{Hn-2}]\left[V_{G1}V_{G2}\dots V_{Gf}\right]\Sigma^{-1}\left[{U_G}^T{}_f{U_G}^T{}_{f-1}\dots{U_G}^T{}_2{U_G}^T{}_1\right]\tilde{b}, \tag{5.21}$$

thus,

$$B^{-1} = \left[V_{G1}V_{G2}\dots V_{Gf}\right]\Sigma^{-1}\left[{U_G}^T{}_f{U_G}^T{}_{f-1}\dots{U_G}^T{}_2{U_G}^T{}_1\right]. \tag{5.22}$$

Since

$$\Sigma = \begin{pmatrix} \sigma_1 & & & 0 \\ & \sigma_2 & & \\ & & \ddots & \\ 0 & & & \sigma_n \end{pmatrix}, \tag{5.23}$$

Σ^{-1} is simply obtained by taking the reciprocals of each of its diagonal elements. It is clear that all of the U_{Gi} rotations may be quickly applied to $\tilde{b}$ as they are generated during the convergence to a diagonalized matrix Σ, as seen in the next equation, which is obtained by simplifying Equation 5.21.

$$\overline{x} = [V_{H1}V_{H2}\ldots V_{Hn-2}]\left[V_{G1}V_{G2}\ldots V_{Gf}\right]\Sigma^{-1}\tilde{\tilde{b}}, \tag{5.24}$$

where

$$\tilde{\tilde{b}} = \left[{U_G}^T{}_f{U_G}^T{}_{f-1}\ldots{U_G}^T{}_2{U_G}^T{}_1\right]\tilde{b}. \tag{5.25}$$

The new right hand side becomes (for later use in Equation 5.41),

$$\tilde{\tilde{\tilde{b}}} = \Sigma^{-1}\left[{U_G}^T{}_f{U_G}^T{}_{f-1}\ldots{U_G}^T{}_2{U_G}^T{}_1\right]\tilde{b}. \tag{5.26}$$

However the class of right side Givens' rotations V_{Gi} may not be applied to $\tilde{\tilde{\tilde{b}}}$, before convergence to singular values, nor can they be applied to a non existing matrix V_b. Therefore they must be placed in a storage array (along with the corresponding column **indices** of B to which each rotation was applied) until after the iteration for the Σ^{-1} matrix in Equation 5.24 has been completed. Furthermore, the storage must occur in the proper sequence, so that the rotations can be applied later in reverse order to the vector $\tilde{\tilde{\tilde{b}}}$. As is soon discussed, this amounts to storing four items of information for every right side rotation (V_{Gi}) and is the reason for the enormous

amount of memory required to catalogue and bookkeep these transformations. The average number of iterations needed to converge to the singular values is $O(n^2)$, where n is the number of unknown parameters being estimated. (In the case of CHAMP and GRACE there are exactly $\frac{53,272,938}{2}$ and $\frac{348,039,215}{2}$ right side Givens' rotations, respectively, and multiplying each ratio by 4 gives the total number of items to be stored for both cases.)

5.2.2 Givens' Rotation

Also called a *plane rotator*, a Givens' rotation is an n x n matrix that is never explicitly formulated to selectively set elements of a target matrix to zero. The kth plane rotator has the form

$$V_{Gk} = \begin{pmatrix} 1 & & & & & & & & & & & & \\ & 1 & & & & & & & & & & & \\ & & \ddots & & & & & & & & & & \\ & & & 1 & & & & & & & & & \\ & & & & c & & & & -s & & & & \\ & & & & & 1 & & & & & & & \\ & & & & & & \ddots & & & & & & \\ & & & & & & & 1 & & & & & \\ & & & & s & & & & c & & & & \\ & & & & & & & & & 1 & & & \\ & & & & & & & & & & \ddots & & \\ & & & & & & & & & & & 1 \end{pmatrix} \begin{matrix} \\ \\ \\ \\ \longleftarrow i \\ \\ \\ \\ \longleftarrow j \\ \\ \\ \\ \end{matrix} \qquad \begin{matrix} c = \cos\theta \\ s = \sin\theta \end{matrix}$$
$$\begin{matrix} \uparrow & & \uparrow \\ i & & j \end{matrix} \tag{5.27}$$

All blank entries in the matrix above are zeros. Therefore a plane rotator looks like an Identity matrix except that one pair of columns and rows contains the two

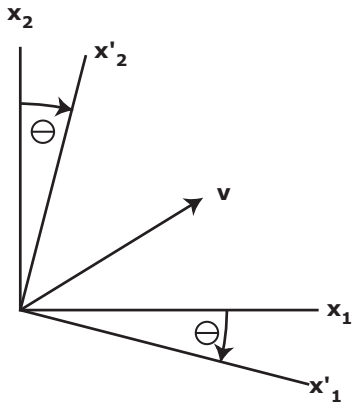


Figure 5.2: Reference Frame Rotation.

dimensional plane information. It is these column and row indices and the "c" and "s" values which are stored into the array, discussed previously, for bookkeeping during the multiplication of each rotation against the right hand side vector $\tilde{\tilde{b}}$. In *compact* form the rotator looks like a 2 by 2 matrix

$$V_{Gk} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}, \tag{5.28}$$

that rotates a reference frame of a vector through an angle θ as seen in Figure 5.2. So, for every $x \in \Re^2$, there exists a rotator V_G such that $V_G^T x = \begin{bmatrix} y \\ 0 \end{bmatrix}$. Now that we have an idea what Givens' rotations are and how each one sets a matrix element to zero, the alternating left and right manner in which they are all applied to the bidiagonal matrix B is discussed next.

5.2.2.1 Convergence to Singular Values: Chasing the Bulge

Applying the Givens' rotations, until matrix B is diagonal, is the process of matrix B converging to its singular values. It is also known as a QR iterative

process where each QR transformation is a step in a *pattern* referred to as "chasing the bulge" (also called the Francis sweep). The standard algorithm for finding singular values of a bidiagonal matrix B is the QR algorithm applied implicitly to $B^T B$ (Golub and Kahan [61]). This *pattern* or algorithm computes a sequence B_i of bidiagonal matrices starting from $B_0 = B$ as follows. From B_i the algorithm computes a shift σ^2, which is usually taken to be the smallest eigenvalue of the bottom 2 by 2 block of $B_i B_i^T$. Then the algorithm does an implicit QR factorization of the shifted matrix $B_i^T B_i - \sigma^2 I = QR$, where Q is orthogonal and R upper triangular, from which it computes a bidiagonal B_{i+1} such that $B_{i+1}^T B_{i+1} = RQ + \sigma^2 I$. As i increases, B_i converges to a diagonal matrix whose elements correspond to the singular values of the initial matrix B_0 (Demmel and Kahan [69]). The detection of convergence is an important issue and is addressed in their article. The algorithm used in this study is a variation of this standard QR algorithm which computes all of the singular values of a bidiagonal matrix, even the tiniest ones, with guaranteed high relative accuracy (Demmel and Kahan [69]). It is called the "implicit zero-shift QR algorithm". When the σ^2 term is zero, the algorithm is equivalent to the standard. However, this improved algorithm is organized in such a way as to guarantee that each entry of B_{i+1} is computed from B_i to nearly full machine precision (Demmel and Kahan [69]). When the ratio of the largest to smallest singular value (condition number) is modest, the standard QR is used and when this ratio is large, the implicit zero-shift QR is used. This hybrid algorithm is discussed at length in the paper by Demmel and Kahan, (Demmel and Kahan [69]), and the reader is referred to that document for a detailed description. (It is worth mentioning, that

this algorithm was written in Fortran and was wrapped inside the **PLASS** tool. For **PLASS**2, the algorithm was altered so that a multitude of processors could perform the iteration on matrix B and implement the necessary global updates to the V_b matrix.)

To summarize the standard QR algorithm, we need to define some simple notation, and the example and text seen in Demmel and Kahan [69] will be recreated here for the sake of simplicity and clarity in *indices*. Keep in mind that in the following figures and equations, any rotation on the left side of matrix B corresponds to a U_{Gk} rotation and any rotation on the right side of matrix B corresponds to a V_{Gk} rotation.

Let $J(i, j, \theta)$ denote the Givens rotation entries i and j by angle θ. (Note these indices correspond exactly to those seen in Equations 5.27, 5.28 and Figure 5.2.) In other words, $J(i, j, \theta)$ is an n by n identity matrix except for rows and columns i and j whose intersections consist of the following 2 by 2 rotation matrix:

$$\begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}. \tag{5.29}$$

Given the vector $\overline{x}$, choosing θ so that $\frac{x_j}{x_i} = \tan\theta$, means that the i-th and j-th entries of $J(i, j, \theta)\overline{x}$ will contain $\pm\sqrt{(x_i^2 + y_i^2)}$ and 0, respectively. To illustrate this, let us carry this out on a 4 by 4 example, where we use x and + to indicate nonzero entries and 0 and blank to indicate zero entries. Initially B_i is in the form

$$B_i = \begin{pmatrix} x & x & & \\ & x & x & \\ & & x & x \\ & & & x \end{pmatrix}. \tag{5.30}$$

We begin by postmultiplying B_i by $J_1 \equiv J(i,j,\theta_1)$. The very important choice of θ_1 will be discussed in a moment. This introduces a nonzero entry in the (2,1) position:

$$B_i J_1 = \begin{pmatrix} x & x & & \\ + & x & x & \\ & & x & x \\ & & & x \end{pmatrix}. \tag{5.31}$$

$B_i J_1$ will now be pre- and postmultiplied by a sequence of Givens' rotations whose purpose is to "chase the bulge" indicated by "+" off the end of the matrix. Choose θ_2 so that $J_2 \equiv J(1,2,\theta_2)$ introduces a zero in the (2,1) entry of $J_2 B_i J_1$:

$$J_2 B_i J_1 = \begin{pmatrix} x & x & + & \\ 0 & x & x & \\ & & x & x \\ & & & x \end{pmatrix} \tag{5.32}$$

Next choose θ_3 in $J_3 \equiv J(2,3,\theta_3)$, θ_4 in $J_4 \equiv J(2,3,\theta_4)$, θ_5 in $J_5 \equiv J(3,4,\theta_5)$ and θ_6 in $J_6 \equiv J(3,4,\theta_6)$, to give the following sequence of transformations:

$$J_2 B_i J_1 J_3 = \begin{pmatrix} x & x & 0 & \\ & x & x & \\ & + & x & x \\ & & & x \end{pmatrix};\ J_4 J_2 B_i J_1 J_3 = \begin{pmatrix} x & x & & \\ & x & x & + \\ & 0 & x & x \\ & & & x \end{pmatrix} \tag{5.33}$$

$$J_4 J_2 B_i J_1 J_3 J_5 = \begin{pmatrix} x & x & & \\ & x & x & 0 \\ & & x & x \\ & & + & x \end{pmatrix};\ J_6 J_4 J_2 B_i J_1 J_3 J_5 = \begin{pmatrix} x & x & & \\ & x & x & \\ & & x & x \\ & & 0 & x \end{pmatrix}, \tag{5.34}$$

where $B_{i+1} \equiv J_6 J_4 J_2 B_i J_1 J_3 J_5$. The error analysis of Givens' rotations (Wilkinson [70]) shows that the computed B_{i+1} is the exact transformation of a matrix $B_i + E$ where E is on the order of $p(n)\epsilon B_i$, $p(n)$ a modest function of n (Demmel and Kahan [69]). To choose θ_1, the shift σ^2 is computed. The shift is generally chosen to

be the smallest eigenvalue of the bottom right 2 by 2 submatrix of the matrix $B_i B_i^T$. θ_1 is selected so that J_1 introduces a zero into the (2,1) entry of $J_1^T(B_i^T B_i - \sigma^2 I)$. Therefore, this means that

$$\tan\theta_1 = \frac{(B_i^T B_i)_{12}}{\sigma^2 - (B_i^T B_i)_{11}}. \tag{5.35}$$

This choice of shift is called the Wilkinson's shift and it guarantees at least linear convergence and generally yields asymptotic cubic convergence of the offdiagonal entries of B_i to zero (Parlett [71]). The assumption is that the arithmetic is done exactly. To illustrate the zero-shift, let us take $\sigma^2 = 0$. For simplicity of notation the subscript i is dropped on B and from Equation 5.35 we see that $\tan\theta_1 = -\frac{b_{12}}{b_{11}}$ so that after the first rotation on this 4 by 4 matrix, the result is

$$B^{(1)} \equiv BJ_1 \equiv \begin{pmatrix} b_{11}^{(1)} & 0 & & \\ b_{21}^{(1)} & b_{22}^{(1)} & b_{23} & \\ & & b_{33} & b_{34} \\ & & & b_{44} \end{pmatrix}. \tag{5.36}$$

The superscript on matrix B and its entries indicate that J_1 has been applied. Comparing Equation 5.36 to 5.31 it is clear that the (1,2) entry is zero instead of nonzero. This zero will propagate through the rest of the algorithm and is the key to its effectiveness (Demmel and Kahan [69]). After applying the rotation J_2 we have

$$B^{(2)} \equiv J_2 B J_1 \equiv \begin{pmatrix} b_{11}^{(2)} & b_{12}^{(2)} & b_{13}^{(2)} & \\ 0 & b_{22}^{(2)} & b_{23}^{(2)} & \\ & & b_{33} & b_{34} \\ & & & b_{44} \end{pmatrix} \tag{5.37}$$

It is interesting to note that

$$\begin{pmatrix} b_{12}^{(2)} & b_{13}^{(2)} \\ b_{22}^{(2)} & b_{23}^{(2)} \end{pmatrix} = \begin{pmatrix} \sin\theta_2 b_{22}^{(1)} & \sin\theta_2 b_{23} \\ \cos\theta_2 b_{22}^{(1)} & \cos\theta_2 b_{23} \end{pmatrix} \tag{5.38}$$

is a rank one matrix. Therefore, postmultiplication by J_3 to zero out the (1,3) entry will also zero out the (2,3) entry:

$$B^{(3)} \equiv J_2 B J_1 J_3 \equiv \begin{pmatrix} b_{11}^{(2)} & b_{12}^{(3)} & 0 & \\ 0 & b_{22}^{(3)} & 0 & \\ & b_{32}^{(3)} & b_{33}^{(3)} & b_{34} \\ & & & b_{44} \end{pmatrix}. \tag{5.39}$$

Comparing matrix $B^{(3)}$ to that seen in Equation 5.32 one notices that there is an extra zero on the superdiagonal. Applying rotation J_4 on the left side, repeats the situation. The submatrix of $J_4 J_2 B J_1 J_3$ from the intersection of rows 2 and 3 and columns 3 and 4 is rank one, and a rotation by J_5 on the right side will not only zero out entry (2,4) but also entry (3,4). This pattern or movement of the nonzero "bulge" continues in a downward fashion, toggling above and below the diagonal until it is "chased" out of the matrix. Then, starting at the top left of matrix B_i, this entire pattern, or "bulge chasing" is repeated again from choosing a new θ_1 based on the current state of B_i. Each superdiagonal element entry will converge closer to zero with each cycle of "bulge chasing". When all superdiagonal elements have approached a value very close to zero, convergence is declared and the singular values of the original matrix A, will exist along the diagonal elements of the diagonalized matrix B. (As mentioned earlier, the detection of this convergence is of utmost importance. A discussion of this can be found in section four of Demmel and Kahan [69] for the curious reader.) Now that the singular values have been converged to the diagonal matrix Σ, Σ^{-1} is obtained by taking the reciprocal of its diagonal elements so that Equation 5.24 may be simplified into the following expression

$$\overline{x} = [V_{H1} V_{H2} \dots V_{Hn-2}] \left[V_{G1} V_{G2} \dots V_{Gf} \right] \tilde{\tilde{b}} \tag{5.40}$$

leaving us with two major steps remaining to calculate the solution vector $\overline{x}$.

5.2.3 Applying Right Side Givens' Transformations

At this stage in the SVD algorithm of **PLASS**1, all of the saved Givens' rotations that were applied to the right side of matrix B must be released from the storage array (in the reverse sequential order in which they were recorded) and applied to the transformed right hand side vector called $\tilde{\tilde{\tilde{b}}}$ to produce the vector $\hat{b}$, expressed as

$$\hat{b} = \left[V_{G1} V_{G2} \ldots V_{Gf}\right] \tilde{\tilde{\tilde{b}}}. \tag{5.41}$$

The multiplication of all the Givens' rotations in Equation 5.41 against $\tilde{\tilde{\tilde{b}}}$ can require a maximum of exactly $3n^2$ (where n is the number of unknown parameters) floating point operations. Because we need to record four elements of information for every rotation, i.e. $\cos\theta, \sin\theta$, global column and global row (of matrix B), for the right side Givens' transformations, the storage array discussed in Section 5.2.1.1 needs to be large enough to contain $6n^2$ floating point numbers. If n = 25917, the case of the GRACE 160x160 gravity field problem, the number of stored elements for right side Givens' iterations, takes on a maximum value of $4.030145 \cdot 10^9$. This would require at least 176 processors to handle this storage array, assuming about 23 mega words of available memory per processor. Once $\tilde{\tilde{\tilde{b}}}$ has been completely transformed by all of the right side Givens' rotations, the resulting $\hat{b}$ vector may be substituted into Equation 5.40 to give

$$\overline{x} = \left[V_{H1} V_{H2} \ldots V_{Hn-2}\right] \hat{b}. \tag{5.42}$$

Now, the next and final step is to apply the right side Householder transformations to $\hat{b}$, which are stored in the form of Householder vectors directly over their corresponding columns and rows of matrix A. These Householder vectors are to be released and applied to $\hat{b}$ in the reverse order in which they were stored over matrix A. Because no Householder matrices are explicitly created during this procedure and the transformations applied to the $\hat{b}$ vector involve one inner product between a Householder vector and the $\hat{b}$ vector, and a subsequent addition of two vectors together, the total number of operations is $2n^2 - 5n + 2$.

At this point we now have our solution vector $\overline{x}$. The next section describes how we can solve for new solution vectors by reusing both the storage array containing the right side Givens' iterations and the right side Householder vectors.

5.2.4 Fast Calculation of new Solutions

In this study one objective is to search for an optimum solution with a given input matrix, whether it be a square dense matrix from the normal equations or an upper triangular matrix resulting from the blocked Householder orthogonal transformations, without having to re-decompose these matrices for every different solution. Such a task would be impractical. But by reusing the stored right side Givens' iterations and Householder reflections, different solutions may be quickly calculated in an almost trivial amount of time. It is by selectively discarding singular values along the diagonal seen in Equation 5.23, that the matrix Σ^{-1} in Equation 5.26 can be altered. Subsequently, this alteration changes the contents of $\tilde{\tilde{b}}$, leading to a different correction vector, $\overline{x}$. Thus every different set of singular values chosen

yields a different solution.

5.3 SVD Tool Method Two: PLASS2

PLASS2 starts just like **PLASS**1 by bidiagonalizing matrix A until it reaches the same stage of **PLASS**1's decomposition, illustrated in Equation 5.19. It is at this point that the two methods begin to differ (the so called *fork* in the *road*). Employing **PLASS**2 is in general a better choice to make even though it may require twice the amount of time in comparison to **PLASS**1. As the size of the gravity field to be determined gets larger, **PLASS**1 must store a larger number of right side Givens' iterations/rotations into a dynamic storage array during SVD convergence. This becomes burdensome, as the required number of processors to dynamically store the plane rotation data may quickly exceed the available number of processors. In **PLASS**2, all right side plane (Givens') rotations can be *absorbed* into the matrix V_b, which is an accumulation of all right side Householder transformations. The term *absorbed* implies that the rotations can be multiplied against matrix V_b (of Equation 5.47) as soon as they are generated during the decomposition of matrix B in its convergence to the diagonal matrix Σ. Thus, no large storage array is required and fewer processors are necessary for a given gravity field problem. Another advantage of **PLASS**2, is that it produces the null-space matrix V, whose columns are the singular vectors spanning the null space of the original matrix A. **PLASS**1 does not have this capability. These singular vectors are particularly useful as they can be used to calculate an SVD covariance matrix and other linear objects necessary for error analyzes. In summary, implementing **PLASS**2 requires more time but fewer

processors and produces the null space V of A. **PLASS**1 requires more processors but takes less time and does not return V.

5.3.1 Computing the SVD and the Right Side Orthonormal Basis

The calculation of the right side orthonormal basis of matrix A, also called the null space, begins as an Identity matrix and is transformed with all of the rank one updates from all right side Householder and Givens' transformations. Let V represent this matrix:

$$V = [V_{H1} V_{H2} \dots V_{Hn-2}] \left[V_{G1} V_{G2} \dots V_{Gf} \right] \tag{5.43}$$

Clearly the accumulation must begin with the Householder transformations, starting with V_{H1}, and proceed in the forward direction. Each Householder matrix is not formed explicitly but rather applied as rank one updates to identity matrices. The following section describes how all of the right side Householder transformations are accumulated.

5.3.1.1 Block Accumulation of Householder Reflections

The technique **PLASS**2 uses to accumulate the right side Householder reflections is based on block representations for products of Householder matrices. In general, the forward accumulation of Householder matrices is represented by

$$Q = Q_1 Q_2 \cdots Q_r \qquad Q_i = I - \beta_i v^{(i)} {v^{(i)}}^T \tag{5.44}$$

where Q_i is a Householder transformation, and v is its Householder vector. Since each Q_i is a rank-one modification of the identity, it follows from the structure of

the Householder vectors that Q is a rank-r modification of the identity and can be written in the form (Golub [23])

$$Q = I + WY^T \tag{5.45}$$

where W and Y are n-by-r matrices. The key to computing the *block representation* in Equation 5.45 is the following lemma.

Lemma 5.3.1 (Lemma). *Suppose $Q = I + WY^T$ is an n-by-n orthogonal matrix with W,Y $\in \Re^{nxj}$. If $P = I - \beta vv^T$ with $v \in \Re^n$ and $z = -\beta Qv$, then*

$$Q_+ = QP = I + W_+ {Y_+}^T \tag{5.46}$$

where $W_+ = \left[\begin{array}{cc} W & z \end{array}\right]$ and $Y_+ = \left[\begin{array}{cc} Y & v \end{array}\right]$ are each n-by-(j+1) (Golub [23]).

Proof.

$$\begin{aligned} QP &= \left(I + WY^T\right)\left(I - \beta vv^T\right) = I + WY^T - \beta Qvv^T \\ &= I + WY^T + zv^T = I + \left[\begin{array}{cc} W & z \end{array}\right]\left[\begin{array}{cc} Y & v \end{array}\right]^T. \end{aligned}$$

□

By repeatedly applying this lemma, the block representation of Q in Equation 5.44 can be generated (Golub [23]). The following is an algorithm to implement this technique.

Algorithm 5.1. Suppose $Q = Q_1 Q_2 \cdots Q_r$ is a product of n-by-n Householder matrices as described in Equation 5.44. This algorithm computes matrices $W, Y \in \Re^{nxr}$ such that $Q = I + WY^T$.

$Y = v^{(1)}$

$W = -\beta_1 v^{(1)}$

for j = 2:r

$z = -\beta_j (I + WY^T) v^{(j)}$

$W = \left[\begin{array}{cc} W & z \end{array} \right]$

$Y = \left[\begin{array}{cc} Y & v^{(j)} \end{array} \right]$

end

This algorithm involves about $2r^2 n - \frac{2r^3}{3}$ flops if the zeros in the $v^{(j)}$ are exploited (Golub [23]). This means that as the algorithm proceeds, there are an increasing amount of zero entries in the upper portion of the each subsequent $v^{(j)}$ causing the same zeros to appear in each new z vector, each of which are the columns of matrix W. Thus, matrix W will be lower triangular with non unity values along its diagonal. Matrix Y^T is the matrix of right side Householder vectors that were stored over the original matrix A, therefore matrix Y is lower triangular with unity on its diagonal. When the algorithm above is completed, the final computation of Q is one triangular matrix times the transpose of another triangular matrix, both of which whose contents are about 50 percent zeros, then an addition of the identity matrix. Although the W matrix is constructed one column at at time, it is the avoidance of any multiplication involving these zeros during the final triangular multiplication, that enables the accumulation of the Householder transformation to

be accomplished in a very small amount of time. Without these zeros, the application of the SVD to large gravity problems would be intractable. The next stage in **PLASS**2 is to use the matrix of accumulated right side Householder reflections to *absorb* the right side Givens' plane rotations as they are produced during the convergence of matrix B to its diagonal matrix, Σ, of singular values. The following equation defines the matrix, V_b, that represents the accumulated right side Householder transformations, for n unknown parameters.

$$V_b = [V_{H1} V_{H2} \ldots V_{Hn-2}] \tag{5.47}$$

The next equation illustrates the stage of the algorithm after V_b has been accumulated. Remember, that the **left** side Householder reflections have already been applied to the right hand side, and the convergence of matrix B to Σ has not yet commenced.

$$\overline{x} = V_b B^{-1} \tilde{b}, \quad \tilde{b} = \left[{U_H}^T{}_n \ldots {U_H}^T{}_2 {U_H}^T{}_1 \right] \overline{b} \tag{5.48}$$

Notice that in Equation 5.21 all Givens' right side (V_{Gi}) transformations generated during convergence to singular values, appear on the right side of matrix V_b. This means that when each plane rotation is multiplied against V_b, any **pair of columns** of V_b are updated. (Remember, two columns are updated by a Givens' plane rotation, because as can be seen from Equation 5.27, it is only the intersection of the ith and jth column and row of matrix V_{Gk} which is non-zero or non-unity.) The next section describes a critical parallel computation technique for the process of updating **column pairs** of matrix V_b, which is absolutely essential to **PLASS**2's ability to handle very large gravity problems. Without it, applying the SVD to sys-

tems of linear equations involving more than about 5000 unknown parameters, is not practical.

5.3.2 Performing Convergence to Singular Values on all Processors

The parallel computation technique that enables the SVD solution of very large gravity problems is made possible by the "multi-vector" linear algebra object of the PLAPACK (van de Geijn [37]) infrastructure. The multi-vector data type is a way in which a matrix can be distributed among processors for a given problem (van de Geijn [37]). The main philosophy of this matrix distribution is based on matrix rows. In other words, matrix V_b is distributed such that each processor contains **all** of the columns of the matrix but only a portion of its rows. Because only a portion of the rows of a matrix is fitted onto one processor, they are distributed in a "wrapped" fashion among them. Figure 5.3 illustrates this important matrix distribution concept (multi-vector linear algebra object) for a case where the wrapping is chosen to be exactly one fourth the number of rows of a matrix. (This is done for ease of illustrative purposes only. Often the wrapping occurs at values of 16 or 32 rows.) Because each processor (0, 1, 2, and 3 in Figure 5.3) has access to all columns of matrix V_b and only a portion of its rows, the iteration for singular values may be performed on every processor, **independently**. Remember that the right side Givens' rotations transform **column pairs** of V_b. By copying the main and superdiagonal of matrix B (only two vectors of data) to every processor we can perform the convergence of the bidiagonal matrix B, to its Σ matrix (one vector of data) on every processor independently, while updating any column pair of V_b on any pro-

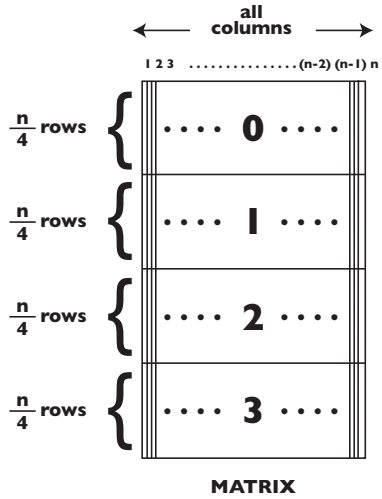


Figure 5.3: Multi-vector matrix distribution.

cessor. This means that during the SVD convergence stage, any right side Givens' rotation as it is generated on any processor, may be applied **asynchronously** among processors, to any two columns of the matrix V_b, with guaranteed correct update sequence continuity for the **entire** matrix V_b. For this investigation, the importance and these benefits of the multi-vector object can not be overemphasized. Figure 5.4 illustrates the "column pair multi-vector update" concept. Clearly, no matter which order (among processors) the labelled steps in Figure 5.4 are carried out, when completed, any two columns of V_b will have been completely and correctly updated. Furthermore, it is obvious, that there is no required communication between processors, as each owns an identical copy of the initial bidiagonal matrix B. Thus, all plane rotations generated among them will be identical, which explains why they can be applied independently.

Once all right side Givens' plane rotations have been multiplied against V_b,

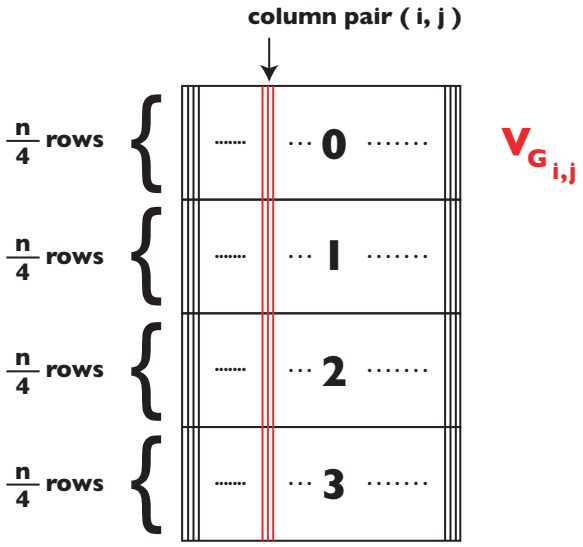


Figure 5.4: Givens' update to any two columns of a Multi-vector Matrix.

matrix B will have converged to the diagonal matrix Σ of its singular values. At this point, matrix V_b is fully transformed into the null space matrix V as shown in the next equation.

$$V = V_b \left[V_{G1} V_{G2} \ldots V_{Gf} \right] \tag{5.49}$$

At this stage in the algorithm of **PLASS**2, $\overline{x}$ may be represented as

$$\overline{x} = V \Sigma^{-1} \tilde{\tilde{b}}, \tag{5.50}$$

remembering that $\tilde{\tilde{b}}$ is given as

$$\tilde{\tilde{b}} = \left[{U_G}^T{}_f {U_G}^T{}_{f-1} \ldots {U_G}^T{}_2 {U_G}^T{}_1 \right] \left[{U_H}^T{}_n \ldots {U_H}^T{}_2 {U_H}^T{}_1 \right] \overline{b}. \tag{5.51}$$

5.3.3 Fast Calculation of new Solutions

It is obvious from Equation 5.50 that different correction vectors $\overline{x}$ can be calculated by selectively setting singular values along the diagonal of Σ to zero,

before Equation 5.50 is multiplied out completely. Because the null space matrix V is also saved (both dynamically among processors, and on disk for future optimization experiments) multiple correction vectors $\overline{x}$ may be quickly re-calculated, enabling the search for an optimal solution based on singular value disposal. This concludes the description of the **PLASS** methods of computing the Singular Value Decomposition in this study.

5.4 Summary

This chapter presented the two main algorithms of how the SVD is calculated in this study. Both are the same except that one algorithm does not accumulate the right singular vectors (**PLASS1**), rather it stores the right side Givens' rotations into a storage array. The other method (**PLASS2**) *does* accumulate the right singular vectors which span the corresponding null space. The next chapter, Chapter 6 presents the results of this investigation.

Chapter 6

Application of PLASS to CHAMP and GRACE

This chapter presents the results of the author's tool **P**arallel **LA**rge **S**vd **S**olver (**PLASS**) in its application to the normal equation of CHAMP in the solution of a 100x100 gravity field and to the orthogonalized equations of GRACE in the solution of a 160x160 gravity field. In both cases, the main purpose is to demonstrate the viability and practicality of applying the parallel solver **PLASS** to very large ill-conditioned systems of satellite only observations and to find the optimal number of retained singular values that produces the best gravity field. Four standard techniques of solution optimization criteria are discussed and a relation between singular value disposal and Kaula's rule, the Kaula Singular Value (KSV) relation, is independently introduced, to provide a fifth method of obtaining an optimum stabilized estimate. To select the best gravity field among the five optimal solutions for CHAMP or GRACE given by **PLASS**, each stabilized candidate gravity field is evaluated. It will be concluded that **PLASS** provides a feasible method to obtain alternative solutions for large gravity fields.

PLASS also has the capability to perform parallel inplace covariance matrix calculations as described in Hinga [57] for full rank solutions involving the orthogonalized equations of GRACE. **PLASS** can also compute error propagations in the

form of geoid height errors (referenced to the standard ellipsoid) based on its computation of an SVD covariance matrix. These **PLASS** capabilities were developed as a necessity for solution evaluations.

6.1 Challenging Mini-satellite Payload(CHAMP) 100x100 Gravity Field

6.1.1 Introduction

The CHAMP satellite was launched on July 15, 2000 into an almost circular orbit, with an inclination of 87.2490 degrees and is managed by the GeoForschungsZentrum (GFZ) in Potsdam, Germany. The satellite was injected into an initial altitude of 454 km and was designed to last five years. Because of its low altitude, it can be tracked continuously using the Global Positioning System (GPS) constellation for constant and accurate monitoring of orbital perturbations and is equipped with a high-precision three-axes accelerometer for measuring surface force accelerations. One of its scientific objectives is to determine very precisely the global static Earth gravity field and its temporal variation. The linearized system of equations which relate the model of the CHAMP satellite motion to the GPS observations are solved by forming the normal equation. With this satellite, the n by n normal matrix A is formed to estimate n parameters from m observations in the equation $A = H^T H$, where H is the so called information matrix (which includes the observation weights).

To obtain an alternative gravity solution to the GFZ solution named EIGEN1S, the author's Singular Value Decomposition (SVD) tool **P**arallel **LA**rge **S**vd **S**olver

(PLASS) is applied to the CHAMP normal matrix (**ngl-eigen-1s**) to perform an eigenvalue analysis. The original EIGEN1S solution is based on the Tikhonov regularization method of approximating the ill-conditioned system of equations in a subspace of lower rank. In the alternative solution, based on eigenvalue analysis, poorly determined linear combinations of parameter corrections are removed in the culpable eigenspace of the unconstrained least-squares normal equation. The selection of eigenvalues to be removed can be based upon a method, that was independently introduced for this investigation or four other different common optimization/truncation criteria. The introduced method optimizes the removal of eigenvalues to best satisfy Kaulas rule. The four other techniques are: inspection, relative error, norm-norm minimization (known as the so called L curve), and finding the minimum trace of the mean square error (MSE) matrix. Analysis of five candidate Eigenvalue Decomposition (EVD) gravity fields is performed, and the best are shown to be comparable to the EIGEN1S CHAMP solution obtained by the GeoForschungsZentrum (GFZ).

The number of estimated parameters n is 11216. Since the matrix H corresponds to satellite only observations, and because of rapid attenuation of spherical harmonics with increasing altitude, a high degree of unobservability was expected to cause the normal matrix A to be ill-conditioned. Starting with A, **PLASS** produced five stabilized solutions, along with their covariance matrices and subsequent geoid height error propagations. The five optimal solutions are then presented.

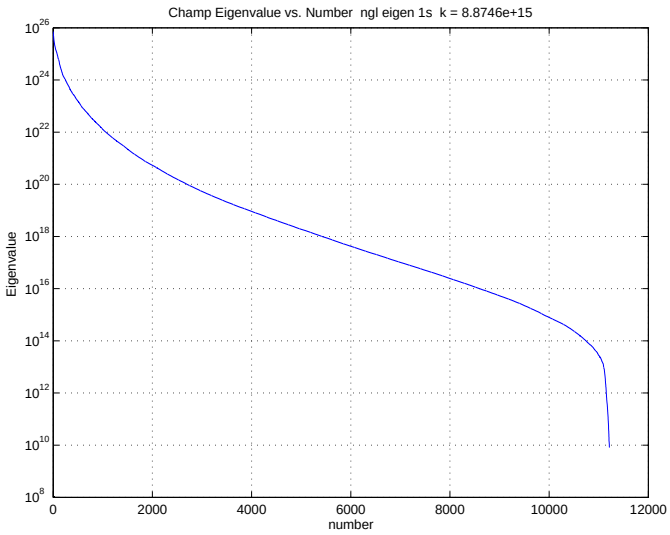


Figure 6.1: CHAMP Eigenvalue vs. Number

6.1.2 Inspection

Before an inspection analysis can be performed, a graph of the eigenvalues versus number (where "number" is the ith eigenvalue), must be obtained. Figure 6.1 is such a graph. The largest and smallest eigenvalues are $1.0x10^{26}$ and $1.0x10^{10}$, respectively, which yields a condition number of $8.87x10^{15}$; indicating an ill-conditioned system of equations. Often when displaying eigenvalues in this manner, they are sorted in either increasing or decreasing size, however in this case no such arrangement was made. The original order of appearance in solution was maintained.

Because the smallest eigenvalue is much greater than zero and there is such a smooth transition throughout most of the graph, it is difficult to know which of the eigenvalues are responsible for the ill-conditioned nature of the normal matrix. It

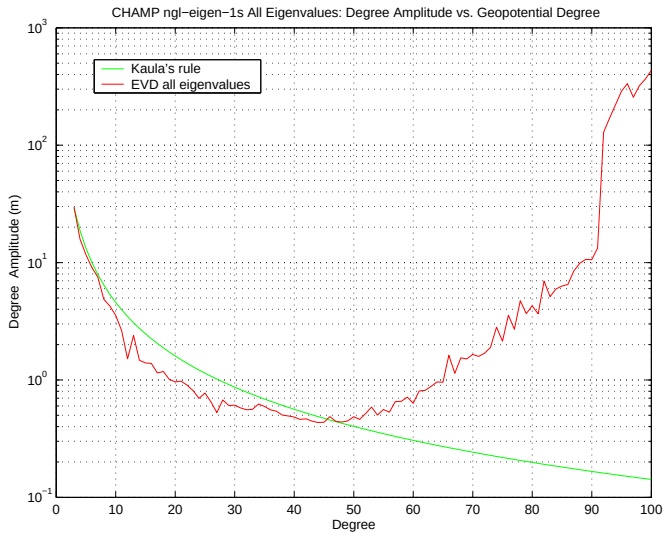


Figure 6.2: CHAMP: All Eigenvalues solution

was hoped that eigenvalues very close to zero would be found, because such quantities are usually guilty of causing harm to a solution. Therefore three guesses had to be made. The first guess was to dispose the smallest 4000 eigenvalues. The next guesses were to eliminate the 6000 and 10000 smallest eigenvalues, with each guess producing a separate solution. For comparison, a solution in which all eigenvalues were retained, was performed. All four solutions are presented in Figures 6.2 - 6.3 as graphs of degree amplitude versus harmonic degree overlayed with Kaula's rule. Clearly, it can be seen from Figure 6.2, that without stabilization from eigenvalue disposal, the estimates of parameters above a harmonic degree of about 50 are quite inflated. But when the eigenvectors which contribute to this inflation (through linear combination in the eigenspace) are eliminated by eigenvalue disposal, their deleterious effects are set to zero, and the solution is improved. Apparently, the guess to remove the 10000 smallest eigenvalues is the better of the three. Because

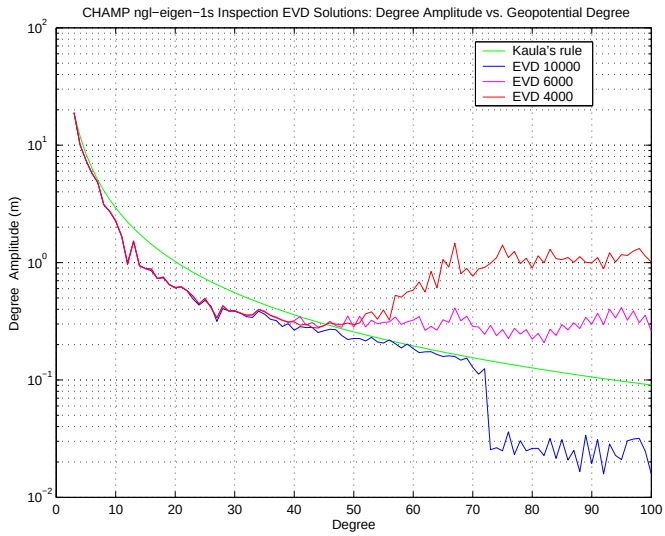


Figure 6.3: CHAMP: Three Inspection Solutions.

the inspection method is subjective and of only marginal value, objective methods are needed to determine how many eigenvalues are to be thrown away, to obtain an optimal EVD solution. The next section describes the results of the Relative Error method.

6.1.3 Relative Error

Because the relative error calculation in Equation 4.36 is an explicit formula, it is a simple matter to evaluate the relative error for all smallest eigenvalue disposal permutations. Since there are 11216 eigenvalues, the same number of relative error scenarios are computed. This leads to the graph of relative error as a percentage versus the number of eigenvalues retained, see Figure 6.4. As expected, when all eigenvalues are thrown away the relative error becomes 100 percent, and zero when all are kept. One may contend that the best relative error is zero percent,

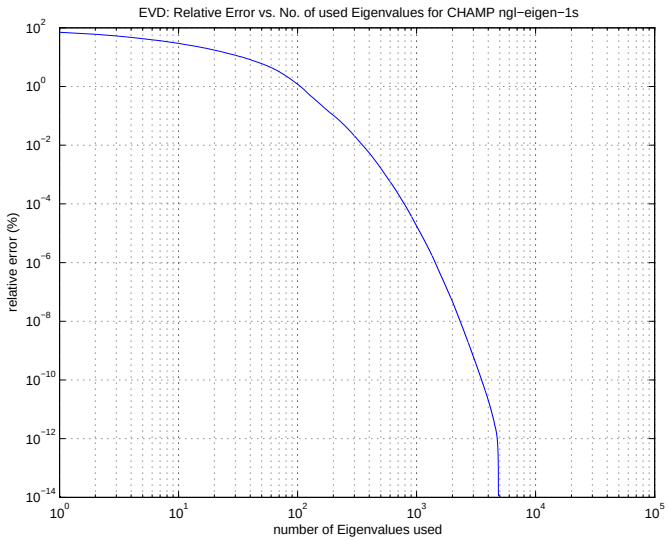


Figure 6.4: Relative Error Graph for CHAMP.

however we are looking for something in between the trivial solutions. That point was determined, from the graphing data, to be at 4871 retained eigenvalues, equivalent to 6345 discarded. Figure 6.5 displays the corresponding degree amplitude versus harmonic degree spectrum, overlayed with Kaula's rule. By comparing the solutions of Figure 6.3 and Figure 6.5, one can see that the relative error formula provides an improved solution that merits further evaluation. Thus the best gravity field obtained from the relative error formula can be submitted as a candidate solution for further gravity field analysis.

6.1.4 Mean Square Error (MSE)

The computation of an MSE solution, requires the accumulation of the eigenspace V and the calculation of the z^2 vector in Equation 4.43. Fortunately, the scaling of the eigenspace matrix by selected Kaula quantities, as coordinated by

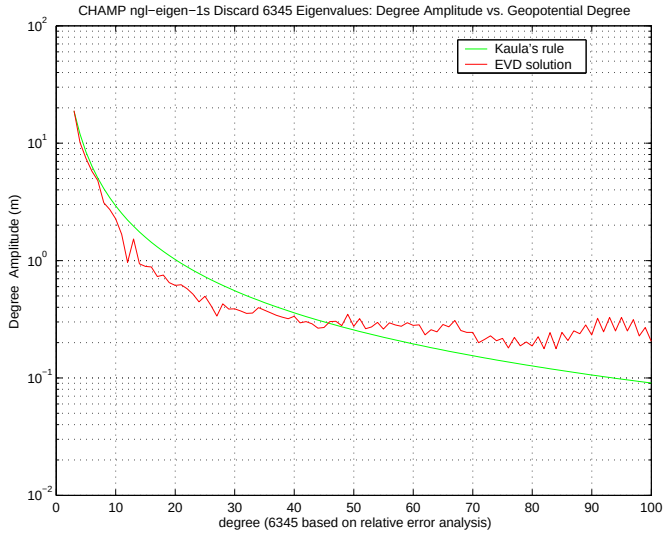


Figure 6.5: Discard 6345 Eigenvalues solution, CHAMP.

the parameter-name-list vector, is carried out only once, before a complete permutation of retained and disposed eigenvalue combinations is initiated. The result is three curves: total noise, total bias as a result of eigenvalue disposal (approximated by Kaula's rule) and mean squared error (MSE). The MSE curve is the addition of the other two curves that are functions of unitless scalar values. The minimum of the MSE function is the optimum combination that minimizes the scalar contribution from both the total noise and total bias. That point is found to occur exactly at the value of 6045 retained or 5171 discarded eigenvalues, see Figure 6.6. Figure 6.7 shows the corresponding gravity solution expressed as degree amplitude versus harmonic degree overlayed with Kaula's rule. It can be seen that in comparison with Kaula's rule, coefficients above harmonic degree 45 are still somewhat inflated and in need of further stabilization. Nevertheless, it will be submitted as a candidate for later gravity field analysis.

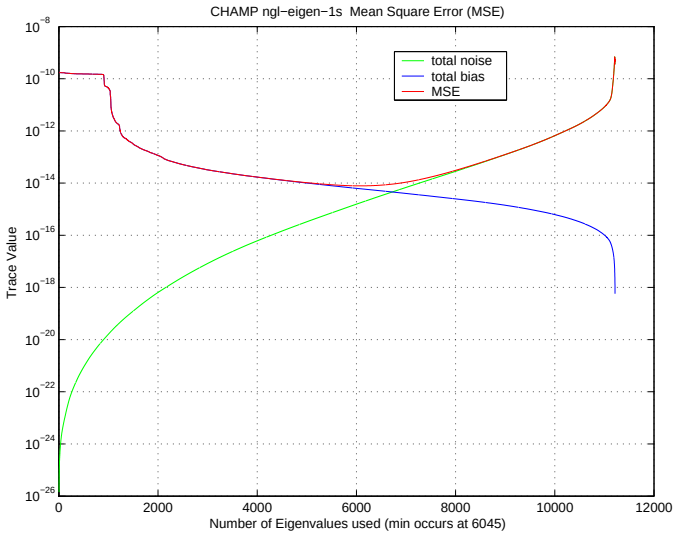


Figure 6.6: CHAMP: Mean Squared Error.

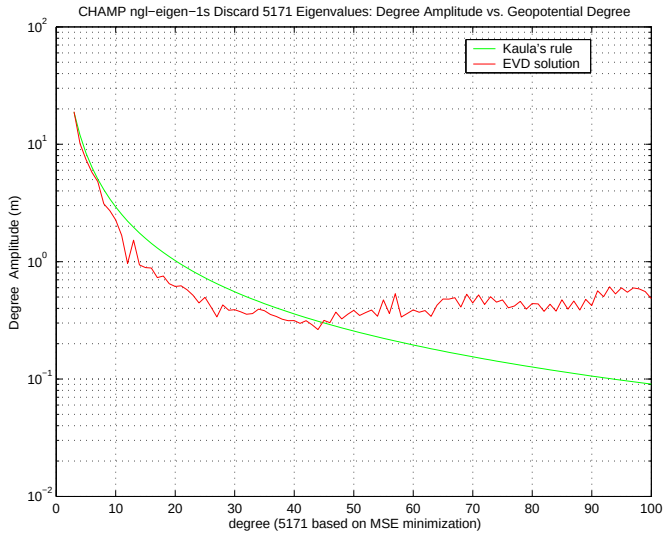


Figure 6.7: CHAMP: Discard 5171 Eigenvalues solution.

6.1.5 Norm-Norm Minimization

The computation of the norm of the residual and the norm of the solution can be carried out with or without the eigenspace matrix. If the eigenspace matrix is not desired and only solutions are needed, then **PLASS**1 should be used. **PLASS**2 should be applied, if other stabilization schemes are to be tested and error analyzes carried out. In this norm-norm minimization experiment, **PLASS**2 was used to produce 11216 different solutions, enabling a complete sweep of all permutations in the calculation of solution corrections $\|\hat{x}\|_2 = \|\widetilde{A}^{-1}\overline{b}\|_2$, as seen in Equation 4.37, and residuals $\|\overline{\rho}\|_2 = \| \widetilde{A}\hat{x} - \overline{b} \|_2$ as seen in Equation 4.38, for the number of eigenvalues retained. For example, as each eigenvalue was individually disposed, a separate solution correction vector and residual was calculated for that scenario. The norm of each of the correction vectors and residuals were computed and plotted. Using one processor to perform a complete sweep of eigenvalue disposal for large problem norm-norm analysis is prohibitive due to the enormous amount of computer time and memory required. However, given enough processors, **PLASS** eliminates these obstacles and completes the analysis in a matter of a few hours. Figure 6.8 is a log-log plot of the norm of the residual versus the norm of the correction vector for all possible scenarios. The point on the norm-norm curve that is closest to the origin is where the optimal balance between these two norms is found. It is the point where the square root of the sum of the squares of the elements of the residual vector, $\overline{\rho}$ and solution vector $\hat{x}$ is a minimum. This point occurs in the case where 8373 eigenvalues are thrown away. Since the optimum suggested by the norm-norm graph corresponds to a gravity field *correction*, this correction must be added to the

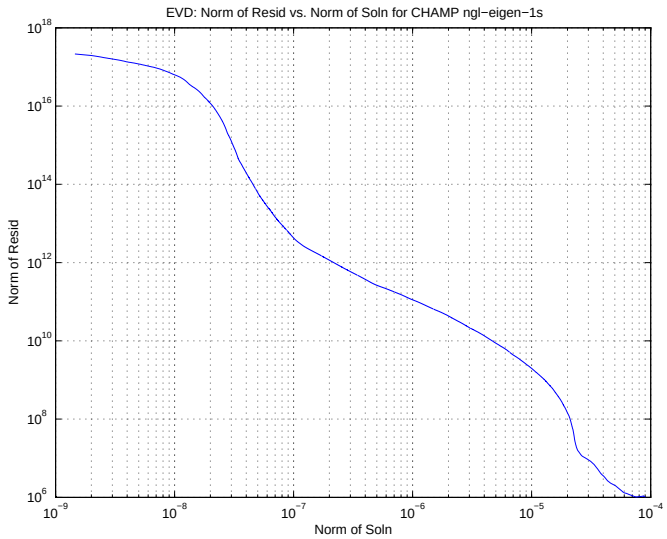


Figure 6.8: CHAMP: Norm of Resid. vs Norm of Soln.

nominal (*a priori*) gravity field for evaluation. Figure 6.9 shows the corresponding updated gravity solution expressed as degree amplitude versus harmonic degree overlayed with Kaula's rule. It can be seen that the power in the estimated coefficients which causes their inflation above Kaula's rule has been removed. Since there are 11216 eigenvalues, we have discarded 75 percent of the information contained in the normal matrix "ngl-eigen-1s". It may be that too many eigenvalues have been removed from this normal matrix to obtain a solution, however it will be submitted as a candidate for gravity field analysis. The next stabilization technique involves Kaula's rule as a reference to eigenvalue removal for solution, which may lead to more information retention.

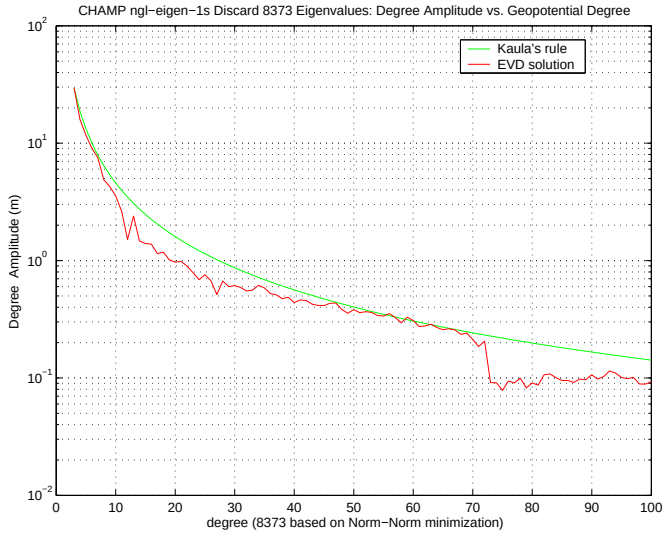


Figure 6.9: CHAMP: Discard 8373 Eigenvalues solution.

6.1.6 Kaula Eigenvalue (KEV) Relation

Using **PLASS**2, and the eigenspace matrix generated from the norm-norm analysis, every case of discarding one eigenvalue at a time was considered when implementing the Equation 4.47, repeated here for convenience,

$$\alpha(k) = \left\{ \sum_{l=1}^{lmax} \left[v(l)_{evd(k)_{l,m}} - v(l)_{Kaula} \right]^2 \right\}^{\frac{1}{2}}. \tag{6.1}$$

The subscript “evd” indicates Eigenvalue Decomposition (as called for by a normal matrix). The dimensionless term $\alpha(k)$ is plotted versus number (k) of eigenvalues used. There are a total of 11216 alpha values along the generated function and it was expected that there would be one global minimum which best satisfies Kaula’s rule. Figure 6.10 shows the entire search for that minimum and Figure 6.11 displays a magnified view of the area where the global minimum occurs, which is the point where 7581 eigenvalues were discarded (3635 are kept). Figure 6.12 shows

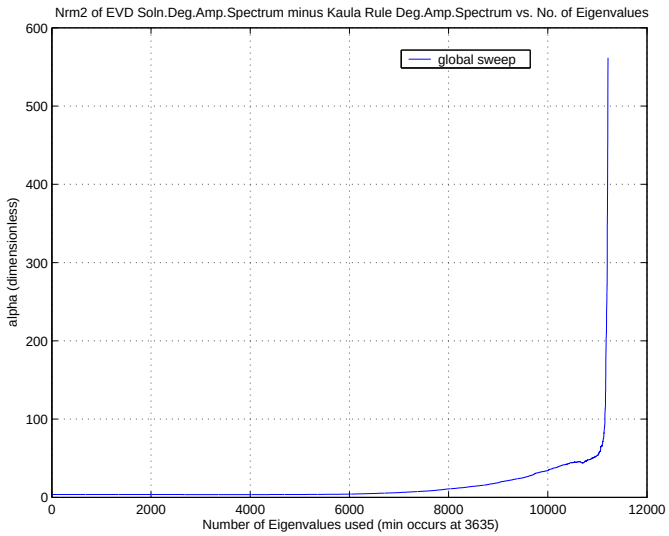


Figure 6.10: CHAMP: Global Sweep of Eigenvalue Disposal Cases.

the corresponding updated gravity solution expressed as degree amplitude versus harmonic degree overlayed with Kaula's rule. In this case fewer eigenvalues were discarded and 7 percent more of the information in the normal matrix was retained, in comparison to the best norm-norm solution. Clearly, Figure 6.12 fulfills the expectation that this solution follows Kaula's rule better than the previous four best gravity fields. It is only with further evaluation however, can the best of the five fields be selected.

6.1.7 Evaluation of Gravity Fields

Because it is not clear which of the five candidate gravity fields is the best, evaluations based on error propagation from the computation of geoid height errors, orbit fit calculations, geoid comparisons, and degree error variance analyzes are performed. It must be remembered however, that the covariance matrix corre-

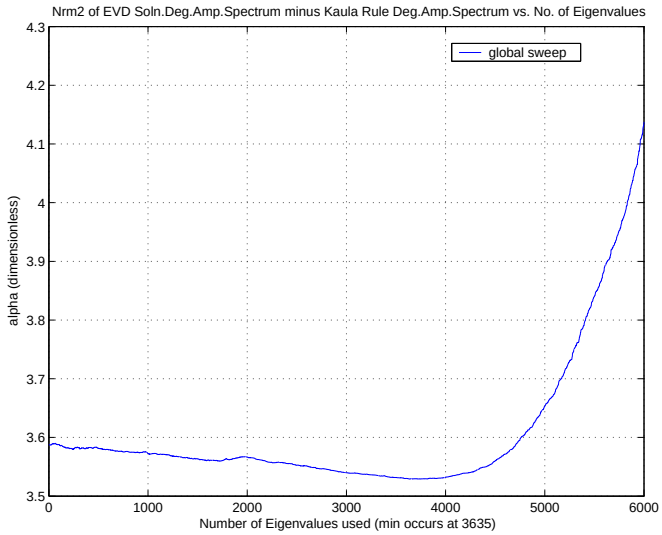


Figure 6.11: CHAMP: Global Minimum of Used Eigenvalue Cases.

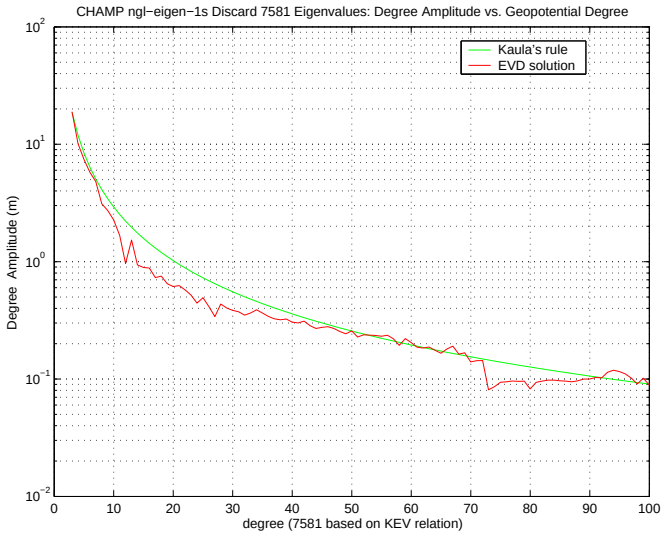


Figure 6.12: CHAMP: Discard 7581 Eigenvalues Solution.

Method	Optimum Number of Used Eigenvalues
Inspection	1216
Relative Error	4871
MSE	6045
Norm Norm minimization	2843
KEV relation	3635

Table 6.1: Five Candidate Gravity Fields to be Evaluated

sponding to an Eigenvalue Decomposition (EVD) or Singular Value Decomposition (SVD) estimated solution, is not an adequate measure of error for an estimate if any eigenvalues or singular values have been discarded. Because eigenvalues (in the case of CHAMP) have been set to zero, their eigenvectors were not included into the estimation process. This causes the estimate to be biased, meaning that the calculated gravity field is "shifted" by some amount away from the true gravity field and the confidence in the estimated coefficients may be too optimistic, i.e. perhaps their variances are tighter than variances from unbiased estimates. However, if the bias introduced by an EVD/SVD estimate is "small", their estimates may be considered unbiased in an approximate sense. This will be discussed after the best gravity field has been chosen. Table 6.1 summarizes the used eigenvalues in each candidate gravity field and for convenience, Figure 6.13 presents the apparent "best" two candidates in a common graph overlayed with Kaula's rule. Clearly, as eigenvalues are discarded, the solution becomes more stabilized, in the sense that the power in the coefficients causing inflation is removed. The next section will test these gravity fields by using them to model the motion of selected Earth satellites, comparing their predicted states with actual observations and reporting the results in the form of orbital fit residuals.

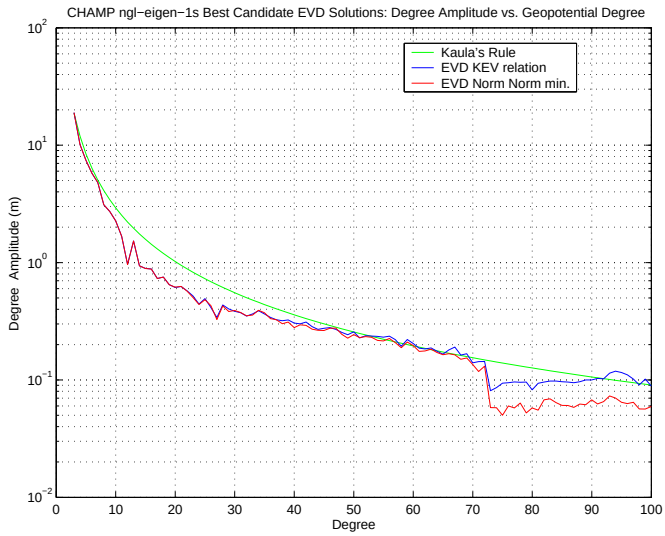


Figure 6.13: CHAMP: Apparent Best Two Candidate Solutions.

Satellites	Semi-major axis (km)	Inclination (deg.)	Eccentricity
Topex/Poseidon	7705	65.99	0.0010
Starlette	7335	49.81	0.0206
Stella	7200	98.00	0.0001
Lageos 1	12271	109.83	0.0040
Lageos 2	12162	52.63	0.0137
GFZ1	6764	51.64	0.0073

Table 6.2: Orbital Elements of Selected Satellites

6.1.7.1 Orbital Arc Fit Computations

The satellites selected to fly through the estimated EVD gravity fields are shown in Table 6.2. This group of satellites were chosen because they represent a good sample of inclinations and altitudes necessary for an adequate orbital fit test. Satellite GFZ1 is used to test the higher degrees/orders because of its low altitude. Starlette, Stella, and Topex are the satellites used to assess the somewhat lower degrees/orders of an estimated gravity field, because of their higher altitudes. To

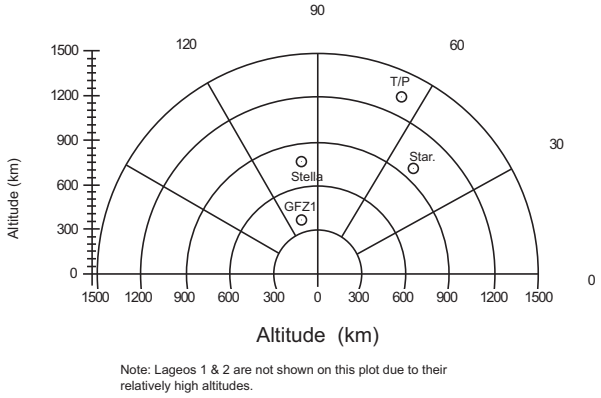


Figure 6.14: Selected Satellites for Orbit Fit Analysis.

scrutinize the quality of the low degrees/orders, the satellites Lageos 1 and 2 were chosen for their high altitudes. Figure 6.14 illustrates schematically their orbital heights above the Earth. Each orbit fit of the five EVD gravity fields was computed using UTOPIA (Tapley et al. [68]) and compared with the actual observation data for a chosen satellite. For uniformity in the orbit fit calculations, all of the gravitational and non-gravitational perturbations of the orbit estimations in the UTOPIA runs are kept consistent for each case and each test satellite. The gravitational perturbations are those due to the non-spherical geopotential of the Earth, sun, moon, other planets, the solid Earth tides, ocean tides, rotational deformation, and general relativity. The non-gravitational perturbations are those due to atmospheric drag, solar radiation pressure, Earth radiation pressure and other unknown forces. Since the epoch of the CHAMP EVD fields is years later than the epochs of the satellites, J2, J3 and J4 were propagated to a common epoch. Table 6.3 contains the arc lengths, number of arcs, and satellite epochs used in the UTOPIA runs. Table 6.4 shows the SLR (Satellite Laser Range) orbit fits in centimeters RMS (radial) of all

Satellites	Arc length (days)	Number of Arcs	Epoch
Topex/Poseidon	10	1	June 1999
Starlette	5	6	Jan. 1998
Stella	5	6	Sept.1993
Lageos 1	3	10	May 2002
Lageos 2	3	10	May 2002
GFZ1	3	7	Aug. 1995

Table 6.3: Observation Data used in UTOPIA Runs

five candidate gravity fields on all selected satellites. Notice, that for the Inspection case (1216 eigenvalues used), all satellites fall out of orbit, causing the test to fail. Therefore this case can be eliminated as a candidate gravity field, while the others remain in contention. The case EIGEN1S is the gravity field produced by the GFZ from the same CHAMP normal matrix "ngl-eigen-1s" of this investigation, but stabilized using a different technique. It is this EIGEN1S gravity field to which the four remaining EVD fields are compared. Other than the GFZ1 satellite, the RMS values are fairly similar for all cases of the EVD gravity fields and the EIGEN1S. It is not until using the test satellite GFZ1, with a low altitude, does the effect of eigenvalue disposal deliver a considerable influence to orbital fit residuals. Because the GFZ1 satellite is at a lower altitude, the EVD deflation effects on the poorly observed higher degree and order coefficient corrections, can be perceived.

6.1.7.2 EVD Degree Error Variance and Geopotential Variance Difference vs. EIGEN1S

In the previous section, one gravity field was eliminated, because it caused all satellites to fall out of orbit in every flight simulation. By comparing the formal degree error variances of each remaining candidate gravity field with that of

Case	GFZ1	Lageos 1	Lageos 2	Starlette	Stella	Topex
Inspection	crash	crash	crash	crash	crash	crash
Eigen-Kaula	11.52	8.12	10.77	3.08	3.64	2.32
MSE	15.65	10.77	10.77	2.95	3.64	2.32
Norm-Norm	11.34	8.13	10.77	3.08	3.64	2.39
Relative Error	11.22	8.12	10.77	2.97	3.64	2.34
EIGEN1S	74.03	8.11	10.76	3.07	3.31	2.37

Table 6.4: Orbital Arc Fits of Candidate Gravity Fields (cm. radial RMS)

EIGEN1S, further objective elimination of solutions is possible. The formal degree error variance is a measure of the power of the geopotential errors for a specific harmonic degree and is represented as the sum of the squares of the formal variance of the estimated gravity coefficients at a particular harmonic degree l. It is expressed in Equation 6.2 as

$$\sigma_l^2 = \sum_{m=1}^{l_{max}} \left(\sigma^2_{\overline{C}_{lm}} + \sigma^2_{\overline{S}_{lm}} \right), \tag{6.2}$$

where $\sigma_{\overline{C}_{lm}}$ and $\sigma_{\overline{S}_{lm}}$ are the standard deviation of the normalized estimated geopotential coefficients for degree l and order m. The value σ_l^2 is then scaled into millimeters of height and plotted as a function of degree amplitude versus harmonic degree. To generate degree amplitude difference curves for the power spectrum differences between the estimated coefficients of the EVD and EIGEN1S, the degree variance difference $\triangle_l^2$ is calculated. This relation is a measure of the power of the geopotential differences between two gravity fields at a specific harmonic degree l, and is expressed in Equation 6.3 as,

$$\triangle_l^2 = \sum_{m=1}^{lmax} \left(\triangle^2_{\overline{C}_{lm}} + \triangle^2_{\overline{S}_{lm}} \right), \tag{6.3}$$

where $\triangle_{\overline{C}_{lm}}$ and $\triangle_{\overline{S}_{lm}}$ are the scalar differences of the normalized geopotential coefficients for a specific harmonic degree l and order m.

Since every candidate gravity field solution is biased (because eigenvalues are discarded), the EVD error covariance matrix contains only random error, as mentioned in Section 4.1.1. However, the variances of a candidate gravity field and of a reference field (which is unbiased) may be compared to evaluate the difference between the two fields with respect to the error variance of the reference field. Figure 6.15 shows the degree error variance of the KEV solution in comparison to that of the EIGEN1S reference field. The differences between their geopotential coefficient variance spectra in the form of geopotential power per harmonic degree is shown. It can also be seen, that the formal error variances of the biased KEV solution are all within the error variances of EIGEN1S, implying that this candidate EVD field is within the uncertainty (in a random sense) of the EIGEN1S gravity field. Figures 6.16, 6.17 and 6.18 illustrate the same information for the candidate gravity fields produced by the eigenvalue analyzes of the MSE, norm-norm minimization, and the relative error criteria, respectively. These plots show that all degree error variances of the four EVD solutions are within the degree error variances of EIGEN1S. However, because the degree variance differences to EIGEN1S in the cases of MSE and Relative Error become excessive with harmonic degree, these two candidate fields are excluded from further consideration. At this point we have eliminated three of five EVD gravity fields. In the next section, we compare the geoids generated from the remaining fields to that of EIGEN1S.

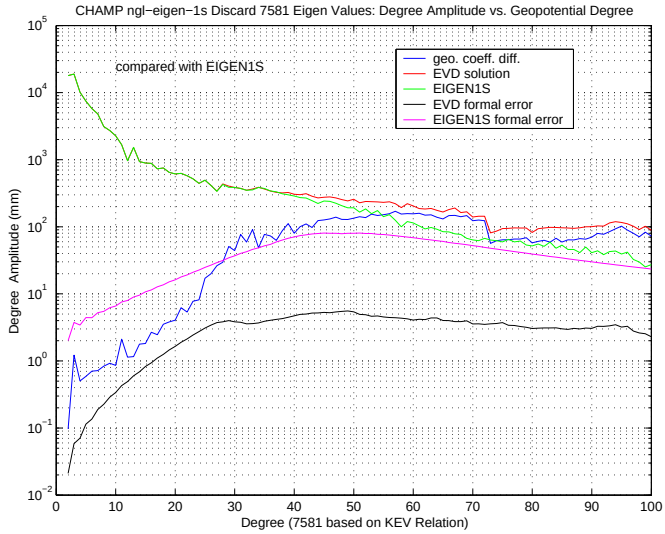


Figure 6.15: CHAMP:KEV Degree Error Var. and Var. Geopotential Difference to EIGEN1S

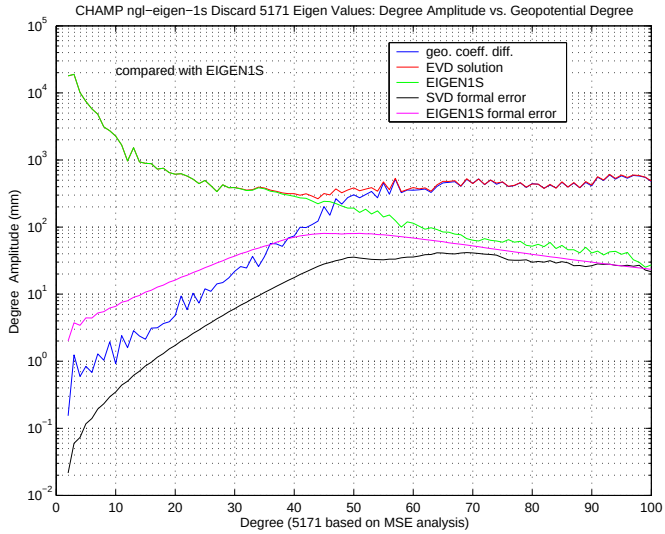


Figure 6.16: CHAMP:MSE Degree Error Var. and Var. Geopotential Difference to EIGEN1S

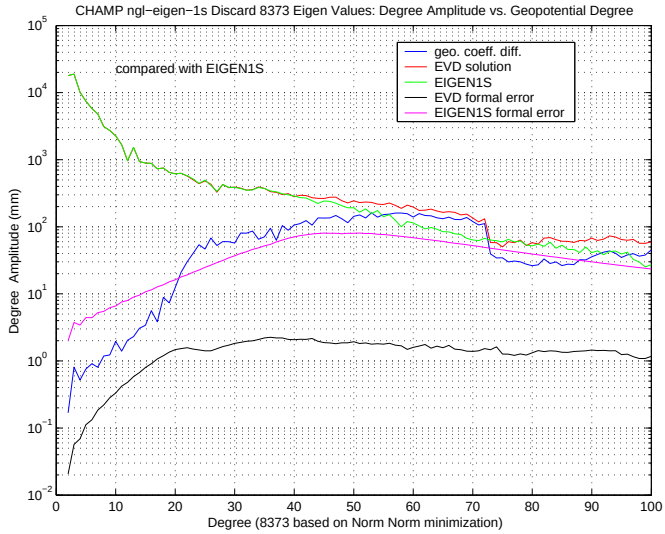


Figure 6.17: CHAMP:Norm Degree Error Var. and Var. Geopotential Difference to EIGEN1S

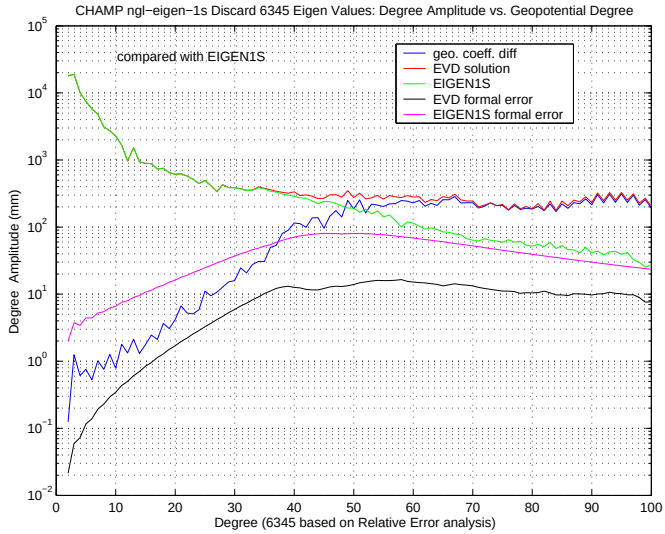


Figure 6.18: CHAMP:Rel.Error Degree Error Var. and Var. Geopotential Difference to EIGEN1S

6.1.7.3 EVD Geoid Differences to EIGEN1S

The simple geoid undulation above the Earth's spheroid of radius a_e is computed as (Condi [72])

$$N = a_e \sum_{l=2}^{l_{max}} \sum_{m=0}^{l} \overline{P}_{l,m}(\sin\phi)[\Delta\overline{C}_{l,m}\cos m\lambda + \Delta\overline{S}_{l,m}\sin m\lambda] \tag{6.4}$$

where $\Delta\overline{C}_{l,m}$ and $\Delta\overline{S}_{l,m}$ are the differences of the normalized geopotential coefficients between two gravity fields. Plotting N, then reveals the geoid undulation differences.

Figure 6.19 shows the geoid undulation map of the CHAMP EIGEN1S field of the GFZ. The distinctive pattern of low regions south of India, Antarctica and off the West and East coasts of North America can be seen, as well as the high regions of the North Atlantic and East South Pacific. To illustrate the incremental effects (to the geoid undulation differences with EIGEN1S) of eigenvalue disposal, the eliminated MSE EVD gravity field in Figure 6.22 is presented as the first geoid undulation difference contour plot, as it is the case of the least amount of eigenvalues removed, namely 5171. Over the majority of the Earth, both land and ocean, the differences undulate roughly between -6 cm and 7 cm. Only at the poles can wider differences be seen. Discarding another 1174 eigenvalues in the Relative Error case (cumulative total of 6345), Figure 6.23 indicates a very similar global pattern, except that the differences oscillate between about -1.3 cm and 7 cm. The removal of yet another 1236 eigenvalues, and an additional 792, seen in Figures 6.24 and 6.25 respectively, has the effect of tightening this pattern to produce a fluctuation

between about -1 cm and 2 cm, while also emphasizing a Himalayan anomaly first revealed in the EIGEN1S.EVD.7581 case.

Table 6.5 summarizes the extrema and mean geoid undulation values of the GFZ reference field EIGEN1S and the EVD gravity fields, except for the inspection case (EIGEN1S.EVD.10000), which fails to generate a geoid. This is not surprising, as this case also caused all test satellites to fall out of orbit during orbit fit evaluation. Only the geoids from the KEV and norm-norm minimization solutions are presented in Figures 6.20 and 6.21, respectively, as they are the best solutions. Table 6.6 summarizes the extrema and mean values of the differences between the GFZ EIGEN1S and these EVD gravity fields. Although the EVD solutions EIGEN1S.EVD.5171 and EIGEN1S.EVD.6345 have been eliminated from further contention, they are included in this table to illustrate the incremental effects of eigenvalue disposal in the geoid differences. It can be seen that the spread between the maximum and minimum geoid points for each case of Table 6.6 is reduced as the number of removed eigenvalues increases. This is another indication that the EVD solution continues to improve with eigenvalue removal. Of course if too many eigenvalues are removed (case EIGEN1S.EVD.10000) the estimated coefficients may be nonsensical, even though inflation has been removed and their degree variance power spectrum falls below Kaula's rule. To propagate the uncertainties of the EVD gravity fields, **PLASS**2 was tasked to compute the geoid height errors for every calculated EVD covariance matrix. To compute the geoid height errors of the EIGEN1S solution, a given covariance matrix, supplied by the GFZ, was input to **PLASS**2.

Number of geoid points computed: 64,800
Parameters of the mean Earth ellipsoid used in geoid undulation computations:
GM $= 3.986004415$ x $10^{14} \frac{m^3}{s^2}$, $a_e = 6.37813630$ x 10^6 m
f $= \frac{1}{298.2570}$, $\omega_e = 7.29215$ x $10^{-5} \frac{rad}{sec}$
$\diamond$ = eliminated, all table entries: units in cm.

Gravity Fields	Max. Height	Min. Height	Mean Value of Geoid Undulation
EIGEN1S	79.82	-104.83	-0.8147
EIGEN1S.EVD.5171$^\diamond$	83.89	-110.13	-0.8178
EIGEN1S.EVD.6345$^\diamond$	82.58	-105.83	-0.8217
EIGEN1S.EVD.7581	83.06	-105.44	-0.8313
EIGEN1S.EVD.8373	82.33	-104.94	-0.8096

Table 6.5: Geoid Undulations of Gravity Fields

Number of geoid points computed: 64,800
Parameters of the mean Earth ellipsoid used in geoid undulation computations:
GM $= 3.986004415$ x $10^{14} \frac{m^3}{s^2}$, $a_e = 6.37813630$ x 10^6 m
f $= \frac{1}{298.2570}$, $\omega_e = 7.29215$ x $10^{-5} \frac{rad}{sec}$
$\diamond$ = eliminated, all table entries: units in cm.

Gravity Fields	Max. Height	Min. Height	RMS about Mean
EIGEN1S - EIGEN1S.EVD.5171$^\diamond$	32.43	-30.84	5.3713
EIGEN1S - EIGEN1S.EVD.6345$^\diamond$	25.94	-19.46	3.3128
EIGEN1S - EIGEN1S.EVD.7581	8.193	-7.500	1.0885
EIGEN1S - EIGEN1S.EVD.8373	8.590	-7.170	0.9302

Table 6.6: Geoid Undulation Differences of EVD Gravity Fields to EIGEN1S

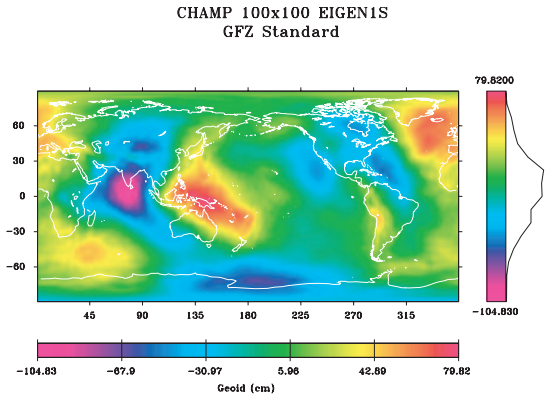


Figure 6.19: CHAMP: GFZ Standard EIGEN1S Geoid

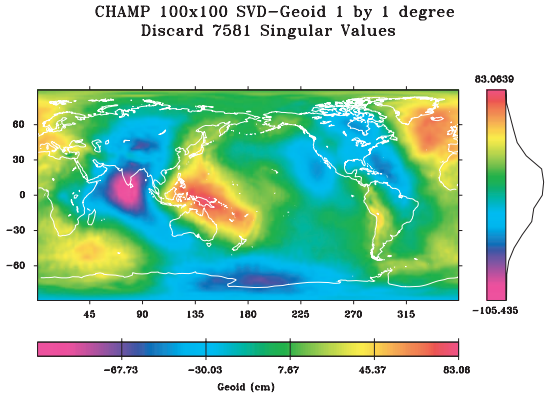


Figure 6.20: CHAMP: Geoid EIGEN1S.EVD.7581

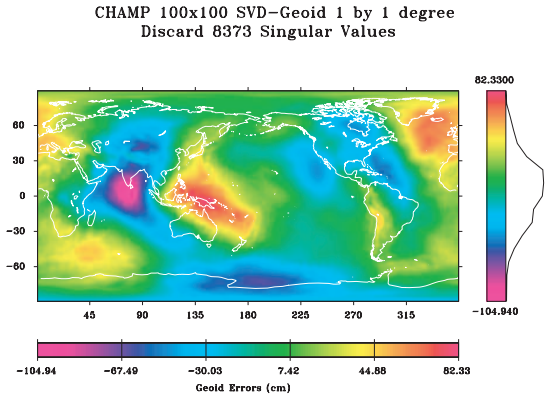


Figure 6.21: CHAMP: Geoid EIGEN1S.EVD.8373

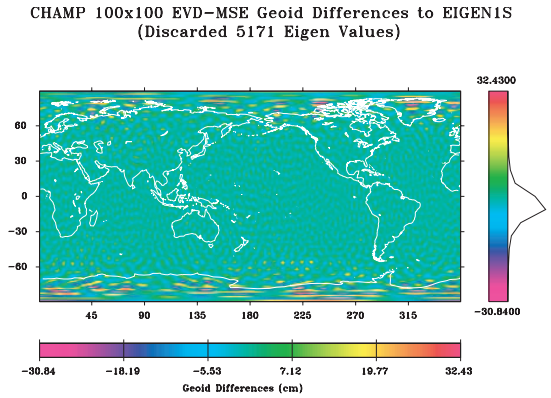


Figure 6.22: CHAMP: Geoid Diff: EIGEN1S-EIGEN1S.EVD.5171

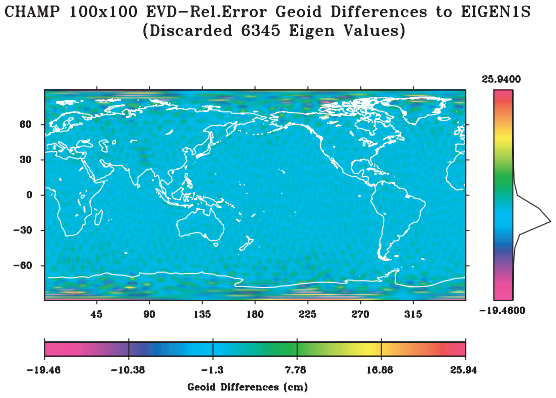


Figure 6.23: CHAMP: Geoid Diff: EIGEN1S-EIGEN1S.EVD.6345

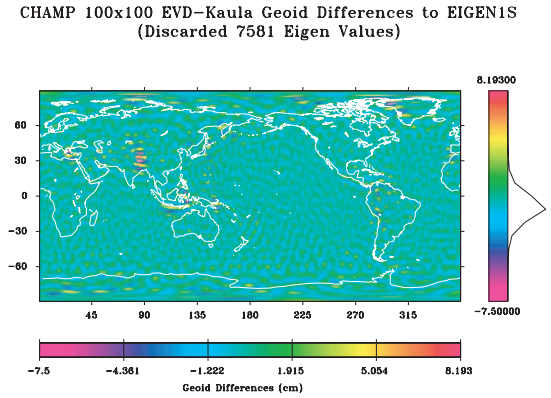


Figure 6.24: CHAMP: Geoid Diff: EIGEN1S-EIGEN1S.EVD.7581

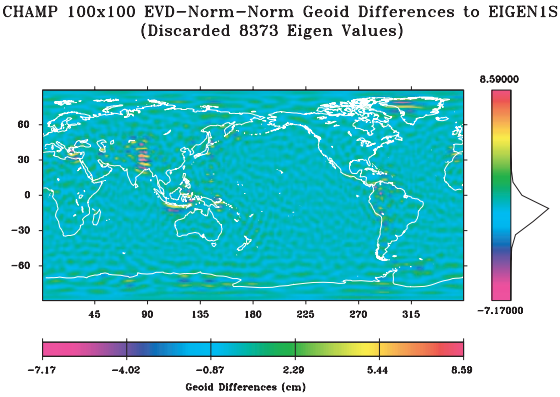


Figure 6.25: CHAMP: Geoid Diff: EIGEN1S-EIGEN1S.EVD.8373

6.1.7.4 Geoid Height Errors of the EVD and EIGEN1S Gravity Fields

To assess the error of a particular gravity solution, it is necessary to compute the covariance matrix, also known as the variance-covariance matrix. It is the diagonal elements of this matrix that convey the statistical confidence of each parameter in an estimated solution. The off diagonals represent their covariances and can be used to derive all correlations among them. The covariance matrix can be used to support “goodness of fit” evaluations or error propagations. In this investigation the covariance matrix is used in an error propagation known as the geoid height error. Such error calculations are the mapping of all estimation uncertainties into a space called the geodetic surface of the Earth.

The calculation of the geoid height errors is traditionally (i.e. with a single processor) carried out by forming the Equation 6.5 (Schrama [73]) which propagates all of the error in the nxn covariance matrix, C, where n is the number of estimated parameters, to one single location on the surface of the globe (reference spheroid), i.e. to a particular degree latitude and longitude. This mapping is ex-

pressed as,

$$\alpha = v^T C v, \tag{6.5}$$

where α is a 1x1 scalar, and v is an nx1 column vector containing the paired elements (a_e[cosmλ]$\overline{P}_{l,m}$(sinϕ)) and (a_e[sinmλ]$\overline{P}_{l,m}$(sinϕ)). The term $\overline{P}_{l,m}$(sinϕ) is the associated normalized Legendre polynomial, a harmonic function of degree l and order m. a_e is the radius of the spheroid Earth having the geodetic angles $\phi =$ latitude, and $\lambda =$ longitude. The organization of these paired terms is the same as the order in which the estimated parameters appear in the covariance matrix C. By multiplying the variance and covariance elements inside matrix C against the identically ordered elements of vector v and then adding them together, as dictated by linear algebra, all uncertainties in the estimate of a gravity field are mapped from the spherical harmonic space and propagated into the space of the geodetic surface of the Earth, namely onto latitude and longitude locations. Obviously, the values inside the vector v must be re-calculated for every change in longitude λ or latitude ϕ. Computing such an equation using only one processor consumes enormous amounts of time, i.e. potentially days of wall clock time for CHAMP or GRACE size gravity fields. Fortunately, the above sequential equation may be re-organized into a form which allows one to bring into bear the power of parallel computation. **PLASS** has this capability and computes the geoid height errors in the following manner.

Instead of performing the calculation where v is an nx1 vector, many columns of matrix D (of Equation 6.6), each of which is a different instance of the vector v, is formed. Therefore 360 points of longitude λ, at a particular latitude ϕ, is contained

in 360 columns of matrix D, with dimension nx360. Multiplying the right side of covariance matrix C with D, produces the matrix B, namely

$$B = CD, \tag{6.6}$$

where B is nx360 and C is nxn. To finish this example calculation of geoid height errors, for a particular latitude ϕ, we must carry out a certain operation of D^T against the left side of matrix C, which is **not** a matrix multiplication. The operation is defined by taking the transpose of the ith column in D

$$\left(\begin{bmatrix} D_1 \end{bmatrix} \quad \begin{bmatrix} D_2 \end{bmatrix} \quad \cdots \quad \begin{bmatrix} D_{360} \end{bmatrix} \right) \Longrightarrow \text{ transpose } \Longrightarrow \begin{bmatrix} [D_1]^T \\ [D_2]^T \\ \cdots \\ [D_{360}]^T \end{bmatrix} \tag{6.7}$$

and performing a dot product with the ith column of B,

$$\left(\begin{bmatrix} B_1 \end{bmatrix} \quad \begin{bmatrix} B_2 \end{bmatrix} \quad \cdots \quad \begin{bmatrix} B_{360} \end{bmatrix} \right). \tag{6.8}$$

This yields the ith scalar value for the ith geoid height error at the ith point, i.e. (ϕ, λ) on the reference spheroid, and can be expressed as,

$$\alpha_{\phi,\lambda} = [D_i]^T \cdot \begin{bmatrix} B_i \end{bmatrix}. \tag{6.9}$$

Performing this calculation 360 times, once around the globe, results in a row array (not a vector) of squared geoid height errors with dimension 1x360,

$$\left[\alpha_{\phi,1}, \alpha_{\phi,2}, \alpha_{\phi,3}, \quad \cdots \quad , \alpha_{\phi,360}\right], \tag{6.10}$$

for a particular latitude ϕ. The square root of each term is then taken to get the final form of the propagated error for each location. If we increase the number of

columns in matrix D by the multiple k, where k is the number of latitude points, then matrix D and B assume the dimensions of n x 360k. In the end, this leads to a one dimensional array of geoid height errors that has length of 360k, where each element corresponds to one particular latitude and longitude. **PLASS** can execute these calculations in one step per latitude, propagating the uncertainties once around the globe in one step, or if the user desires and if **PLASS** is provided with enough processors, it can compute all 64,800 points of latitude and longitude geoid height errors in one step.

To provide a proper reference for the geoid height errors that correspond to each of the EVD gravity fields, the covariance matrix of the standard GFZ EIGEN1S gravity field was propagated using **PLASS** to a global grid resolution of one degree of latitude and longitude. (Note: The calculation of the geoid height errors was carried out at a 1^ox1^o grid resolution although the 100 degree and order CHAMP gravity field contains only enough gravity information to support a (ϕ, λ) grid resolution of 5 degrees. Therefore, it should be remembered that the grid resolution is actually 5^ox5^o in a physical sense.)

Figure 6.26 shows the geoid height error reference calculated at a grid resolution of 1^ox1^o. The eliminated gravity fields of cases EIGEN1S.EVD.5171, and EIGEN-1S.EVD.6345, and EIGEN1S.EVD.10000 are included in this section to demonstrate the incremental effects of eigenvalue removal to the calculated geoid height errors. Comparing the case which contains the most number of used eigenvalues, Figure 6.27, to that of the EIGEN1S propagated error, Figure 6.26, it can be seen that the disposal of eigenvalues removes geoid height error but does not affect the

overall pattern of propagated error. Very similar zonal patterns exist in both plots, only in the case EIGEN1S.EVD.5171, they are weaker. Careful examination of the zonal band between 30^o and -30^o latitude, reveals a faint vertical pattern which may indicate the inclination of the CHAMP satellite. The tightening of the spread between maximum and minimum geoid undulation differences, seen in the previous section, occurs as a result of eigenvalue removal. Figures 6.28, 6.29, 6.30, and 6.31 illustrate a similar trend of tightening in **geoid height errors** as more and more eigenvalues are thrown away. Although, the errors appear to be declining in these figures, one should bear in mind that the bias introduced from the exclusion of eigenvalues is not being represented. These plotted geoid height errors are due to random effects only and may be overly optimistic and possibly too far away from those of a non-biased solution that is near the truth. For the case of EIGEN1S.EVD.10000, its orbit fit analysis was fortunate enough to fail, thus exposing the fact that is was too far from the true solution. Although many eigenvalues were truncated in the cases of EIGEN1S.EVD.7581 and EIGEN1S.EVD.8373, they may still be considered as un-biased estimates in an approximate sense. This issue will be discussed in Section 6.1.7.5. Table 6.7 summarizes the propagated error extrema seen in the geoid height error contour plots and the parameters used to calculate them.

6.1.7.5 The KEV and Norm-Norm EVD Bias

The SVD bias assessment technique described in Section 4.1.3 can be applied to quantify bias introduced from an EVD solution. Three different bias vectors

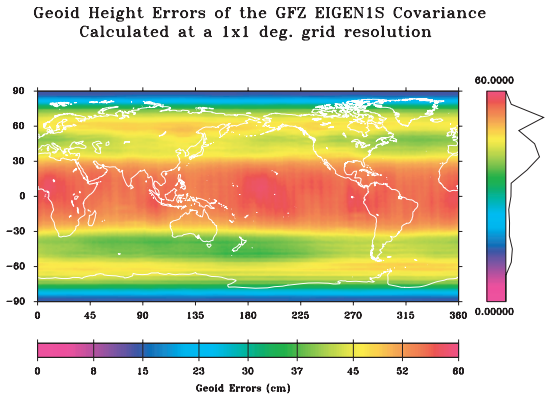


Figure 6.26: CHAMP: Geoid Height Errors GFZ EIGEN1S

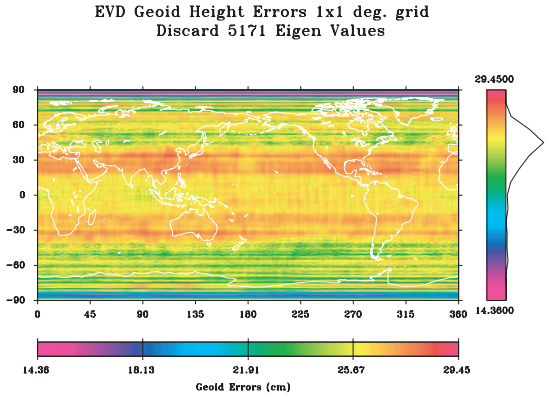


Figure 6.27: CHAMP: Geoid Height Errors EIGEN1S.EVD.5171

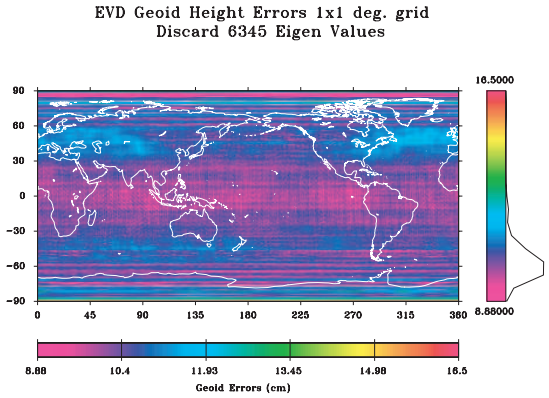


Figure 6.28: CHAMP: Geoid Height Errors EIGEN1S.EVD.6345

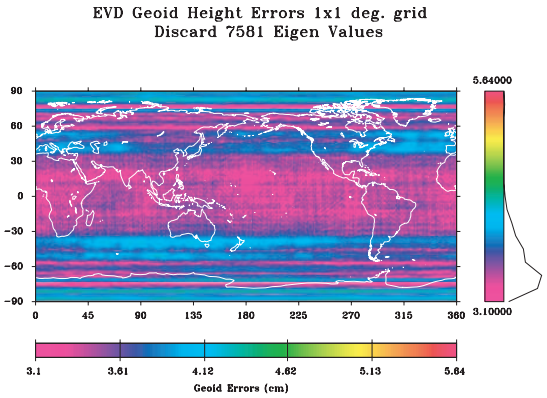


Figure 6.29: CHAMP: Geoid Height Errors EIGEN1S.EVD.7581

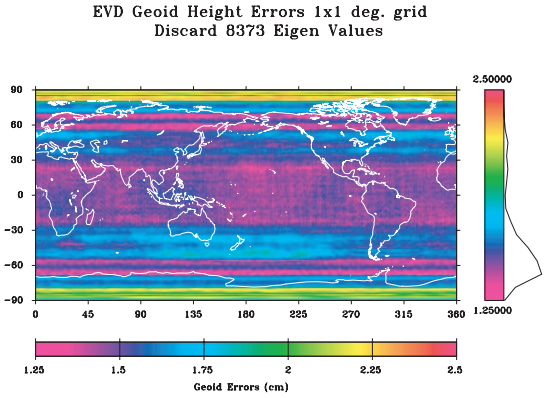


Figure 6.30: CHAMP: Geoid Height Errors EIGEN1S.EVD.8373

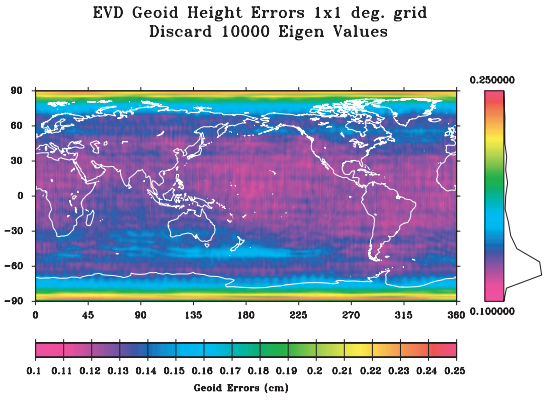


Figure 6.31: CHAMP: Geoid Height Errors EIGEN1S.EVD.10000

Number of geoid height error points computed: 64,440 Parameters used to map uncertainties to geodetic surface: GM $= 3.986004415$ x $10^{14} \frac{m^3}{s^2}$, $R_e = 6.37813630$ x 10^6 m $\diamond$ = eliminated, all table entries: units in cm.			
Gravity Fields	Max. Error	Min. Error	Mean Error
EIGEN1S	58.33	13.26	43.1959
EIGEN1S.EVD.5171$^\diamond$	29.45	14.36	25.3325
EIGEN1S.EVD.6345$^\diamond$	16.50	8.88	10.5623
EIGEN1S.EVD.7581	5.64	3.10	3.6295
EIGEN1S.EVD.8373	2.41	1.28	1.6001
EIGEN1S.EVD.10000$^\diamond$	0.25	0.11	0.1412

Table 6.7: Geoid Height Errors of all EVD Gravity Fields.

for $\tilde{b}$ are calculated by considering the unbiased state estimates, EIGEN1S, TEG4 and EGM96, to be true state values, for separate evaluations of the KEV and Norm-Norm EVD biases. In each case the vector $\tilde{b}$ is approximated as

$$\begin{aligned} \tilde{b} &= \hat{X}_{\text{EIGEN1S.EVD.7581}} - \hat{X}_{UNBIASED} \\ \text{or} & \\ \tilde{b} &= \hat{X}_{\text{EIGEN1S.EVD.8373}} - \hat{X}_{UNBIASED}. \end{aligned} \tag{6.11}$$

The traces of the error variance-covariance matrix corresponding to EIGEN1S.EVD.-7581 and EIGEN1S.EVD.8373 have the scalar values

$$\begin{aligned} trace(P_{\hat{X}_{EIGEN1S.EVD.7581}}) &= 4.7889x10^{-07} \\ trace(P_{\hat{X}_{EIGEN1S.EVD.8373}}) &= 1.8195x10^{-14}, \end{aligned} \tag{6.12}$$

True Soln.	KEV $trace(\tilde{\tilde{b}}\tilde{\tilde{b}}^T)$	KEV ratio	Norm $trace(\tilde{\tilde{b}}\tilde{\tilde{b}}^T)$	Norm ratio
EIGEN1S	$2.0503x10^{-14}$	$4.2814x10^{-08}$	$1.8195x10^{-14}$	$8.9057x10^{-08}$
EGM96	$2.6835x10^{-14}$	$5.6037x10^{-08}$	$2.0901x10^{-14}$	$1.0230x10^{-07}$
TEG4	$2.6449x10^{-14}$	$5.5230x10^{-08}$	$2.0604x10^{-14}$	$1.0084x10^{-07}$

Table 6.8: CHAMP: KEV and Norm-Norm EVD Ratios of Bias to Random Error

and the trace of the outer product of $\tilde{b}$ is expressed in general as

$$trace(\tilde{\tilde{b}}\tilde{\tilde{b}}^T), \tag{6.13}$$

so that the ratio of these two traces is given by

$$\text{ratio} = \frac{trace(\tilde{\tilde{b}}\tilde{\tilde{b}}^T)}{trace(P_{\hat{X}_{EIGEN1S.EVD}})}. \tag{6.14}$$

Fortunately, since the ratios given in Table 6.8 are much less than unity, the biases in EIGEN1S.EVD.7581 and EIGEN1S.EVD.8373 can be ignored (Bouman [64]), because they are small compared to the random error in the EVD estimates. Therefore the KEV and Norm-Norm EVD solutions can be considered as unbiased in an approximate sense.

6.1.7.6 Discussion

PLASS performed the Eigenvalue Decomposition (EVD) on an ill conditioned system of equations to solve for 11216 parameters. It demonstrated a new feasibility in the application of the EVD in the solution for large gravity fields. Because the CHAMP normal matrix **ngl-eigen-1s** contains information derived from satellite only observations, it was expected that the attenuation of the short wave gravitational anomalies, those signals whose wavelengths are much shorter than the

satellite's altitude, would cause inflation in the estimation of their corresponding spherical harmonic coefficients. Four common optimization methods (Inspection, Relative Error, Mean Square Error (MSE), Norm-Norm minimization) and the introduced criterion of this investigation, which relates Kaula's rule to eigenvalue disposal (the KEV relation), were written into **PLASS** and implemented to find the best solution among the many that can be generated by an EVD stabilized estimate. All five methods of choosing the best regularization result, each produced their own optimum gravity field. In the Inspection case, the disposal of 10000 eigenvalues was deemed to be the optimum. In the Relative Error, MSE, KEV, and Norm-Norm cases, the removal of 6345, 5171, 7581, and 8373 eigenvalues, respectively, were found to be an optimum. To choose the best optimum among the five solutions, further evaluation of these gravity fields was carried out. Once all candidates were properly contrasted to the GFZ EIGEN1S gravity field, it was the solutions stabilized with the Kaula rule eigenvalue relation and the Norm-Norm minimization that survived scrutiny.

In all five solutions, there was bias introduced into the estimate due to eigenvalue truncation. Because bias is not quantified in the error variance-covariance matrix of an EVD estimate, this matrix is not an adequate measure of solution accuracy. Only by calculating the bias and comparing it to the random error, can one judge the quality of the solution. In the case of the KEV EVD and Norm-Norm stabilized gravity solutions, the comparison of the trace of the outer product of the bias vector $\tilde{b}$ to the trace of the error variance-covariance matrix, in the form of a ratio, showed that the bias was small compared to the random error.

By calculating the EVD covariance matrix for every candidate gravity field and reading in the (given) covariance matrix of the EIGEN1S, **PLASS** was able to propagate the uncertainties in all solutions by computing their geoid height errors. This was done to illustrate the incremental effects of eigenvalue disposal on an EVD covariance matrix. It was shown that the geoid height errors become smaller with eigenvalue removal, thus indicating an increase in confidence of random errors. Although an EVD solution is biased and its covariance matrix represents only a random effect, it was determined that the bias in the solution of EIGEN1S.EVD.7581 (KEV) and EIGEN1S.EVD.8373 (Norm-Norm) was sufficiently small enough to consider these estimates as non-biased in an approximate sense. In other words, their "shifted distance" away from the true solution, due to bias, may be considered negligible in that there were no deleterious effects detected by the analyzes of this investigation. Because the solutions stabilized by the Kaula Eigenvalue (KEV) relation and the Norm-Norm method survived close scrutiny, they are considered to be the best CHAMP EVD gravity fields of this study.

6.2 Gravity Recovery and Climate Experiment (GRACE) 160x160 Gravity Field

6.2.1 Introduction

The twin GRACE satellites were launched on March 17, 2002 into an almost circular orbit at an altitude 485 km, with an inclination of about 89 degrees. The primary science goal of the GRACE project is stated as: "High resolution, mean and time variable gravity field mapping for Earth System Science applications". Both

satellites are equipped with high-precision three-axes accelerometers for measuring surface force accelerations, while the distance between them is measured at the micron level using a microwave ranging system. Because of their low altitude, the satellites can be tracked continuously using the GPS constellation for constant and accurate monitoring of orbital perturbations, just as in the case of CHAMP.

The GRACE project is a joint effort between the National Aeronautics and Space Administration (NASA) and the Deutsche Forschungsanstalt für Luft und Raumfahrt (DLR) in Germany. Prof. Byron Tapley of The University of Texas Center for Space Research (UTCSR) is the Principal Investigator (PI), and Prof. Christoph Reigber of the GeoForschungsZentrum (GFZ) Potsdam is the Co-Principal Investigator (Co-PI). Project management and systems engineering activities are carried out by the Jet Propulsion Laboratory (JPL), while engineering orbit operations are conducted by the German Space Operations Center (GSOC) in Oberpfaffenhofen, Germany.

One of the GRACE scientific objectives is to determine very precisely the global long-wavelength features of the static Earth gravity field and its temporal variation. The linearized system of equations which relate the model of the GRACE satellites' motion to the GPS observations, $y = Hx + \epsilon$, are solved by transforming (using Householder reflections discussed in the previous chapter) them to an orthogonalized system of equations, Equation 4.21. Matrix H is the so called information matrix of dimension m x n (m observations and n parameters) that already includes observation weights. After orthogonalization, upper triangular matrix R represents the transformed system of equations. **PLASS** is then applied to perform

the Singular Value Decomposition (SVD), yielding the singular values of R that are used to stabilize the system of equations and obtain a gravity solution. The R matrix of GRACE33(Case 61) is used in this investigation. The number of estimated parameters n, is 25917 and the number of observations m, is 8309302. Since the nxn matrix R corresponds to satellite only observations, and because of rapid attenuation of spherical harmonics with increasing altitude, a high degree of unobservability causes this matrix to be ill-conditioned. Using R, **PLASS** produced five stabilized solutions, along with three covariance matrices and their subsequent geoid height error propagations. It was deemed unnecessary to compute all five SVD covariance matrices. The five optimal solutions are presented in order of increasing quality.

6.2.2 Inspection

Before an inspection analysis can be performed, a graph of the singular values versus number (where "number" is the ith singular value), must be obtained. Figure 6.32 is the graph of the singular values of matrix R. The largest and smallest singular values are $1.037 \cdot 10^7$ and $8.293 \cdot 10^1$, respectively, which yields a condition number of $1.2506 \cdot 10^5$; indicating an ill-conditioned system of equations. Often when displaying singular values in this manner, they are sorted by either increasing or decreasing size. In Figure 6.32, the GRACE singular values are arranged by decreasing size. For solution, their original sequence is maintained.

Because the smallest singular value is much greater than zero and there is such a smooth transition throughout most of the graph, it is difficult to know which

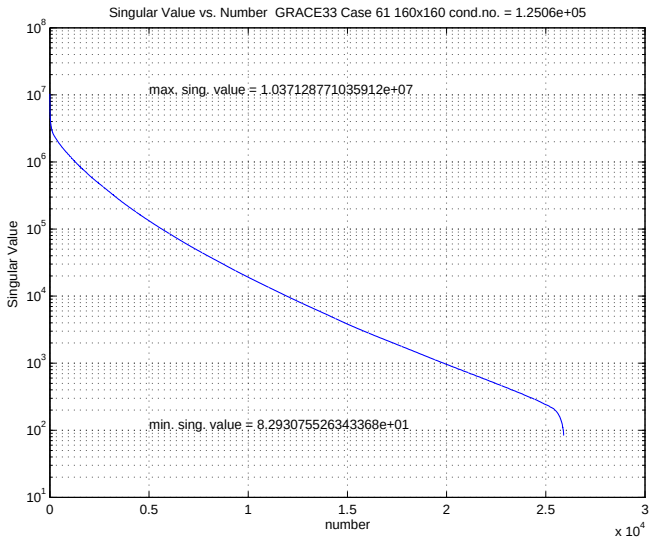


Figure 6.32: GRACE33(Case61) Singular Value vs Number.

of the singular values are responsible for the ill-conditioned nature of the R matrix. It was hoped that singular values very close to zero would be found, because such quantities are usually responsible for causing inflation in a solution. Therefore five guesses were made. The first guess was to dispose the smallest 1000 singular values. The next four guesses were to remove the 5000, 10000, 15000, and 20000 smallest singular values, producing a separate solution for each guess. For comparison, a solution in which all singular values were retained, was carried out. The degree variance spectrum of the solution in which no singular values were removed, is shown in Figure 6.33 overlayed with Kaula's rule. All five solution spectra of the Inspection cases are presented together in Figure 6.34 as graphs of degree amplitude versus harmonic degree overlayed with Kaula's rule. Clearly, it can be seen from Figure 6.33, that without stabilization from singular value disposal, the estimates of parameters above a harmonic degree of about 100 are quite inflated. But when the

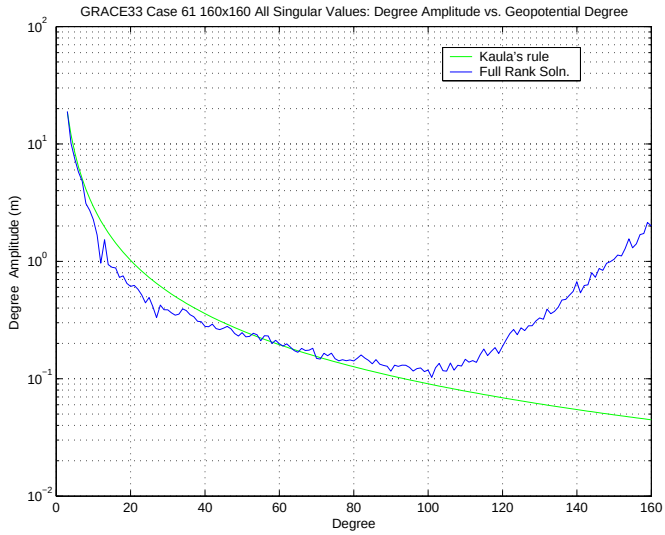


Figure 6.33: GRACE33(Case61): All Singular Values Solution

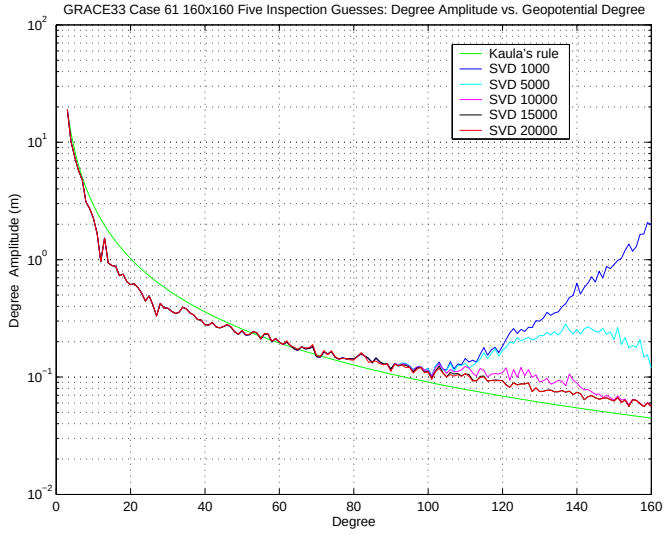


Figure 6.34: GRACE33(Case61): Five Inspection Solutions

singular vectors which contribute to this inflation (through their linear combination in the null space) are eliminated by singular value disposal, their deleterious effects are set to zero, and the solution is improved. Because the inspection method is subjective, it is not a reliable technique of determining how many singular values to throw away for an optimal SVD solution. However, the guess to remove the 20000 smallest singular values is the best of all five guesses, and will be considered as a candidate gravity field for further evaluation. The next section describes the results of the Relative Error method.

6.2.3 Relative Error

Because the relative error calculation in Equation 4.36 is an explicit formula, it is a simple matter to evaluate the relative error for all smallest singular value disposal permutations. Since there are 25917 singular values, the same number of relative error scenarios are computed. This leads to the graph of relative error as a percentage versus the number of singular values retained, see Figure 6.35. As expected, when all singular values are thrown away the relative error becomes 100 percent, and zero when all are kept. One may contend that the best relative error is zero percent, however we are looking for a non-trivial solution in between the extremes. By graphing the percentage data, that point was determined to be at 25916 retained singular values, equivalent to 1 discarded. Figure 6.36 displays the corresponding degree amplitude versus harmonic degree spectrum, overlayed with Kaula's rule. Unfortunately, this result is not desirable. The difference between an unstabilized solution and one in which only one singular value is removed, is

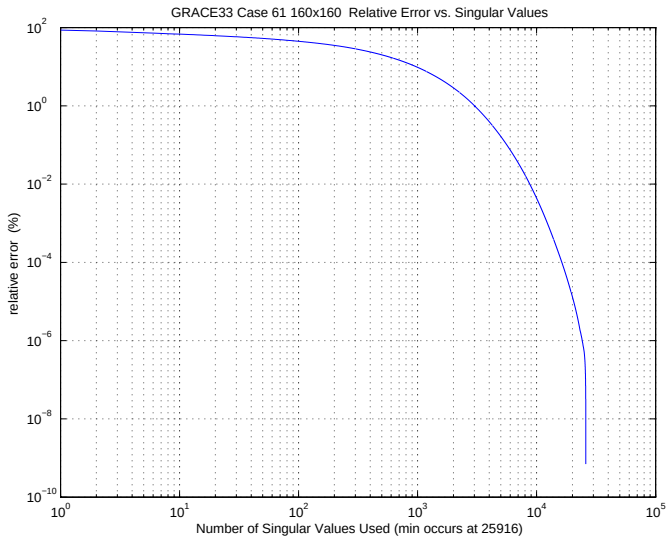


Figure 6.35: Relative Error Graph for GRACE33(Case61).

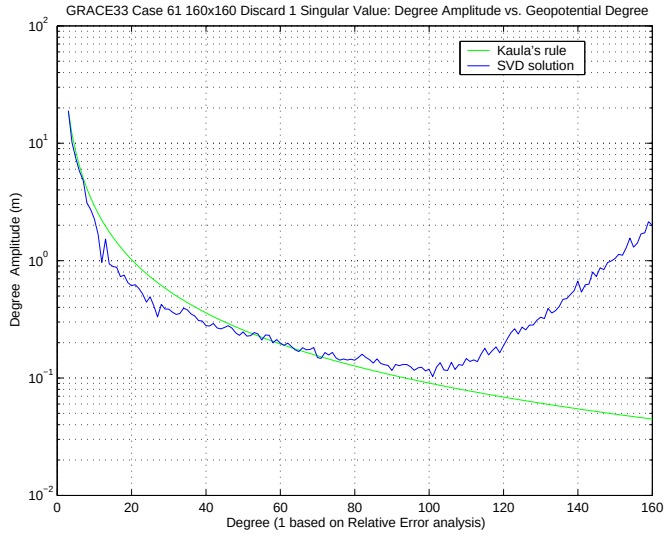


Figure 6.36: GRACE33(Case61): Discard 1 Singular Value.

negligible. Thus, this solution is eliminated from contention as a candidate gravity field.

6.2.4 Mean Square Error (MSE)

The computation of an MSE solution in the case for GRACE33(Case61), requires the accumulation of the null space matrix V and the calculation of the z^2 vector in Equation 4.43. Fortunately, the scaling of the null space matrix by selected Kaula quantities, as coordinated by the parameter-name-list vector, is carried out only once before a complete permutation of retained and disposed singular value combinations is initiated. The result is three curves: total noise, total bias as a result of singular value disposal (approximated by Kaula's rule) and mean squared error (MSE). The MSE curve is the addition of the other two curves that are functions of unitless scalar values. The minimum of the MSE function is the optimum combination that minimizes the scalar contribution from both the total noise and total bias. That point was found to occur exactly at the value of 3 retained or 25914 discarded singular values, see Figure 6.37. Unfortunately, the results of the MSE analysis proved to be of little help. By looking at the solutions seen in the five cases of the inspection analysis, it is obvious that removal of only 3 singular values is not sufficient to remove the excess power in the unregularized solution spectrum above harmonic degree 100. For this reason, the MSE solution is eliminated from contention as a candidate gravity field.

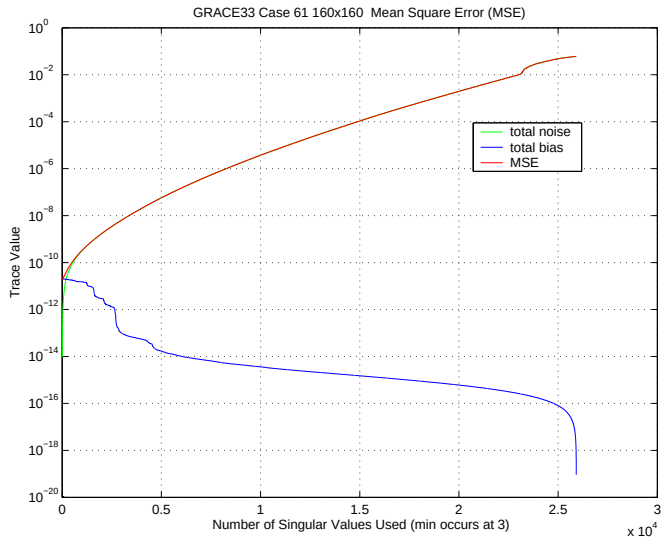


Figure 6.37: GRACE33(Case61): Mean Square Error

6.2.5 Norm-Norm Minimization

The computation of the norm of the residual and the norm of the solution can be carried out with or without the null space matrix V. If the null space matrix is not desired and only solutions are necessary, then **PLASS**1 should be used. **PLASS**2 should be employed, if other stabilization schemes are to be tested and error analyzes carried out. In the norm-norm plot analysis of GRACE33(Case61), **PLASS**2 is used to produce 1296 different GRACE33(Case61) SVD gravity solutions, calculating them at a granularity of 20 singular values (i.e. removing 20 at a time, for each computed solution). Once the global minimum is found at this resolution, a subsequent search at the precision of one singular value is performed. This enables an efficient search on all of the 25917 possible permutations in the calculation of solution corrections $\|\hat{x}\|_2 = \|\widetilde{A}^{-1}\overline{b}\|_2$, as seen in Equation 4.37, and

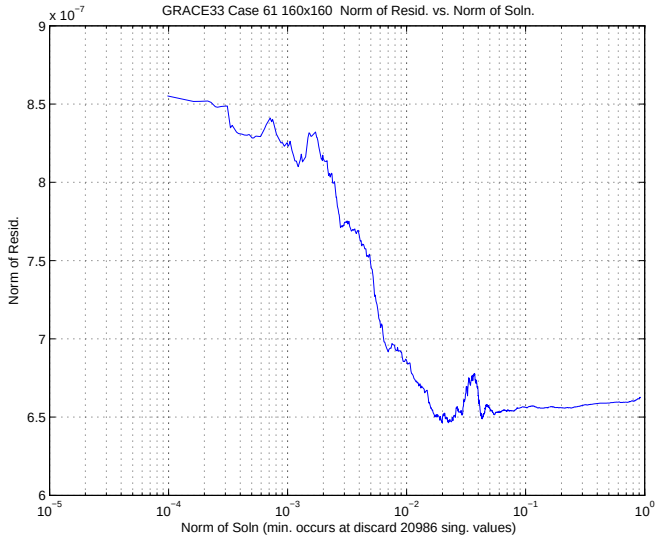


Figure 6.38: GRACE33(Case61): Norm of Resid. vs Norm of Soln.

residuals $\|\overline{\rho}\|_2 = \| \widetilde{A}\hat{x} - \overline{b} \|_2$ as seen in Equation 4.38, to find the solution corresponding to the optimum number of singular values retained. For example, as each singular value is disposed, a separate solution correction vector and residual is calculated for that scenario. The norm of each of the correction vectors and residuals is taken and then plotted. Traditionally, using one processor, performing a search of singular values for large problem norm-norm plot analysis, is prohibitive due to the enormous amount of memory and the computer time a sequential algorithm requires. However, given enough processors, **PLASS** eliminates these obstacles and completes the analysis in a matter of a few hours. Figure 6.38 is a log-log plot of the norm of the residual versus the norm of the correction vector for the scenarios computed. The point on the norm-norm curve that is closest to the origin is where the optimal balance between these two norms is found. It is the point where the square root of the sum of the squares of the elements of the residual vector $\overline{\rho}$ and solution

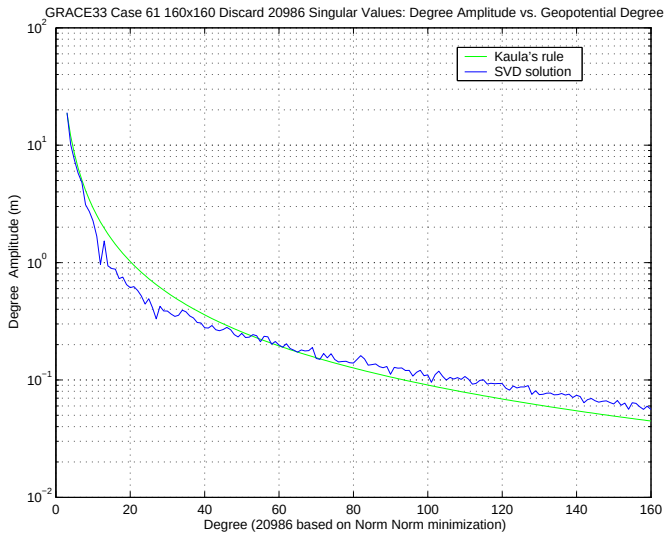


Figure 6.39: GRACE33(Case61): Discard 20986 Singular Values.

vector $\hat{x}$ are a minimum. This point occurs in the case where 20986 singular values are thrown away. Since the optimum suggested by the norm-norm graph corresponds to a gravity field *correction*, this correction must be added to the nominal (*a priori*) gravity field for evaluation. Figure 6.39 shows the corresponding updated gravity solution expressed as degree amplitude versus harmonic degree overlayed with Kaula's rule. It can be seen that most of the power in the estimated coefficients which causes their inflation above Kaula's rule has been removed. Since there are 25917 singular values, we have discarded 81 percent of the information contained in the GRACE33(Case61) upper triangular matrix R to invert for solution. Perhaps too many singular values have been removed from R to obtain solution, however it will be submitted as a candidate for further gravity field analysis. The next stabilization technique involves Kaula's rule as a reference (not a constraint) to singular value removal for solution, leading perhaps to more information retention.

6.2.6 Kaula Singular Value (KSV) Relation

Using **PLASS**2 and the null space matrix generated from the previous norm-norm analysis, every case of singular value disposal was evaluated with Equation 4.47, repeated here for convenience,

$$\alpha(k) = \left\{ \sum_{l=1}^{lmax} \left[v(l)_{svd(k)_{l,m}} - v(l)_{Kaula} \right]^2 \right\}^{\frac{1}{2}}, \tag{6.15}$$

where the subscript "svd" indicates Singular Value Decomposition (as called for in the case of the GRACE orthogonal matrix R). The index k is the number of retained singular values. The dimensionless term $\alpha(k)$ is plotted versus number of singular values used. There are a total of 1296 alpha values in the generated function at a granularity of 20 singular values. After the coarse global minimum of this curve was found, alphas were calculated again, but at a granularity of 1 singular value in a region spanning both sides of the global minimum. By "focusing" in on the global minimum in this manner, it is possible to find the precise number of used singular values that best satisfies Kaula's rule. Figure 6.40 shows the rough search for the global minimum and Figure 6.41 displays the refined search in the area where the global minimum occurs, which is at the point where 21372 singular values were discarded (4545 are kept). Figure 6.42 illustrates the corresponding updated gravity solution expressed as degree amplitude versus harmonic degree overlayed with Kaula's rule. Comparing this case to that of the previous section, more singular values (82.5 percent total) were discarded. Thus 17.5 percent of the original information inside matrix R was retained for the SVD stabilized inversion. In the case of the norm-norm solution, it was 19 percent. Clearly, Figure 6.42

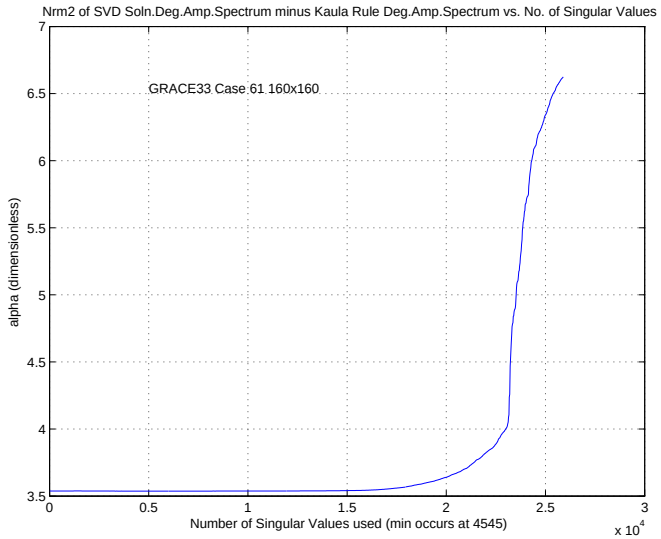


Figure 6.40: GRACE33(Case61): Global Rough Sweep of Singular Value Disposal Cases

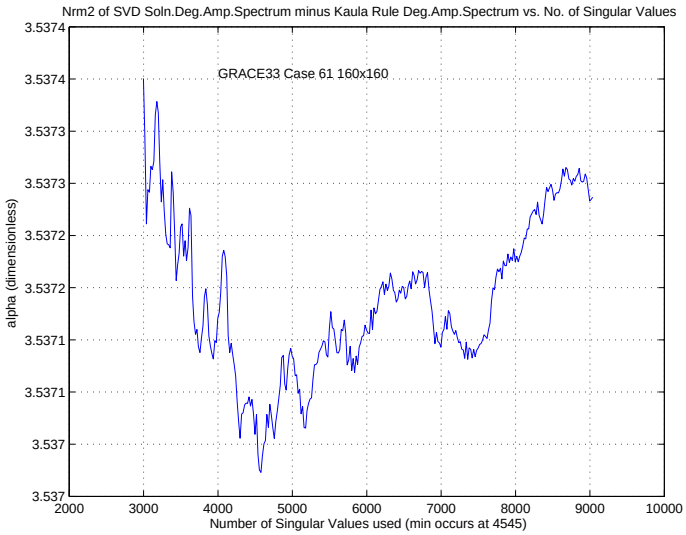


Figure 6.41: GRACE33(Case61):Refined Sweep near Global Min. of Used Singular Value Cases

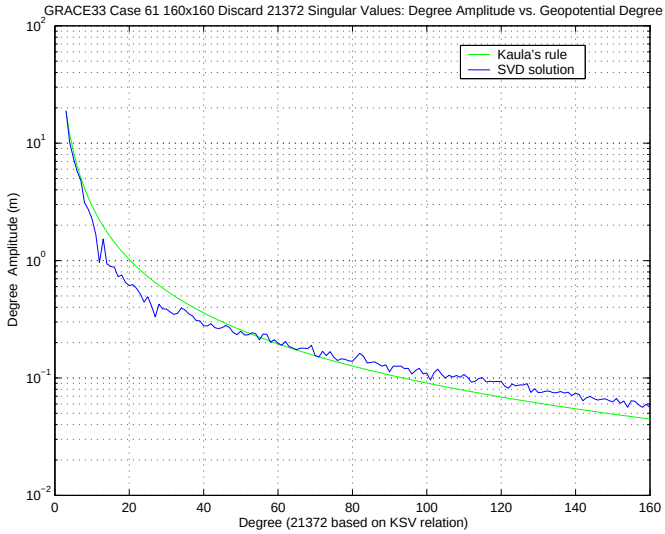


Figure 6.42: GRACE33(Case61): Discard 21372 Singular Value Solution

fulfills the expectation that this solution follows Kaula's rule well. It is only with further evaluation however, can the best of the three fields (inspection, norm-norm, KSV) be determined.

6.2.7 Evaluation of Gravity Fields

To determine the best of the three candidate gravity fields, evaluations based on error propagation from the computation of geoid height errors, orbit fit calculations, geoid comparisons, and degree error variance analyzes are performed. It must be remembered however, that the covariance matrix corresponding to a Singular Value Decomposition(SVD) estimated solution is not an adequate measure of error for an estimate. The error represented by an SVD covariance matrix is only that of the random effect and does not include the error of bias, introduced by the removal of singular values. Because singular values (in the case of GRACE33(

Method	Optimum Number of Used Singular Values
Inspection	5917
$RelativeError^{\diamond}$	25916
$MSE^{\diamond}$	3
Norm Norm minimization	4931
KSV relation	4545
$\diamond$ = eliminated	

Table 6.9: Three Candidate Gravity Fields to be Evaluated

Case61)) have been removed (set to zero) in these solutions, their corresponding singular vectors were not included into the estimation process. This is the cause of the bias and it produces a "shift" in the estimated gravity field by some amount away from the true gravity field. The confidence in the estimated coefficients may also be too optimistic, i.e. perhaps their variances are tighter than the variances of the full rank unbiased estimate. However, if the bias introduced by an SVD estimate is "small", the estimate may be considered unbiased, at least in an approximate sense. This will be discussed after the best SVD gravity field has been chosen. Table 6.9 summarizes the used singular values in all estimated GRACE gravity fields and for convenience, Figure 6.43 presents all of the solutions' power spectra in a common graph overlayed with Kaula's rule. (Note: Unfortunately, all solutions are not visible in this figure, as some are unavoidably hidden by others when plotted.) It can be seen that as singular values are discarded, power in the coefficient corrections causing inflation is removed and the solution becomes more stabilized. The next section will test these gravity fields by using them to model the motion of selected Earth satellites, comparing their predicted states with actual observations and reporting the results in the form of orbital fit residuals.

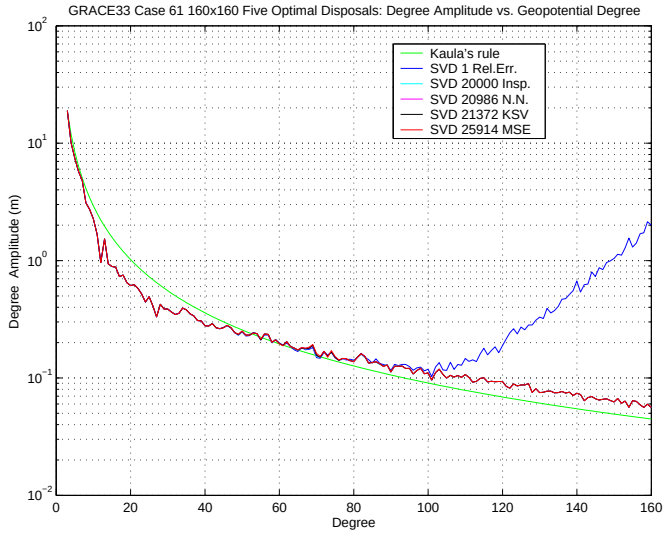


Figure 6.43: GRACE33(Case61): All Five Candidate SVD Solutions.

6.2.7.1 Orbital Arc Fit Computations

Since the orbital arc fit analysis employed in this section evaluates only those coefficients of an estimated gravity field below degree and order of 100, the gravity coefficients of the GRACE33(Case61) field above this value have little or no effect on orbit fit residuals. Therefore, this section offers only a partial, but valuable, insight to the quality of the three candidate solutions. The satellites selected to fly through the estimated SVD 160x160 gravity fields are the same as those used in the previous CHAMP analysis. For convenience they are shown again in Table 6.10. This group of satellites were chosen because they represent a good sample of inclinations and altitudes necessary for an adequate orbital fit test. Due to its low altitude, satellite GFZ1 can test the higher degrees and orders, while Starlette, Stella, and Topex are satellites that are well suited to assess the *somewhat* higher degrees and orders of an estimated gravity field. To scrutinize the quality of the low degrees

Satellites	Semi-major axis (km)	Inclination (deg.)	Eccentricity
Topex/Poseidon	7705	65.99	0.0010
Starlette	7335	49.81	0.0206
Stella	7200	98.00	0.0001
Lageos 1	12271	109.83	0.0040
Lageos 2	12162	52.63	0.0137
GFZ1	6764	51.64	0.0073

Table 6.10: Orbital Elements of Selected Satellites

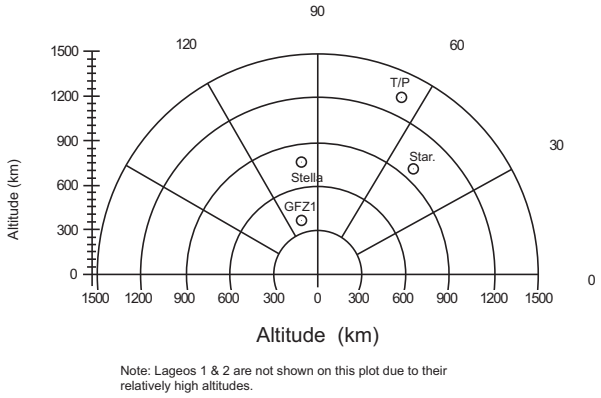


Figure 6.44: Selected Satellites for Orbit Fit Analysis.

and orders, the satellites Lageos 1 and 2 were chosen for their high altitudes. Figure 6.44, repeated in this section for convenience, illustrates schematically their orbital heights above the Earth. Each orbit fit of the three SVD gravity fields were computed using the UTOPIA (Tapley et al. [68]) program and compared with the actual observation data for a chosen satellite. For uniformity in the orbit fit calculations, all of the gravitational and non-gravitational perturbations of the orbit estimations in the UTOPIA runs are kept consistent for each case and each test satellite. The gravitational perturbations are those due to the non-spherical geopotential of the Earth, sun, moon, other planets, the solid Earth tides, ocean tides, rotational defor-

Satellites	Arc length (days)	Number of Arcs	Epoch
Topex/Poseidon	10	1	June 1999
Starlette	5	6	Jan. 1998
Stella	5	6	Sept.1993
Lageos 1	3	10	May 2002
Lageos 2	3	10	May 2002
GFZ1	3	7	Aug. 1995

Table 6.11: Observation Data used in UTOPIA Runs

mation, and general relativity. The non-gravitational perturbations are those due to atmospheric drag, solar radiation pressure, Earth radiation pressure and other unknown forces. Since the epoch of the GRACE33(Case61) SVD fields are years later than the epochs of the satellites, J2, J3 and J4 were propagated to a common epoch. Table 6.11 contains the arc lengths, number of arcs, and satellite epochs used in the UTOPIA runs. Table 6.12 shows the SLR (Satellite Laser Range) orbit fits in centimeters RMS (radial) of the three candidate SVD gravity fields on all selected satellites. The case GRACE33(Case61) is the gravity field produced by the full rank solution. It is this gravity field to which the three candidate SVD fields are compared.

Other than the GFZ1 and Stella satellites, the RMS values are fairly similar for all cases of the SVD gravity fields and GRACE33(Case61). It is not until using the two lowest altitude test satellites, GFZ1 and Stella, does the effect of discarding singular values deliver a noticeable influence to RMS orbit fits. This is not surprising, as lower altitude satellites are more sensitive to the shorter wavelength gravity signals corresponding to degrees and orders higher than about 100. The RMS fits of the high altitude satellites Lageos1, Lageos2, Starlette and Topex should not be

Case	GFZ1	Lageos 1	Lageos 2	Starlette	Stella	Topex
Inspection	11.39	1.09	1.08	3.0	5.05	2.16
KSV relation	12.52	1.05	1.03	3.01	7.47	3.08
Norm Norm minimization	12.25	1.07	1.06	2.97	6.52	2.05
GRACE33(Case61)	9.32	1.40	1.50	2.95	3.33	2.05

Table 6.12: Orbital Arc Fits of Candidate Gravity Fields (cm. radial RMS)

influenced by these shorter wavelength anomalies nor should it be expected that their orbit fit residuals could be influenced significantly by SVD deflation effects for the estimates of this study. The SVD regularization serves mostly to remove the inflation in the poorly observed high degree and order coefficients that signify short wavelength anomalies. Of course, since stabilization affects all estimated parameters, there may exist a tradeoff between the deflation in these higher order and degree coefficients with the estimation of the lower degree and order coefficients. This negotiation, between deflation of poorly observed geopotential coefficients and the quality in the estimation of well observed parameters, manifests itself clearly in the RMS fits of satellite GFZ1. The RMS increases by about 2 to 3 centimeters using any of the SVD fields, in comparison to the full rank (unstabilized) residual. Fortunately, the tradeoff of 3.2 centimeters RMS for a well stabilized (KSV) solution is a very reasonable one. Because there appears to be a negligible distinction in the RMS fits of the three SVD fields, all of them will be submitted for further analysis.

6.2.7.2 SVD Degree Error Variance and Geopotential Variance Difference vs. GRACE33 (Case61)

By comparing the formal degree error variances of each candidate SVD gravity field with that of GRACE33(Case61), further objective elimination of solutions is possible. The formal degree error variance is a measure of the power of the geopotential errors for a specific harmonic degree and is represented as the sum of the squares of the formal variance of the estimated gravity coefficients for a particular harmonic degree l. It is expressed in Equation 6.2, but repeated here for convenience, as

$$\sigma_l^2 = \sum_{m=1}^{l_{max}} \left(\sigma^2_{\overline{C}_{lm}} + \sigma^2_{\overline{S}_{lm}} \right), \tag{6.16}$$

where $\sigma_{\overline{C}_{lm}}$ and $\sigma_{\overline{S}_{lm}}$ are the standard deviation of the normalized estimated geopotential coefficients for degree l and order m. The value σ_l^2 is then scaled into millimeters of height and then plotted as a function of degree amplitude versus harmonic degree. To generate degree amplitude difference curves for the power spectrum differences between the estimated coefficients of the SVD and the full rank GRACE33(Case61) solution, the degree variance difference $\triangle_l^2$ is calculated. This relation is a measure of the power of the geopotential differences between two gravity fields at a specific harmonic degree l, and is expressed in Equation 6.3 (repeated here for convenience) as,

$$\triangle_l^2 = \sum_{m=1}^{lmax} \left(\triangle^2_{\overline{C}_{lm}} + \triangle^2_{\overline{S}_{lm}} \right), \tag{6.17}$$

where $\triangle_{\overline{C}_{lm}}$ and $\triangle_{\overline{S}_{lm}}$ are the scalar differences of the normalized geopotential coefficients for a specific harmonic degree l and order m.

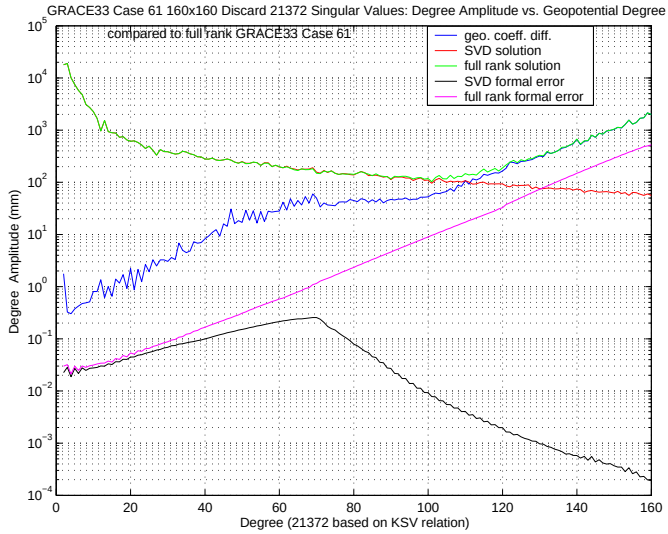


Figure 6.45: KSV Degree Error Var. and Var. Geopotential Difference to Case61

Since every candidate SVD gravity field is biased (because singular values are discarded), the SVD error covariance matrix contains only random error, as mentioned in Section 4.1.1. However, the error variances of a candidate SVD gravity field and of a reference field (which is unbiased) may be compared to evaluate the difference between the two fields with respect to the error variance of the reference field. Figure 6.45 shows the degree error variance of the KSV solution in comparison with that of the full rank GRACE33(Case61) reference field and the differences between their geopotential coefficient variance spectra along with the candidate solution in the form of its geopotential power per harmonic degree. It can be seen that the error variance of the biased KSV SVD solution are all within (smaller than) the error variance of GRACE33(Case61). This implies that the candidate KSV SVD field is within the uncertainty (in a random sense) of the GRACE33(Case61) solution. Figures 6.46 and 6.47 illustrate the same information for the Inspection

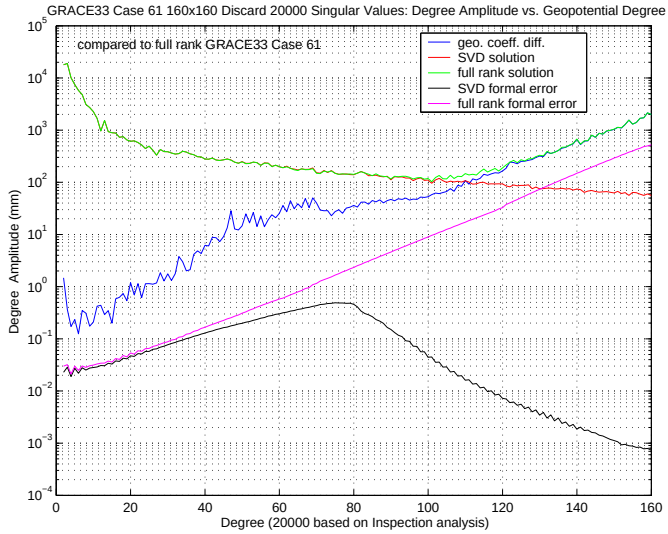


Figure 6.46: Insp. Degree Error Var. and Var. Geopotential Difference to Case61

and Norm-Norm candidate gravity fields respectively. These plots show that the degree error variances of the three SVD solutions are *all* (in a random sense) within the degree error variance of GRACE33(Case61). The spectra of the degree error variances for these solutions, however may indicate an overconfidence (in a random sense) in the estimation for those coefficients above a harmonic degree of about 80, as all (formal error) curves decrease rapidly and monotonically beyond this region. At this point three of five SVD gravity solutions have survived scrutiny. In the next section, we compare their geoids to that of GRACE33(Case61).

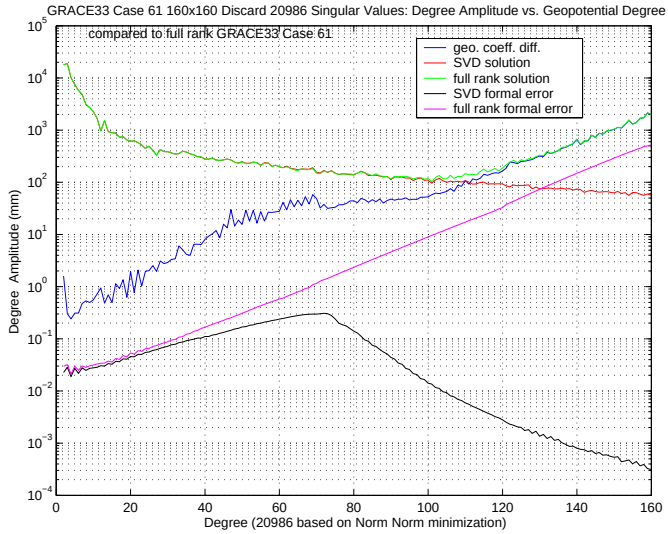


Figure 6.47: Norm Degree Error Var. and Var. Geopotential Difference to Case61

6.2.7.3 SVD Geoid Differences to Full Rank Case61

The simple geoid undulation above the Earth's mean spheroid of radius a_e, is computed (repeated here for convenience) as (Condi [72])

$$N = a_e \sum_{l=2}^{L_{max}} \sum_{m=0}^{l} \overline{P}_{l,m}(\sin\phi)[\Delta\overline{C}_{l,m}\cos m\lambda + \Delta\overline{S}_{l,m}\sin m\lambda] \qquad (6.18)$$

where $\Delta\overline{C}_{l,m}$ and $\Delta\overline{S}_{l,m}$ are the differences of the normalized geopotential coefficients between two gravity fields. Plotting N in a contour map format reveals the geoid undulation differences between two gravity fields.

Figure 6.48 shows a "ripple filled" reference geoid undulation map for the full rank solution GRACE33(Case61) field. These ripple patterns indicate an observability problem in the geopotential coefficients used to calculate this contour plot. The distinctive areas of low regions south of India, Antarctica and off the

West and East coasts of North America are observed, as well as the high regions of the North Atlantic and East South Pacific, which contain an emphasis of the ruffling patterns. These oscillations are smoothed in the geoids of the best two candidate gravity fields given the labels GRACE33.SVD.20986 and GRACE33.SVD.21372, seen in Figures 6.49 and 6.50, respectively.

The effect of removing 20000 singular values on geoid undulations is illustrated in the differences between the geoids of GRACE33(Case61) and GRACE33.-SVD.20000, seen in Figure 6.51. Over the majority of the Earth, both land and ocean, the differences undulate roughly between -7.7 cm and 7 cm. Only in the region midway between Australia and South America, can wider differences be seen. It is not clear why this anomaly appears. Discarding another 986 singular values to minimize the norms of the residual and correction to the nominal, (for a cumulative total of 20986 removed singular values) yields a pattern very similar to that of the inspection case, with very similar extrema as seen in Figure 6.52. The removal of yet another 386 singular values to best satisfy the KSV relation, has little effect on this pattern, see Figure 6.53.

Table 6.13 summarizes the minima, maxima and mean geoid undulation values of the reference field GRACE33(Case61) and those of the candidate SVD gravity fields. It can be seen that the removal of over 20000 singular values results in only a three or four centimeter change in these geoid undulation extrema. Table 6.14 summarizes the extrema and mean values of the differences between the GRACE33(Case61) and the SVD gravity fields. It can be seen that the spread between the maximum and minimum geoid points for each case in Table 6.14 is small,

Number of geoid points computed: 64,800
Parameters of the mean Earth ellipsoid used in geoid undulation computations:
GM $= 3.986004415$ x $10^{14} \frac{m^3}{s^2}$, $a_e = 6.37813630$ x 10^6 m
f $= \frac{1}{298.2570}$, $\omega_e = 7.29215$ x $10^{-5} \frac{rad}{sec}$
all table entries: units in cm.

Gravity Fields	Max. Height	Min. Height	Mean Value of Geoid Undulation
GRACE33(Case61)	87.78	-111.00	-0.7657
GRACE33.SVD.20000	84.13	-106.28	-0.8187
GRACE33.SVD.20986	84.36	-106.33	-0.8205
GRACE33.SVD.21372	84.46	-106.25	-0.8207

Table 6.13: Geoid Undulations of GRACE33 SVD Gravity Fields

changing only slightly as more singular values are removed from the R matrix, for each case. Since only hundreds of singular values are discarded from case to case, compared to the initial 20000 or so needed to initially stabilize for a solution, this result is not surprising.

The SVD solutions continue to improve with singular value removal, because the GRACE33(Case61) R matrix poorly observes coefficients above degree and order 100. But if too many singular values are removed the estimated coefficients may be nonsensical, even though inflation has been removed and their degree variance power spectrum falls below Kaula's rule. This potential hazard was not detected in the orbit fit analyzes as was the case in one instance of a CHAMP solution. To propagate the uncertainties in these three GRACE SVD gravity fields, including the full rank GRACE33(Case61) reference field, **PLASS**2 was employed to compute their geoid height errors.

Number of geoid points computed: 64,800
Parameters of the mean Earth ellipsoid used in geoid undulation computations:
GM $= 3.986004415$ x $10^{14} \frac{m^3}{s^2}$, $a_e = 6.37813630$ x 10^6 m
f $= \frac{1}{298.2570}$, $\omega_e = 7.29215$ x $10^{-5} \frac{rad}{sec}$
all table entries: units in cm.

Gravity Fields	Max. Height	Min. Height	RMS about Mean
GRACE33 - GRACE33.SVD.20000	36.42	-37.17	6.5554
GRACE33 - GRACE33.SVD.20986	36.41	-37.04	6.5599
GRACE33 - GRACE33.SVD.21372	36.43	-37.06	6.5609

Table 6.14: Geoid Undulation Diff. of GRACE33 SVD Grav. Fields to GRACE33

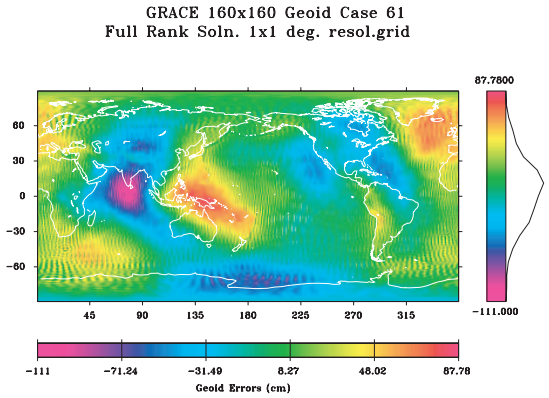


Figure 6.48: GRACE: Full Rank GRACE33(Case61) Geoid

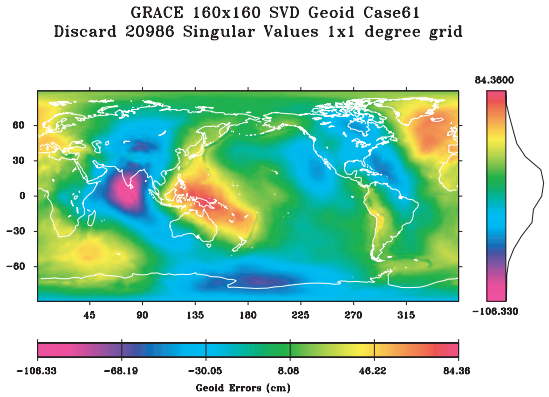


Figure 6.49: GRACE: GRACE33.SVD.20986 Geoid

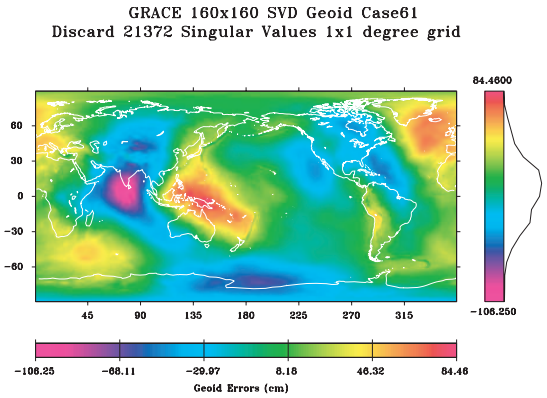


Figure 6.50: GRACE: GRACE33.SVD.21372 Geoid

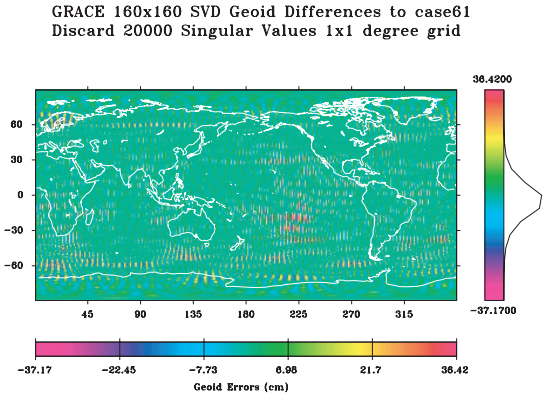


Figure 6.51: GRACE: Geoids: GRACE33(Case61) minus GRACE33.SVD.20000

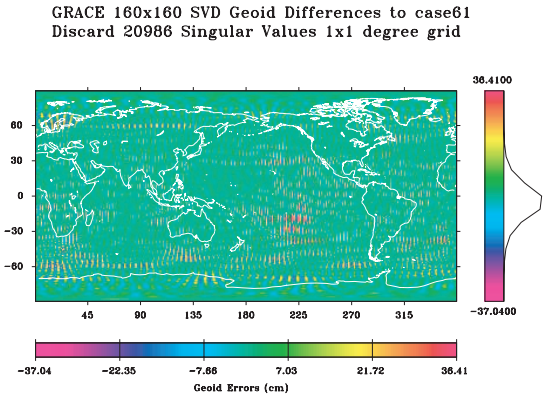


Figure 6.52: GRACE: Geoids: GRACE33(Case61) minus GRACE33.SVD.20986

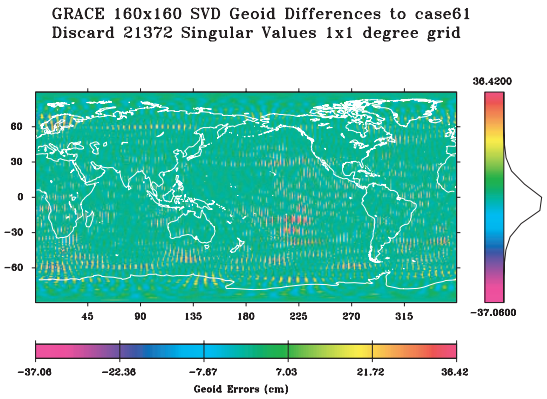


Figure 6.53: GRACE: Geoids: GRACE33(Case61) minus GRACE33.SVD.21372

6.2.7.4 Geoid Height Errors of the SVD and Full Rank GRACE33 (Case61) Gravity Fields

To assess the error of a particular gravity solution, it is necessary to compute the covariance matrix, also known as the variance-covariance matrix. It is the diagonal elements of this matrix that convey the statistical confidence of each parameter in an estimated solution. The off diagonals represent their covariances and can be used to derive all correlations among them. The covariance matrix can be used to support "goodness of fit" evaluations or error propagations. In this investigation the covariance matrix is used in an error propagation known as the geoid height errors. Such error calculations are the mapping of all estimation uncertainties into a space called the geodetic surface of the Earth. The traditional calculation of geoid height errors using one processor and the realized advantage of the parallel method employed by **PLASS**, is discussed in Section 6.1.7.4.

Using the R matrix of GRACE33(Case61), **PLASS** calculated, "**in-place**", the covariance matrix of the full rank GRACE33(Case61) gravity field, namely

$R^{-1}R^{-T}$ (Hinga [57]). The term "in-place" refers to the fact that no *work* space is required to accomplish this operation. Full rank covariance calculations are carried out using only the memory required to contain the matrix R.

After computing the covariance matrix of the full rank GRACE33(Case61) gravity field and that for each candidate SVD gravity field, **PLASS** was used to calculate all geoid height errors at a global grid resolution of one degree of latitude and longitude. Table 6.15 summarizes the propagated error extrema seen in the geoid height error plots and the parameters used to calculate them. (Note: The calculation of the geoid height errors is carried out at a 1^ox1^o grid resolution although the 160 degree and order GRACE gravity field contains only enough gravity information to support a (ϕ, λ) grid resolution of 3 degrees. Therefore, it should be remembered that the grid resolution is actually 3^ox3^o in a physical sense.)

Figure 6.54 shows the full rank geoid height error. Figure 6.55 shows the results of the Inspection case where 20000 singular values are removed to stabilize the solution so that the degree coefficient variance power spectrum falls below Kaula's rule. Clearly, the disposal of 20000 singular values eliminates a considerable amount of error and emphasizes the vertical pattern seen in the full rank geoid height error plot. This pattern may indicate the inclination of the GRACE twin satellites. Figures 6.56 and 6.57 illustrate that the disposal of another 986, then an additional 386 singular values, respectively, further reduces geoid height error, but does not affect the overall pattern of propagated error. The vertical patterns remain in these last two plots.

The tightening of the spread between maximum and minimum errors, as

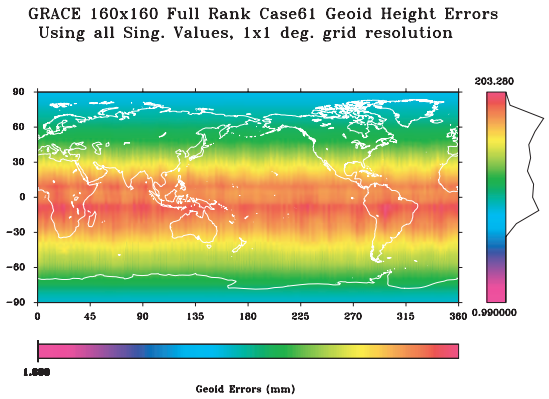


Figure 6.54: GRACE: Geoid Height Errors Full Rank GRACE33(Case61)

seen in the CHAMP section, is again occurring as a result of singular value removal in the case of GRACE, because the variance-covariance matrix is reduced in magnitude by singular value disposal. Although, the errors appear to be declining in these figures, one should bear in mind that the bias introduced from the exclusion of singular values is not being accounted for. These plotted geoid height errors stem from random error only and may be overly optimistic and represent solutions that are overly biased. Although many singular values were truncated in the cases of GRACE33.SVD.20000, GRACE33.SVD.20986, and GRACE33.SVD.21372, they may still be considered as un-biased estimates in an approximate sense. This issue is discussed in section 6.2.7.5.

6.2.7.5 The KSV and Norm-Norm SVD Bias

Two different bias vectors for $\tilde{b}$ are calculated by considering the unbiased state estimates TEG4 and EGM96, to be true state values, for separate evaluations of the KSV and Norm-Norm SVD biases. In each case the vector $\tilde{b}$ is approximated

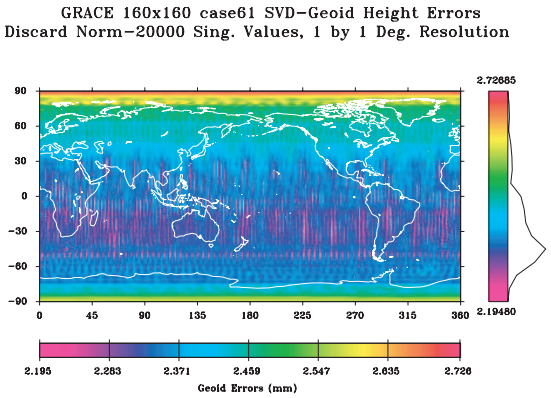


Figure 6.55: GRACE: Geoid Height Errors GRACE33.SVD.20000

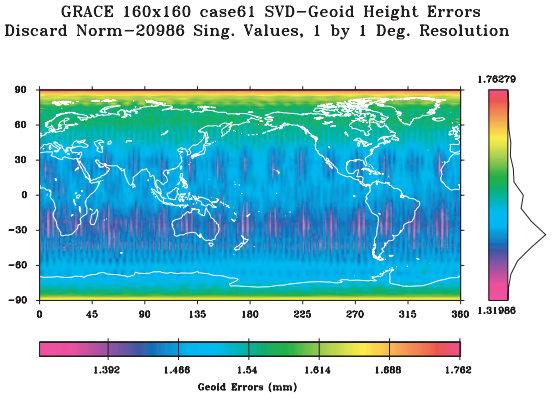


Figure 6.56: GRACE: Geoid Height Errors GRACE33.SVD.20986

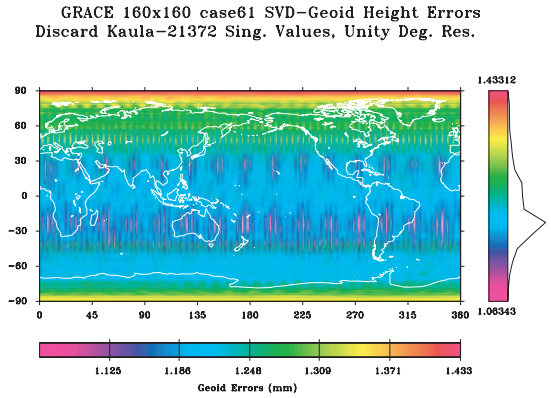


Figure 6.57: GRACE: Geoid Height Errors GRACE33.SVD.21372 Solution

Number of geoid height error points computed: 64,440 Parameters used to map uncertainties to geodetic surface: GM $= 3.986004415$ x $10^{14} \frac{m^3}{s^2}$, $R_e = 6.37813630$ x 10^6 m $\diamond$ = eliminated, all table entries: units in cm.			
Gravity Fields	Max. Error	Min. Error	Mean Error
GRACE33(Case61)	2032.8	861.51	1394.8
GRACE33.SVD.20000	2.73	2.19	2.3936
GRACE33.SVD.20986	1.76	1.32	1.5014
GRACE33.SVD.21372	1.43	1.06	1.2352

Table 6.15: Geoid Height Errors of all SVD Gravity Fields.

as

$$\begin{aligned} \tilde{b} &= \hat{X}_{\text{GRACE33.SVD.20986}} - \hat{X}_{UNBIASED} \\ \text{or} & \\ \tilde{b} &= \hat{X}_{\text{GRACE33.SVD.21372}} - \hat{X}_{UNBIASED}. \end{aligned} \tag{6.19}$$

The traces of the error variance-covariance matrix corresponding to GRACE33.SVD.-21372 and GRACE33.SVD.20986 have the scalar values

$$\begin{aligned} trace(P_{\hat{X}_{GRACE33.SVD.20986}}) &= 1.7715x10^{-08} \\ trace(P_{\hat{X}_{GRACE33.SVD.21372}}) &= 1.4188x10^{-08}, \end{aligned} \tag{6.20}$$

and the trace of the outer product of $\tilde{b}$ is expressed in general as

$$trace(\tilde{b}\tilde{b}^T), \tag{6.21}$$

True Soln.	KSV $trace(\tilde{\tilde{b}}\tilde{\tilde{b}}^T)$	KSV ratio	Norm $trace(\tilde{\tilde{b}}\tilde{\tilde{b}}^T)$	Norm ratio
EGM96	$4.7133x10^{-15}$	$3.3220x10^{-07}$	$4.7390x10^{-15}$	$2.6751x10^{-07}$
TEG4	$3.2587x10^{-15}$	$2.2968x10^{-07}$	$3.3181x10^{-15}$	$1.8730x10^{-07}$

Table 6.16: GRACE33(Case61): KSV and Norm-Norm SVD Ratios of Bias to Random Error.

so that a ratio of these two traces is given by

$$\text{ratio} = \frac{trace(\tilde{\tilde{b}}\tilde{\tilde{b}}^T)}{trace(P_{\hat{X}_{GRACE33.SVD}})}. \tag{6.22}$$

Fortunately, since the ratios given in Table 6.16 are much less than unity, the biases in GRACE33.SVD.21372 and GRACE33.SVD.20986 can be ignored (Bouman [64]), because they are small compared to random error in the SVD estimates. Therefore the KSV and Norm-Norm SVD solutions can be considered as unbiased in an approximate sense.

6.2.7.6 Discussion

The tool **PLASS** performed the Singular Value Decomposition (SVD) on an ill-conditioned system of equations, represented by the matrix R, to solve for 25917 parameters. It demonstrated a new feasibility in the application of the SVD in the solution for large gravity fields. Because the GRACE matrix R contains information derived from satellite only observations, it was expected that the attenuation of the short wave gravitational anomalies, those signals whose wavelengths are much shorter than the twin satellites' altitude, would case inflation in the estimation of their corresponding spherical harmonic coefficients.

Four common methods (Inspection, Relative Error, MSE, Norm Norm mini-

mization) and the technique introduced for this investigation, which relates Kaula's rule to singular value disposal, KSV, were written into **PLASS** and implemented to find the best solution among the many that can be generated by an SVD stabilization. All five methods of choosing singular values to obtain the best solution, each produced its own optimum gravity field. In the Inspection case, the disposal of 20000 singular values was deemed to be an optimum solution. In the Relative Error, MSE, KSV relation and Norm-Norm minimization cases, the removal of 1, 25914, 21372 and 20986 singular values, respectively, were found to be optimums. The solutions from the Relative Error and MSE techniques were immediately thrown out, as they were too close to the trivial solutions of either disposing all or none of the singular values. The optimum gravity fields produced by the Inspection method, Norm-Norm analysis and the KSV relation, were submitted for further evaluation

In all five solutions, there was bias error introduced into the estimate due to singular value truncation. Because bias is not quantified in the error variance-covariance matrix of an SVD estimate, this matrix is not an adequate measure of solution accuracy. Only by calculating the bias and comparing it to the random error, can one judge the quality of the solution. In the case of the KSV SVD and Norm-Norm stabilized gravity solutions, the comparison of the trace of the outer product of the bias vector $\tilde{b}$ to the trace of the error variance-covariance matrix, in the form of a ratio, showed that the bias was small compared to the random error.

It was shown that all geoid height errors became smaller with singular value removal, thus indicating an increase in confidence of the random errors. Although the SVD solutions, GRACE33.SVD.20986 (Norm-Norm) and GRACE33.SVD.-

21372 (KSV) are biased, it was determined that their bias is sufficiently small to consider them as non-biased estimates in an approximate sense. Their “shifted distance” away from the true state, due to their small bias, was also negligible in that there were no deleterious effects detected by the orbit fit and geoid analyzes of this investigation. Because the solutions stabilized using the Kaula Singular Value (KSV) relation and the Norm-Norm plot method are of an objective nature, which survived the scrutiny of this study, they are considered to be the best GRACE SVD gravity fields.

Chapter 7

Conclusion

7.1 Summary of PLASS

Parallel code was developed to apply the SVD to very large systems of equations. With enough processors, **PLASS** can handle very large problem sizes and maintain *scalability* (See Appendix B). Among the capabilities of **PLASS** to apply the EVD or the SVD to solve for very large gravity fields in the form of either the normal equation or orthogonalized equation, respectively, this tool can also perform the following analysis using parallel computation techniques and generate all output necessary for subsequent analysis (note: the term singular may be interchanged with the word eigen):

Inspection: Allows user to choose singular values for removal, from generated singular value distribution graph.

Relative Error: Implement the equation necessary for relative error analysis.

Mean Square Error: Implement all equations and matrix operations necessary for this analysis. Generates data for required graphs.

Norm Norm minimization: Implement all equations and matrix operations necessary for this analysis. Generates data for required graphs.

Kaula Singular Value (KSV) relation: Implement all equations and matrix operations necessary for this analysis. Generates data for required graphs.

Calculate any EVD/SVD Covariance Matrix: After eigenvalue/singular value removal has been determined, corresponding EVD/SVD covariance matrix can be calculated, if desired.

Calculate the Covariance "inplace" using Matrix R: Given the R matrix of the orthogonalized system of equations, covariance matrix can be calculated "inplace", if desired.

Geoid Height Error Calculations: Using any covariance matrix, whether internally computed or read in by **PLASS**, it can propagate and map the uncertainties to the geodetic surface of the Earth at grid resolution of 1 degree latitude and longitude, in little time (i.e. 8 hours of wall clock time for 200x200 TEG Gravity Field for entire globe).

Safeguards: **PLASS** can be stopped and then started again at the same point at anytime during the stages of the EVD/SVD process.

Projections: **PLASS** reports the exact number of total Givens' iterations necessary for convergence. This is used to calculate expected wall clock time for convergence to eigenvalues/singular values. Timings will vary depending on core memory availability, number of processors and processor speed.

Orthonormal Basis Space Option: User has the choice of obtaining EVD/SVD estimate with or without the null space or eigen space, whichever the case

may be.

7.2 Summary of Results

7.2.1 Choice of Optimum Singular/Eigen Values

The techniques of singular value removal in this investigation that turned out to be unreliable, for both CHAMP and GRACE, were the methods of Inspection and Relative Error. However, Inspection does have its use for the purpose of getting an initial rough idea of how much deflation is necessary.

The method of Mean Squared Error (MSE) in the case of CHAMP and GRACE provided well behaved graphs of noise and bias. However the resulting optimum number of used CHAMP eigenvalues was inadequate at removing enough inflation in the higher degree and order corrections to geopotential coefficients. The MSE applied to GRACE, removed inflation, but resulted in an unacceptable optimum number of used singular values, as it was too close to the trivial solution of discarding all singular values. Thus, the use of MSE should be used with caution.

The methods of Norm Norm minimization and the Kaula Singular Value (KSV) relation proved to be very dependable for both CHAMP and GRACE. Although these two techniques are reliable and recommended, the author tends to prefer the KSV relation, as it is less expensive to perform and specifically related to gravity field estimation.

7.2.2 Orbit Fit Computations

Orbit fit computations for selected satellites provided very useful insight on both CHAMP EVD and GRACE33 SVD gravity fields. In comparison to EIGEN1S, marked improvements in the GFZ1 satellite RMS fits were seen in all EIGEN1S.EVD stabilized gravity fields except for that of the Inspection case, which caused this spacecraft to crash into the atmosphere. This dramatic result was informative, as it revealed the fact that the geopotential coefficients of that case were nonsensical, even though their power spectrum did not violate Kaula's power rule. The RMS fits of all other satellites in the EVD stabilized fields were of negligible difference to the EIGEN1S case.

In the comparison of the SVD stabilized GRACE33 gravity fields to their full rank solution, no appreciable differences were seen, for the higher altitude satellites Topex, Lageos1, Lageos2, and Starlette. This makes sense, as their orbits would be little affected by the higher degree and order geopotential coefficients which were deflated from SVD stabilization. In the cases of the lower altitude satellites, GFZ1 and Stella, it can be seen that regularization of the gravity field increases their RMS fits from about 2 cm to 4 cm. The tradeoff of a maximum of 4.14 cm RMS for stabilized solutions is deemed worthwhile.

7.2.3 Geodetic Evaluation of the SVD Fields

The geoids of the EIGEN1S and GRACE33(Case61) were calculated and the appropriate differences between them and the geoids from the stabilized EVD and SVD solutions, respectively, were computed, plotted and examined. In all nom-

inated cases, the stabilized solutions contained sufficient information to be comparable to their respective reference solutions. In the case of GRACE, the noticeable effect of the SVD was to smooth the ripples seen in the reference GRACE33(Case61) geoid.

The geoid height errors of all CHAMP EVD fields and selected SVD fields of GRACE33 were calculated. The effect of eigenvalue/singularvalue removal on the magnitude of the variance-covariance matrices was revealed. Size of the random uncertainties decrease with eigenvalue/singularvalue removal, thus causing geoid height errors to decline.

7.2.4 Bias of the SVD Solution

The proper quality measure of a biased estimate is the mean square error instead of the error variance (Rummel [63]). For an adequate assessment of the quality of an SVD estimate, the error variance (the error variance-covariance matrix) must be compared to the bias error. To satisfy this standard, the biases were computed for the two best EVD/SVD estimates, in both the CHAMP and GRACE examples. All bias errors were determined to be small in comparison to all random errors. Thus, these four solutions can be considered to be non-biased in an approximate sense.

7.2.5 Conclusion

The following are conclusions of this study.

- **PLASS** demonstrated the viability, scalability and practicality of applying the

SVD to solve for large Earth gravity fields.

- The Kaula Singular Value (KSV) method of choosing singular values or eigenvalues is the best technique to stabilize a gravity field solution.
- Using Norm Norm minimization to choose singular values or eigenvalues is reliable.
- The methods of inspection, relative error and MSE are not reliable.
- The biases in the best stabilized solutions of this study are small enough to consider these solutions as "non-biased" in an approximate sense.
- The tradeoff between bias and overall error in the best stabilized solutions of this study is worthwhile.

7.2.6 Suggestions for Further Study

In the case of the MSE method to choose the optimal number of eigen/singular values, the bias portion of the trace of the mean square error matrix was approximated using Kaula's rule. It is suggested, that this MSE bias could be represented by using the coefficients of a standard gravity field, for example those of TEG4. Such a substitution may render the MSE method to be reliable. The results of such an experiment would be interesting to the author of this investigation.

Although Kaula's rule was well satisfied in the cases of both KEV EVD and KSV SVD stabilization, through eigenvalue and singular value disposal respectively, it may have been the case, that too many eigen/singular values were

thrown away. The number of retained eigen/singular values as seen in the plots of Figure 6.10 and Figure 6.40 were far into the region in which little change in alpha is seen as further eigen/singular values are discarded. (This is discussed further in Appendix C.) More useful information from the normal matrix N (CHAMP) or the orthogonalized matrix R (GRACE) could have been extracted and included into solution without significant loss of stabilization. Thus, modifying the criterion of the KEV/KSV relation by considering also in what region of the alpha curve are eigen/singular values being discarded, would be worthwhile. In other words, the "delta" changes in alpha should be monitored as eigen/singular values are thrown away. If Δalpha remains small after subsequent stabilized solutions, further eigen/singular value disposal is not necessary. An investigation into this aspect and into other methods of objective inversion stabilization techniques would be topics of worthy study.

Appendices

Appendix A

Validation of PLASS

This section describes the validation test that proves the algorithms used in PLASS1 and PLASS2 implement their SVD decompositions correctly. This validation test is written into the software so that it is performed concurrently as the algorithm is applied to the actual data. Therefore the user of PLASS receives the final residual result of the validation test along with the final result of the SVD applied to the actual input data. The validation test is described (in step format) as follows:

Step 1 Pick a random vector x and ANY real matrix R.

Step 2 Form $b = Rx$ (b is obviously in the range of R). Therefore $b \equiv Rx$

Step 3 Calculate $\hat{x} = R^{-1}b$ (using whatever method) where "$\hat{x}$" indicates computed x

Step 4 Compute $\hat{b} = R\hat{x}$

Step 5 Form the residual ratio $\frac{\|b-\hat{b}\|}{\|b\|} =$ exact 0,but why ??

Step 6 Substituting $b = Rx$ and $\hat{b} = R\hat{x}$ into step 5, yields

$\frac{\|Rx-R\hat{x}\|}{\|Rx\|}$.

Step 7 Substituting $\hat{x} = R^{-1}b$ into step 6 then yields

$\frac{\|Rx - RR^{-1}b\|}{\|Rx\|}$.

Step 8 Finally, substituting $b = Rx$ into step 7 gives

$\frac{\|Rx - RR^{-1}Rx\|}{\|Rx\|} = \frac{\|Rx - Rx\|}{\|Rx\|} = 0$ (exact theoretical).

Theoretically, this residual ratio will evaluate to exact zero iff the term "R^{-1}" is the true inverse of the original matrix R. **Obviously** the term "R^{-1}" for this investigation is the SVD calculated "psuedo-inverse". Using numerical computation however, this residual ratio will evaluate to be "small" iff the term "R^{-1}" is calculated *correctly*. It could never evaluate to "exact zero" because of, i.e. truncation (chopping) or round off errors, during the floating-point arithmetic. If the algorithm miscalculates the term "R^{-1}" or there is something wrong in the algorithm, then that "blunder" will manifest itself in the form of a large residual ratio.

Why do we use a ratio? Because a relative error in the residual is more interesting and informative than an absolute error in the residual. After all, we want to *compare* the computed answer to what the true answer is. The next concept to clarify is our definition of "small". That value is on the order of 1×10^{-16} or 1×10^{-15}.

If we use double precision IEEE floating point standard, the mantissa has 53 binary digits (about 16 decimal digits) of accuracy. Rounded arithmetic has a unit "u" roundoff largest relative error of about

$$u = 2^{-53} \approx 1 \times 10^{-16} \tag{A.1}$$

GRACE Data	PLASS1	PLASS2	matrix
matrix size n	residual	residual	condition number
5037	7.6381×10^{-15}	1.2810×10^{-14}	1.6764×10^{03}
14637	2.4107×10^{-14}	2.5443×10^{-14}	1.0982×10^{04}
25917	1.9014×10^{-14}	2.5992×10^{-14}	1.2505×10^{05}

Table A.1: **PLASS** GRACE Data Validation Results: relative residual

PLASS1	PLASS2	matrix size n = 1000
residual	residual	matrix condition number
6.403508×10^{-15}	6.901393×10^{-15}	1.0369×10^{00}
6.452276×10^{-15}	7.084092×10^{-15}	1.0181×10^{04}
6.420774×10^{-15}	7.211693×10^{-15}	1.0181×10^{06}
6.465525×10^{-15}	6.951665×10^{-15}	1.0181×10^{09}
6.595521×10^{-15}	7.113030×10^{-15}	1.0181×10^{15}

Table A.2: **PLASS** Random Data: relative residual, keep all singular values

This applies to an individual operation, but if our final computed results (after a great many operations) yield a relative error of this magnitude, then we can be very confident that our numerical algorithm, to calculate "R^{-1}", is sound and correct. The tables of this section present the residuals calculated in this validation test using both GRACE data and randomly generated matrices. Notice that regardless of matrix condition number, the residual ratio is always small. This is proof that the algorithms of PLASS (PLASS1 and PLASS2) are correct and can be trusted.

PLASS1	PLASS2	matrix size n = 1000
residual	residual	matrix condition number
6.786532×10^{-15}	7.017996×10^{-15}	1.0181×10^{18}
3.493306×10^{-15}	2.810962×10^{-15}	1.0181×10^{23}
6.612903×10^{-15}	7.138595×10^{-15}	∞

Table A.3: **PLASS** Random Data: relative residual, discard small/zero singular values

Appendix B

Scalability and Numerical Issues of the SVD

Because the SVD is rich in level 2 (Basic Linear Algebra Subprograms) BLAS, it may not be apparent at first how parallel SVD computation provides a benefit (order n squared operations while communicating only n squared data). For the problem sizes of this investigation, the scalability analysis, (such as that discussed in Section 2.2.3) which compares parallel performance of an algorithm against the performance implemented by a sequential machine, must be adjusted. A single processor is not able to calculate a large SVD solution, at least within a time scale of less than weeks or months of wall clock time, if it could satisfy the SVD memory requirements. Thus, a scalability analysis of **PLASS** is conducted, in which the performance behavior of **PLASS** for the two cases of data (CHAMP and GRACE) are discussed and compared.

This analysis reports performance per processor at some fixed *memory* per processor as discussed by van de Geijn et al. [48]. The term “flops”, used below, is an acronym for “**fl**oating **op**erations per **s**econd ”, while the performance acronym “MFLOPS” refers to “**m**ega **flop** per **s**econd”.

B.1 Scalability

B.1.1 PLASS1

PLASS1 contains the following three stages for an SVD solution. For each stage the flop count is given. (note: Householder updates never entail the explicit formation of the Householder matrix.)

- Stage 1: Bidiagonalization
- Stage 2: Convergence to Singular Values
- Stage 3: Applying Storage Array of Givens' Iterations to the right hand side and apply right side Householder reflections to r.h.s.

Bidiagonalization of the square R matrix (of size n), involves n left-side and (n-2) right side Householder reflections. Each calculation of a Householder reflection itself requires 3n flops. Since there are 2n-2 Householder reflections, we have a flop count of

$$6n^2 - 6n \tag{B.1}$$

required just to *compute* the Householder reflections. The flop count for applying a Householder reflection and updating the rest of the matrix is based on the following explanation. A left side reflection with Householder vector u on matrix R is given as:

$$\left(I - \alpha u u^T\right) R \tag{B.2}$$

Table B.1 summarizes how Equation B.2 is carried out for the ***first*** Householder reflection. Summing the flop count in Table B.1 we get $4n^2 + n$ flops. For subsequent

First Householder Transformation of R matrix size n $\left(I - \alpha u u^T\right) R$ may be rewritten as $\left(R - \alpha u u^T R\right)$	
Step	Description
R - α u R^Tu	$\longleftarrow$ rewritten as
let w = α R^Tu	$\longleftarrow$ $2n^2$ + n flops
R - u w^T	$\longleftarrow$ rank one update $2n^2$

Table B.1: Flop Count for Steps in the *first* Householder Transformation

left side transformations we have

$$\begin{array}{c} 4(n-1)^2 + (n-1) \\ 4(n-2)^2 + (n-2) \\ 4(n-3)^2 + (n-3) \\ \vdots \\ 4(n-n)^2 + (n-n) \end{array} \tag{B.3}$$

which may be expressed in summation form as

$$4 \sum_{i=0}^{n} (n-i)^2 + \sum_{i=0}^{n} (n-i) \,. \tag{B.4}$$

For the right side Householder transformations the operation is very similar, but the reflections are applied to the right side of matrix R

$$R \left(I - \alpha u u^T\right), \tag{B.5}$$

however there are (n-2) of them. Thus their flop count is expressed in the summation

$$4 \sum_{i=0}^{n-2} (n-i)^2 + \sum_{i=0}^{n-2} (n-i) \,. \tag{B.6}$$

Rewriting left and right side transformations together, rearranging and separating terms we have

$$\begin{aligned}\sum_{i=0}^{n}\left(4n^2+n\right) \;-\; \sum_{i=0}^{n}8n\,(i)+\sum_{i=0}^{n}4(i)^2-\sum_{i=0}^{n}(i)+\\ \sum_{i=0}^{n-2}\left(4n^2+n\right) \;-\; \sum_{i=0}^{n-2}8n\,(i)+\sum_{i=0}^{n-2}4(i)^2-\sum_{i=0}^{n-2}(i).\end{aligned} \tag{B.7}$$

Equation B.7 can be approximated in continuous integral form, where the independent discrete variable "i" is replaced by the continuous and independent variable x, namely

$$\begin{aligned}\left(4n^2+n\right)\int_{x=0}^{x=n}dx \;-\; 8n\int_{x=0}^{x=n}x\,dx+4\int_{x=0}^{x=n}x^2\,dx-\int_{x=0}^{x=n}x\,dx+\\ \left(4n^2+n\right)\int_{x=0}^{x=n-2}dx \;-\; 8n\int_{x=0}^{x=n-2}x\,dx+4\int_{x=0}^{x=n-2}x^2\,dx-\int_{x=0}^{x=n-2}x\,dx.\end{aligned} \tag{B.8}$$

Carrying out the integration, summing like terms and simplifying yields

$$\frac{8}{3}n^3-3n^2+4n-\frac{38}{3}, \tag{B.9}$$

as the number of flops required to bidiagonalize square matrix R of size n. The flops required to apply the left side Householder transformations to the right hand side (the b vector in Equation 5.24) is

$$2n^2+3n. \tag{B.10}$$

Adding Equations B.1, B.9 and B.10 yields the total flops required for **PLASS1** Stage 1, as

$$\frac{8}{3}n^3+5n^2-n-\frac{38}{3}. \tag{B.11}$$

During Stage 2 (“chasing the bulge”), each Givens’ rotation calculation requires 2 divisions, 3 multiplications, 1 addition and 1 square root. There are also four multiplications performed outside a single Givens’ rotation scheme for a total of 11 flops per one “bulge chasing” step. Since there are order $O(n^2)$ iteration *steps* necessary to converge the bidiagonal matrix to the diagonal matrix containing singular values, a close approximation of the total flops for chasing the “bulge” is

$$11n^2. \tag{B.12}$$

Applying the left side Givens’ rotations to the right hand side during convergence (the b vector in Equation 5.21) is also a part of Stage 2 and requires

$$3n^2 \text{ flops.} \tag{B.13}$$

Adding Equations B.12 and B.13 gives us the total flops for Stage 2

$$14n^2. \tag{B.14}$$

Stage 3 consists of two steps. First, the release of all the saved right side Givens’ rotations (from the storage array), applying them to the right hand side (the b vector in Equation 5.41), requiring

$$3n^2 \text{ flops}, \tag{B.15}$$

second, the application of the right side Householder reflections to the right hand side, which has the flop count of

$$2n^2 - 5n + 2\ . \tag{B.16}$$

Table B.2 summarizes the stages of **PLASS1**. Table B.3 displays computational statistics of **PLASS1** in calculating a CHAMP 100x100 gravity field.

PLASS1	
Stage 1	
$6n^2 - 6n$	compute Householder reflections
$\frac{8}{3}n^3 - 3n^2 + 4n - \frac{38}{3}$	Householder reflections on R and updates to R
$2n^2 + 3n$	apply left side Householder reflections to r.h.s.
$\frac{8}{3}n^3 + 5n^2 - n - \frac{38}{3}$	Stage 1 Total flops
Stage 2	
$11n^2$	chase the bulge
$3n^2$	apply left side Givens' rotations to r.h.s.
$14n^2$	Stage 2 Total flops
Stage 3	
$3n^2$	apply saved right side Givens' rotations to r.h.s
$2n^2 - 5n + 2$	apply right side Householder reflections to r.h.s.
$5n^2 - 5n + 2$	Stage 3 Total flops
Total	
$\frac{8}{3}n^3 + 24n^2 - 6n - \frac{32}{3}$	Overall Total Flop Count
Case	
3.76 $\times 10^{12}$ Flops	CHAMP 100x100
46.4 $\times 10^{12}$ Flops	GRACE 160x160

Table B.2: Flop Count for **PLASS1** SVD Solution

CHAMP 100x100 Gravity Field: n = 11216 parameters
vampir HP machine at GFZ: shared memory,
1 GHZ 1 GB per processor, peak performance 2 GFLOPS per processor
total number of Givens' rotations required to converge to eigen values:
expected: 125798656
actual: 53272938
all stages completed in one job submission

Stages	flops	time (hrs.)	no. of processors	MFLOPS/proc.
All	$\frac{8}{3}n^3 + 24n^2 - 6n - \frac{32}{3}$	12	6	15

Table B.3: CHAMP: **PLASS1** EVD Computational Statistics

B.1.2 PLASS2

PLASS2 contains the following four stages for SVD solution. For each stage the flop count is given. (note: Householder updates never entail the explicit formation of the Householder matrix.)

- Stage 1: Bidiagonalization
- Stage 2: Accumulation of Householder Reflections into matrix V_b
- Stage 3: Convergence to Singular Values
- Stage 4: Application of Singular Vector Space V to r.h.s.

Stage 1 of **PLASS2** is identical to that of **PLASS1**. Stage 2 of **PLASS2** applied to matrix R is the accumulation of the right side Householder reflections into the matrix V_b (described in Section 5.3.1.1) and has the following flop count (Golub and van Loan [23])

$$\frac{4}{3}n^3. \tag{B.17}$$

Stage 3 is the same as that of **PLASS1**, except the flop count required to update the accumulated matrix V_b with all the right side Givens' rotations must be included. That term is described as follows. There are order $O(n)^2$ Givens' iteration *steps* in chasing the bulge. But since we apply only the right side Givens' rotations to matrix V_b, there are approximately $\frac{1}{2}n^2$ matrix V_b updates. Because it requires 6n flops to update matrix V_b with one Givens' rotation, the total number of flops to update V_b until convergence to singular values is

$$6n\frac{1}{2}n^2 \longrightarrow 3n^3. \tag{B.18}$$

Thus the total flops required in Stage 3 of **PLASS2** is

$$3n^3 + 14n^2. \tag{B.19}$$

Stage 4 of **PLASS2** is multiplication of the final and completely updated matrix V_b, which is matrix V of Equation 5.50, by the fully updated right hand side (the vector b in Equation 5.51 multiplied by the inverted diagonal matrix of singular values in Equation 5.50). This operation is a matrix vector multiplication and involves

$$2n^2 \text{ flops}. \tag{B.20}$$

Table B.4 summarizes the stages of **PLASS2**. Tables B.6 and B.7 display computational statistics of **PLASS2** in calculating the CHAMP 100x100 gravity field. Tables B.5, B.8, B.9 and B.10 display computational statistics of **PLASS2** in calculating the GRACE 70x70, 120x120 and 160x160 gravity fields.

Table B.11 displays the results of the scalability (with respect to performance per processor) experiments carried out on the longhorn (IBM) machine. The problem size was increased along with the number of processors. The performance per processor stays more or less constant in SVD stage 1 and stage 3 and 4, but "dips" down somewhat in stage 2. Thus, the PLASS2 SVD algorithm can be considered "approximately" scalable with respect to MFLOPS at some fixed memory per processor.

Table B.12 displays the results of the scalability experiments carried out on the hopper (Origin2000) machine at NASA. The problem size (n=5035) was kept

PLASS2	
Stage 1	
$6n^2 - 6n$	compute Householder reflections
$\frac{8}{3}n^3 - 3n^2 + 4n - \frac{38}{3}$	Householder reflections on R and updates to R
$2n^2 + 3n$	apply left side Householder reflections to r.h.s.
$\frac{8}{3}n^3 + 5n^2 - n - \frac{38}{3}$	Stage 1 Total flops
Stage 2	
$\frac{4}{3}n^3$	accumulate right side Householder reflections
Stage 3	
$11n^2$	chase the bulge
$3n^2$	apply left side Givens' rotations to r.h.s.
$3n^3$	update V_b until convergence to singular values
$3n^3 + 14n^2$	Stage 3 Total flops
Stage 4	
$2n^2$	multiply matrix V against r.h.s
Total	
$7n^3 + 21n^2 - n - \frac{38}{3}$	Overall Total Flop Count
Case	
0.89 $\times 10^{12}$ Flops	GRACE 70x70 Gravity Field
9.87 $\times 10^{12}$ Flops	CHAMP 100x100 Gravity Field
21.9 $\times 10^{12}$ Flops	GRACE 120x120 Gravity Field
121.8 $\times 10^{12}$ Flops	GRACE 160x160 Gravity Field

Table B.4: Flop Count for **PLASS2** SVD Solution

GRACE 70x70 Gravity Field: n = 5035 parameters
longhorn IBM machine at Univ. of Texas at Austin
IBM Power4 System Shared Memory, 224 total processors,
Operating System: AIX
1.3 GHZ 1.792 GB per processor,
5.2 GFLOPS peak performance per processor
total number of Givens' rotations required to converge to singular values:
expected 25351225
actual 18728953
(subsequent solutions are attainable in minutes or less)

SVD Stage	flops	time(hrs.)	no. of processors	MFLOPS
1	$\frac{8}{3}n^3 + 5n^2 - n - \frac{38}{3}$	0.22	1	430
2	$\frac{4}{3}n^3$	0.11	1	439
3 and 4	$3n^3 + 16n^2$	0.88	1	121
Total	$7n^3 + 21n^2 - n - \frac{38}{3}$	1.21	1	205

Table B.5: GRACE: **PLASS2** SVD Computational Statistics (for first solution)

CHAMP 100x100 Gravity Field: n = 11216 parameters
vampir machine at GFZ: shared memory
per processor: 1 GHZ, 1 GB, peak performance 2 GFLOPS
total number of Givens' rotations required to converge to eigenvalues:
expected: 125798656
actual: 53272938
all stages completed in one job submission
(subsequent solutions are attainable in minutes or less)

EVD Stage	flops	time(hrs.)	no. of processors	MFLOPS
1	$\frac{8}{3}n^3 + 5n^2 - n - \frac{38}{3}$	21.09	3	17
2	$\frac{4}{3}n^3$	8.18	6	11
3 and 4	$3n^3 + 16n^2$	1.95	5	120
Total	$7n^3 + 21n^2 - n - \frac{38}{3}$	31.22	6	15

Table B.6: CHAMP: **PLASS2** EVD Computational Statistics (for first solution)

CHAMP 100x100 Gravity Field: n = 11216 parameters
vampir machine at GFZ: shared memory
per processor: 1 GHZ, 1 GB, peak performance 2 GFLOPS
total number of Givens' rotations required to converge to eigenvalues:
expected: 125798656
actual: 53272938
all stages completed in one job submission
(subsequent solutions are attainable in minutes or less)

EVD Stage	flops	time(hrs.)	no. of processors	MFLOPS
1	$\frac{8}{3}n^3 + 5n^2 - n - \frac{38}{3}$	10.73	6	16
2	$\frac{4}{3}n^3$	8.18	6	11
3 and 4	$3n^3 + 16n^2$	1.55	6	127
Total	$7n^3 + 21n^2 - n - \frac{38}{3}$	20.45	6	22

Table B.7: CHAMP: **PLASS2** EVD Computational Statistics (for first solution)

GRACE 120x120 Gravity Field: n = 14643 parameters
longhorn IBM machine at Univ. of Texas at Austin
IBM Power4 System Shared Memory, 224 total processors,
Operating System: AIX
1.3 GHZ 1.792 GB per processor,
5.2 GFLOPS peak performance per processor
total number of Givens' rotations required to converge to singular values:
expected 214417449
actual 146274384
(subsequent solutions are attainable in minutes or less)

SVD Stage	flops	time(hrs.)	no. of processors	MFLOPS
1	$\frac{8}{3}n^3 + 5n^2 - n - \frac{38}{3}$	3.36	2	346
2	$\frac{4}{3}n^3$	0.78	5	295
3 and 4	$3n^3 + 16n^2$	6.80	3	128
Total	$7n^3 + 21n^2 - n - \frac{38}{3}$	10.94	5	112

Table B.8: GRACE: **PLASS2** SVD Computational Statistics (for first solution)

GRACE 160x160 Gravity Field: n = 25917 parameters
longhorn IBM machine at Univ. of Texas at Austin
IBM Power4 System Shared Memory, 224 total processors,
Operating System: AIX
1.3 GHZ 1.792 GB per processor,
5.2 GFLOPS peak performance per processor
total number of Givens' rotations required to converge to singular values:
expected 671690889
actual 348039215
(subsequent solutions are attainable in minutes or less)

SVD Stage	flops	time(hrs.)	no. of processors	MFLOPS
1	$\frac{8}{3}n^3 + 5n^2 - n - \frac{38}{3}$	4	14	230
2	$\frac{4}{3}n^3$	5.25	16	77
3 and 4	$3n^3 + 16n^2$	7.5	12	161
Total	$7n^3 + 21n^2 - n - \frac{38}{3}$	16.75	16	126

Table B.9: GRACE: **PLASS2** SVD Computational Statistics (for first solution)

GRACE 160x160 Gravity Field: n = 25917 parameters
longhorn IBM machine at Univ. of Texas at Austin
IBM Power4 System Shared Memory, 224 total processors,
Operating System: AIX
1.3 GHZ 1.792 GB per processor,
5.2 GFLOPS peak performance per processor
total number of Givens' rotations required to converge to singular values:
expected 671690889
actual 348039215
(subsequent solutions are attainable in minutes or less)

SVD Stage	flops	time(hrs.)	no. of processors	MFLOPS
1	$\frac{8}{3}n^3 + 5n^2 - n - \frac{38}{3}$	5.08	7	363
2	$\frac{4}{3}n^3$	1.20	29	185
3 and 4	$3n^3 + 16n^2$	1.84	29	272
Total	$7n^3 + 21n^2 - n - \frac{38}{3}$	8.76	29	133

Table B.10: GRACE: **PLASS2** SVD Computational Statistics (for first solution)

Field Size	SVD Stage	time(hrs.)	no. of proc.	MFLOPS per proc.
70x70	1	0.22	1	430
120x120	1	3.36	2	346
160x160	1	5.08	7	363
70x70	2	0.11	1	439
120x120	2	0.78	5	295
160x160	2	1.44	16	281
70x70	3 and 4	0.88	1	121
120x120	3 and 4	6.80	3	128
160x160	3 and 4	8.80	10	128

Table B.11: GRACE: **PLASS2** SVD Scalability w.r.t. perf. per proc.

constant while the number of processors increased. Three processors were the minimum for stage 2. It can be seen that the performance degrades (as expected) consistently in all stages going from 8 to 16 processors. This was anticipated because the "data density" per processor goes down in this experiment. It is interesting to note that overall wall clock time improves (goes down) with increased processors, see Table B.13. (If the SVD algorithm were perfectly parallelizable, then the wall clock times would decrease by a factor equal to the number of increased processors.)The *performance* per processor appears to be low. But this is due to the fact that **none** of the operations in the SVD are matrix matrix multiplies, rather they are operations which involve matrix vector multiplications or something else of lower data density.

Table B.14 displays the scalability results with respect to wall clock time. It can be seen that the actual and theoretical (expected) times are fairly close for all stages of the PLASS SVD algorithm. In fact, for stages 3 and 4, the multiprocessor wall clock times performed better than expected. (Note: The reference time for a

SVD Stage	no. of proc.	time(min.)	MFLOPS per proc.
1	2	67.51	89
1	4	35.18	86
1	8	16.80	90
1	16	15.05	50
2	3	20.89	45
2	4	16.61	43
2	8	8.93	40
2	16	10.47	17
3 and 4	2	49.65	64
3 and 4	4	24.26	66
3 and 4	8	18.14	44
3 and 4	16	14.00	29

Table B.12: GRACE: **PLASS2** Const. Prob. Size w. Proc. Increase (for 1st soln.)

No. of Processors	overall MFLOPS per proc.	overall wall clock time (min.)
2	77	138.05
4	70	76.04
8	61	43.87
16	34	39.52

Table B.13: GRACE: **PLASS2** Const. Prob. Size w. Proc. Increase (for 1st soln.)

particular stage is that of a one-processor wall clock time measurement. Thus the expected and actual times will be equivalent when using a single processor for a particular stage.)

Thus, it is claimed that the parallel SVD algorithm developed for this study is *scalable* for two reasons. First: The PLASS SVD algorithm delivered approximately consistent performance for any particular SVD stage (MFLOPS on a per processor basis) as both problem size and number of processors increases. Second: Using the metric of wall clock time, PLASS's actual measurements were close to or better than expected (theoretical).

Note on performance numbers: The low MFLOPS reported in the above tables may raise an eyebrow. The main reason for this is that there are no matrix matrix operations in any of the stages of the SVD. All of the computations utilize either level 1 or level 2 BLAS. This means that the maximum number of operations at any one time is order n^2 using order n^2 data. There is however, one point in the algorithm (PLASS2) were a triangular matrix times a triangular matrix multiply does occur, but this step is not a full matrix matrix multiply. One may argue that it makes no sense to report performance numbers in terms of MFLOPS for the SVD, but the author believes it is of interest.

B.2 A few Words on some Numerical Issues in the SVD

Some important remarks about the effects of error on the singular value decomposition are given in this section. Expressions for the singular value perturbation bound and perturbation expansion are presented along with a note on the

Scalability w.r.t Time (problem size increases with processors)
GRACE 70x70 Gravity Field: n_1 = 5035 parameters
GRACE 120x120 Gravity Field: n_2 = 14643 = 2.91n_1 parameters
GRACE 160x160 Gravity Field: n_3 = 25917 = 1.77n_2 = 5.15n_1 parameters
longhorn IBM machine at Univ. of Texas at Austin
IBM Power4 System Shared Memory, 224 total processors,
Operating System: AIX
1.3 GHZ 1.792 GB per processor,
5.2 GFLOPS peak performance per processor
Stage 1 = $\frac{8}{3}n^3+$ ignore lower order terms
Stage 2 = $\frac{4}{3}n^3$
Stage 3 and 4 = $7n^3+$ ignore lower order terms

Grav.Field	No. of Proc.	SVD Stage	Expected Time (hrs.)	Actual
70x70	1	1	$t_{70} = \frac{\frac{8}{3}n_1^3}{1}$ = t_{70} = 0.22	0.22
120x120	2	1	$t_{120} = \frac{\frac{8}{3}(2.91n_1)^3}{2}$ = 12.32t_{70} = 2.71	3.36
160x160	7	1	$t_{160} = \frac{\frac{8}{3}(5.15n_1)^3}{7}$ = 19.51t_{70} = 4.29	5.08
70x70	2	2	$t_{70} = \frac{\frac{4}{3}n_1^3}{2}$ = t_{70} = 0.11	0.11
120x120	5	2	$t_{120} = \frac{\frac{4}{3}(2.91n_1)^3}{5}$ = 4.93t_{70} = 0.54	0.78
160x160	16	2	$t_{160} = \frac{\frac{4}{3}(5.15n_1)^3}{16}$ = 8.54t_{70} = 0.94	1.44
70x70	1	3,4	$t_{70} = \frac{7n_1^3}{1}$ = t_{70} = 0.88	0.88
120x120	3	3,4	$t_{120} = \frac{7(2.91n_1)^3}{3}$ = 8.21t_{70} = 7.23	6.80
160x160	10	3,4	$t_{160} = \frac{7(5.15n_1)^3}{10}$ = 13.66t_{70} = 12.02	8.80

Table B.14: GRACE: **PLASS2** SVD Problem Size increases w. Processors (for 1st soln.)

effect of perturbation to the singular vectors calculated in this study.

B.2.1 Perturbation Expressions for Singular Values

Errors arise from two sources: rounding-errors made in computing the singular value decomposition and errors initially present in the matrix (Stewart [74]). Rounding errors are generally unimportant, because if a stable algorithm is used to compute the decomposition, their effect is as if the original matrix had been very slightly perturbed (Stewart [74]). It is the effect of the errors initially present in the matrix, that is important to understand. If we let $\tilde{R} = R + E$ be a perturbation on matrix R, where E is a matrix representing the perturbation error, then

$$\tilde{U}^T \tilde{R} \tilde{V} = \left[\begin{array}{c} \tilde{\Sigma} \end{array} \right], \tag{B.21}$$

is the singular value decomposition of $\tilde{R}$. To understand how $\tilde{\Sigma}$ compares with Σ a perturbation bound for the singular values of a matrix are given by Mirsky [75] and Weyl [76],

Theorem B.2.1. ***Weyl*** $|\tilde{\sigma}_i - \sigma_i| \leq \|E\|_2$

Theorem B.2.2. ***Mirsky*** $\sqrt{\sum_i (\tilde{\sigma}_i - \sigma_i)^2} \leq \|E\|_F$,

where the spectral norm $\|\cdot\|_2$ is defined as

$$\|E\|_2 \equiv \max_{\|x\|_2 = 1} \|Ex\|_2 \tag{B.22}$$

and the Frobenius norm is defined by,

$$\|E\|_F \equiv \sqrt{\sum_{ij} \epsilon^2{}_{ij}}. \tag{B.23}$$

These two theorems hold, no matter what the size of the error is and they show that ordering the singular values by magnitude provides a natural pairing: we know immediately which singular value is near which (Stewart [74]). Weyl's theorem also states that the singular values of a matrix are perfectly conditioned, i.e. no singular value can move more than the norm of the perturbations (Stewart [74]).

An expression (Stewart [74]) of the expansion of the perturbations is as follows: Let $\sigma \neq 0$ be a simple singular value (it is not repeated) of matrix R with a left singular vector u and right singular vector v. Then as E approaches zero, there is a unique singular value $\tilde{\sigma}$ of $\tilde{R}$ such that

$$\tilde{\sigma} = \sigma + u^T E v + O(\|E\|_2). \tag{B.24}$$

Provided the perturbation (E) is small, this expansion is very accurate. The closer each σ is to its neighbors, the smaller E will be. If the separation between the singular values is δ, then the second order term is approximately bounded by $\|E\|/\delta$ (Stewart [74]). If the elements of E are independent and random with mean zero and standard deviation of ϵ and we ignore the second order term of Equation B.24, the perturbation in the ith singular value is ${u_i}^T E v_i$. Therefore the expected value of the sum of the squares of the errors in the singular values is (Stewart [74])

$$E\big[\sum_{i=1}^{n} {u_i}^T E {v_i}^2\big] = n\epsilon^2. \tag{B.25}$$

The expected value of the square of the Frobenius norm of E is

$$E[(\|E\|^2{}_F)] = mn\epsilon^2, \tag{B.26}$$

where m and n are the number of rows and columns of matrix R, respectively.

B.2.2 Perturbation of the Singular Vectors

Perturbation theory for singular vectors can be problematic when the singular values of a matrix are very close together. That means that the calculation of the singular vectors can be very sensitive to perturbations in the matrix being decomposed. It may be that small and arbitrary epsilon changes in R may cause enormous changes in its singular vectors. Fortunately, for the matrices in this study, this danger **does not exist at all**. The CHAMP singular values have as their smallest separation a distance of approximately 18 trillion. In the GRACE matrix, the minimum singular value separation is about 0.002. Although this number may only be a hundred or a thousand times larger than a perturbation in matrix R, the singular vectors corresponding to such singular values are easily eliminated before final solution. Also, the singular vectors of this investigation are used only in the transformations from the singular space to the solution space, they are not used in any other calculation. Furthermore, because the SVD is computed by a stable algorithm, the *computed* U, V and Σ satisfy

$$R + G = U\Sigma V^T, \tag{B.27}$$

where G represents a perturbation that is on the order of rounding the matrix R. The use of the perturbed left and right singular vectors in this case, amounts to a negligible change in the original problem (Stewart [74]). Figures B.1 and B.2 show the separation of eigenvalues and singular values for the CHAMP and GRACE matrices, respectively, of this study.

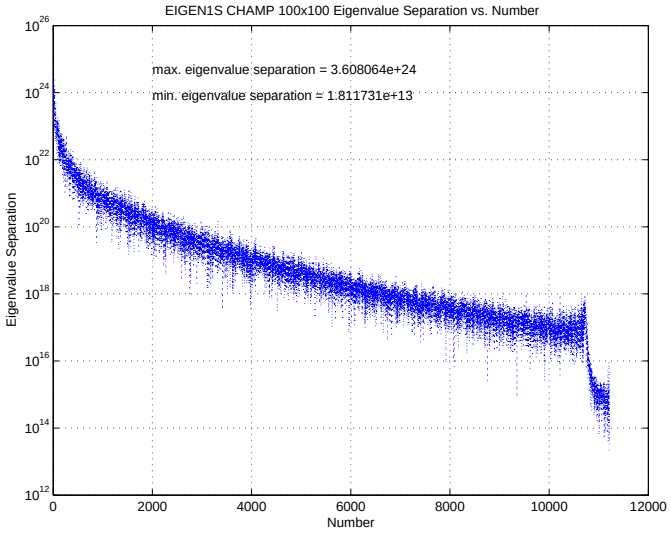


Figure B.1: CHAMP: Eigenvalue Separations

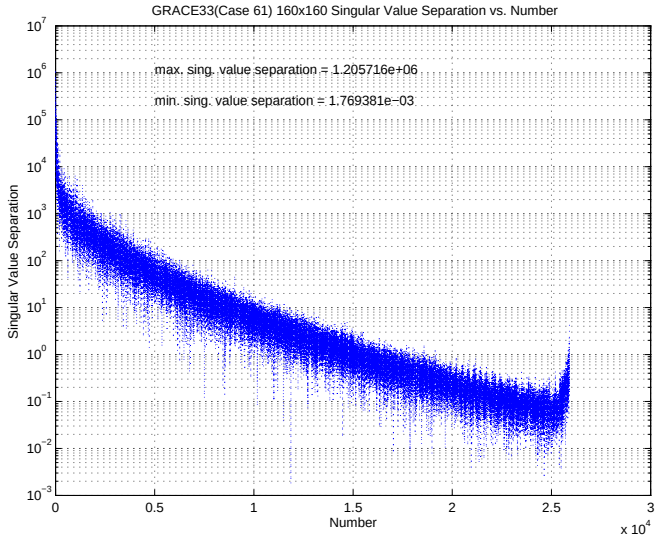


Figure B.2: GRACE33(Case61): Singular Value Separations

B.2.3 Roundoff Properties During Bidiagonalization

Roundoff properties associated with Householder matrices are very favorable. The Householder function of Golub and van Loan [23] used in this study produces a Householder vector $\hat{v}$ very near the exact v, see Wilkinson [70]. If Equation 5.3, repeated here for convenience,

$$\hat{Q} = I - \frac{2}{\hat{v}^T\hat{v}}\hat{v}\hat{v}^T, \tag{B.28}$$

then

$$\|\hat{Q} - Q\|_2 = O(\mathbf{u}), \tag{B.29}$$

meaning that $\hat{Q}$ is *orthogonal to machine precision*. The term **u** is unit roundoff and is defined in the model of floating point arithmetic (Golub and van Loan [23]) as $\mathbf{u} = \frac{1}{2}\beta^{1-t}$, where β is the *base* and t is the *precision*. Moreover, the computed updates with $\hat{Q}$ are close to the exact updates with Q:

$$\begin{aligned} fl(\hat{Q}R) &= Q(R+E), \quad \|E\|_2 = O(\mathbf{u}\|R\|_2) \\ fl(R\hat{Q}) &= (R+E)Q, \quad \|E\|_2 = O(\mathbf{u}\|R\|_2) \end{aligned}$$

B.2.4 Roundoff Properties During Givens' Plane Rotations

The numerical properties of Givens' rotations are as favorable as those for the Householder reflections (Golub and van Loan [23]). Specifically, it can be shown that the computed cosine and sine of the angles (see Equation 5.28), $\hat{c}$ and $\hat{s}$, respectively, in the Givens' function of this study, satisfy

$$\begin{aligned} \hat{c} &= c(1+\epsilon_c), \quad \epsilon_c = O(\mathbf{u}) \\ \hat{s} &= s(1+\epsilon_s), \quad \epsilon_s = O(\mathbf{u}). \end{aligned}$$

If $\hat{c}$ and $\hat{s}$ are subsequently used in a Givens' update, then the computed update is the exact update of a nearby matrix:

$$\begin{aligned} fl[\hat{G}(i,j,\theta)^T R] &= G(i,j,\theta)^T (R+E) \quad \|E\|_2 \approx \mathbf{u}\|R\|_2 \\ fl[R\hat{G}(i,j,\theta)] &= (R+E)G(i,j,\theta) \quad \|E\|_2 \approx \mathbf{u}\|R\|_2 \end{aligned}$$

The detailed error analysis of Givens' rotations may be found in Wilkinson [70].

B.2.5 A Word on the Accuracy of the Computed Singular Values

The subroutine used in this study to converge to singular values takes as the beginning input, the main and upper diagonals of the bidiagonalized matrix (see Section 5.2.2.1) and has the name sbdsqr.f90, written by Demmel and Kahan [69]. It was altered by the author of this study so that **PLASS** would "wrap" around this routine in order to perform the convergence to singular values on every processor in parallel fashion. The singular values obtained by the routine sbdsqr.f90 are determined to the same relative precision as the individual matrix entries of R (Demmel and Kahan [69]). Thus, if the matrix entries are known to high relative accuracy, then the singular values will be known to high relative accuracy, independent of their magnitudes. This algorithm by Demmel and Kahan is a method of computing the singular values of a bidiagonal matrix that *guarantees* high relative accuracy, regardless of their sizes. For details on convergence criteria and a further discussion of the perturbation theory, the curious reader is referred to the Demmel and Kahan paper [69].

Appendix C

GRACE33 (Case61) SVD Results compared to TEG4

One goal of this investigation was to perform the SVD on an ill-conditioned (full rank) GRACE system of equations and demonstrate that the stabilization can be achieved objectively using Kaula's rule within the scope of one particular information matrix. However, it is also important to compare the results of this large KSV SVD stabilized solution to that of other gravity fields. The Texas Earth Gravity 4 (TEG4) model is a field that is used as a nominal model in many of the gravity solutions produced by the Center for Space Research (CSR) and thus is used as another quality check.

Figure C.1 seen also in Section 6.2.6, repeated here for convenience, shows the search (granularity at 20 singular values) for the global minimum using the KSV relation. Notice the minimum occurs to the far left in the graph (4545), a region which is flat in the curve of alpha versus the number of singular values used. This "flatness" (high radius of curvature) suggests that this amount of singular value disposal, although satisfying Kaula's rule well, may be excessive. For example, if one continues to discard singular values beyond 15000, the incremental improvement to minimize the difference between the SVD solution and Kaula's rule may be inadequate to justify throwing away more information (of the orthogonalized in-

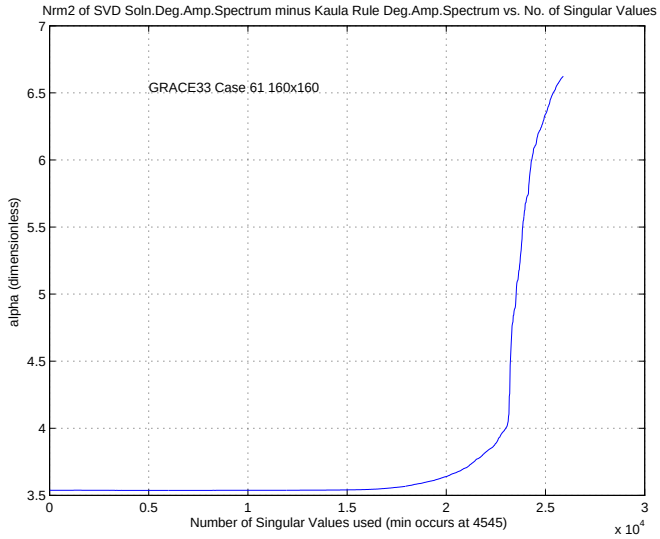


Figure C.1: GRACE33(Case61): Global Rough Sweep of Singular Value Disposal Cases

formation matrix). To support this claim the following figures are shown and for convenience, Figure C.2 seen also in Section 6.2.6 which best satisfies Kaula's rule is repeated. Looking at the so-called "triangle plot", Figure C.3, that corresponds to Figure C.2, it can be seen that too many singular values may have been truncated to best satisfy Kaula's rule.

The triangle plots display the ratio of contributed information from the SVD stabilization to that of the TEG4 solution (in a two-dimensional space of degree vs both negative and positive order coefficients). In other words one can see the amount of (gravity) signal being "added" to a nominal field after a KSV SVD inversion has been completed. For all of the triangle plots, the ratio of the difference in estimated coefficients between the SVD solution and the TEG4 to the TEG4, is displayed in the upper left plot. The upper right plot, illustrates the ratio of the

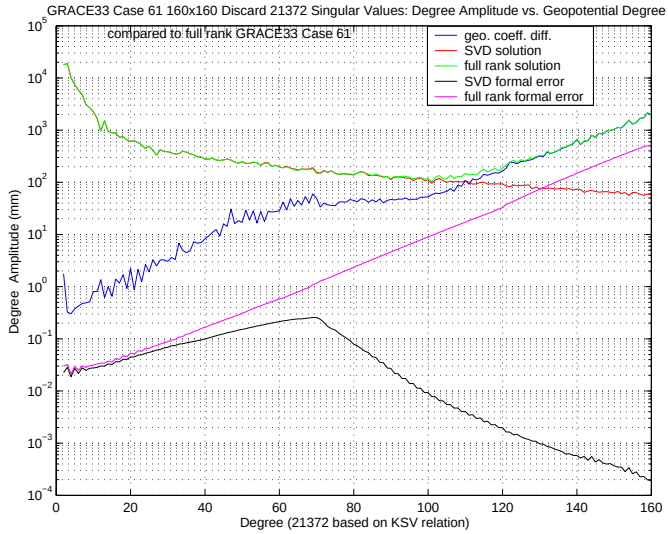


Figure C.2: KSV Degree Error Var. and Var. Geopotential Difference to Case61

SVD variances to the TEG4 variances. In the lower plot, the ratio of the difference in the estimated coefficients between the SVD and TEG4 to the TEG4 error variance, is presented in an attempt to further illustrate the discrepancy between the two solutions. The meaning of a such a ratio may indicate how "optimistic" an SVD solution is, or perhaps how much bias is being introduced by the truncation process. If a ratio is large or the SVD variances drop off too quickly with increasing degree, then the SVD solution may be interpreted as being too optimistic, as well as being too biased with respect to the TEG4 gravity field.

One can see clearly in the triangle plots, that much information has been thrown away for the best KSV SVD solution (GRACE33.SVD.21372). In this plot, the ratio of the SVD variances to the TEG4 variances reach a maximum value of 0.04, or 4 percent, in only a small portion of the plot. In the region between Degree 70 and 80, the ratios quickly drop to very low values. This indicates that little

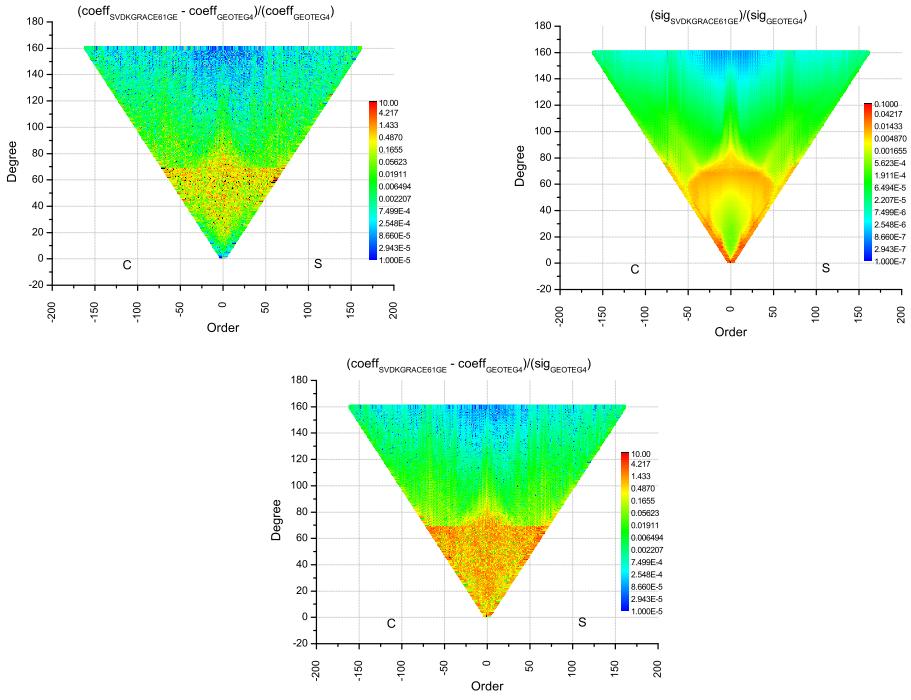


Figure C.3: KSV SVD Case Discard 21372 Singular Values Comparison to TEG4

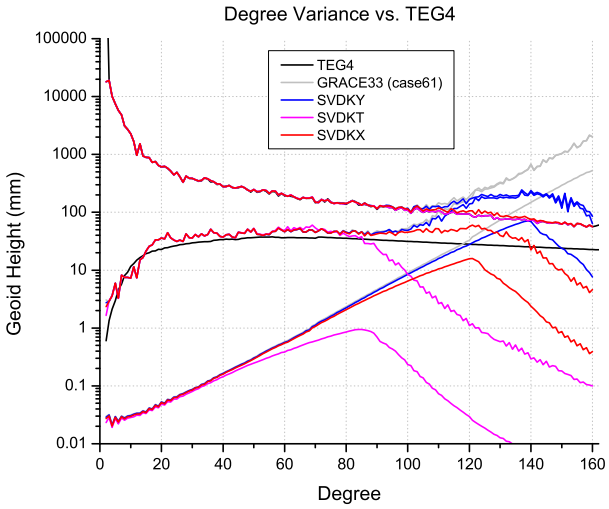


Figure C.4: Cases T,X,Y of KSV SVD Degree Variance vs. TEG4

information is included during/after inversion. This same behavior is evident also in Figure C.2, where the formal error drops quickly in the same region.

In an attempt to capture more information during the KSV SVD inversion, three more solutions were performed and plotted in the same manner. The choice of singular value disposal in these three new cases was based on visual observation of the function curvature in Figure C.1 and (randomly) choosing one local minimum seen in Figure 6.41 of Section 6.2.6. The points chosen were located at singular value disposal cases of 18560 (case T), 10917 (case X) and 5917 (case Y). All three cases are shown in Figure C.4 in the form of geoid height versus harmonic degree. As one discards fewer singular values, the drop in formal error variance is delayed to higher harmonic degrees, which is helpful, however the stabilization becomes less strong. The same trend/behavior is seen in the corresponding triangle plots, but

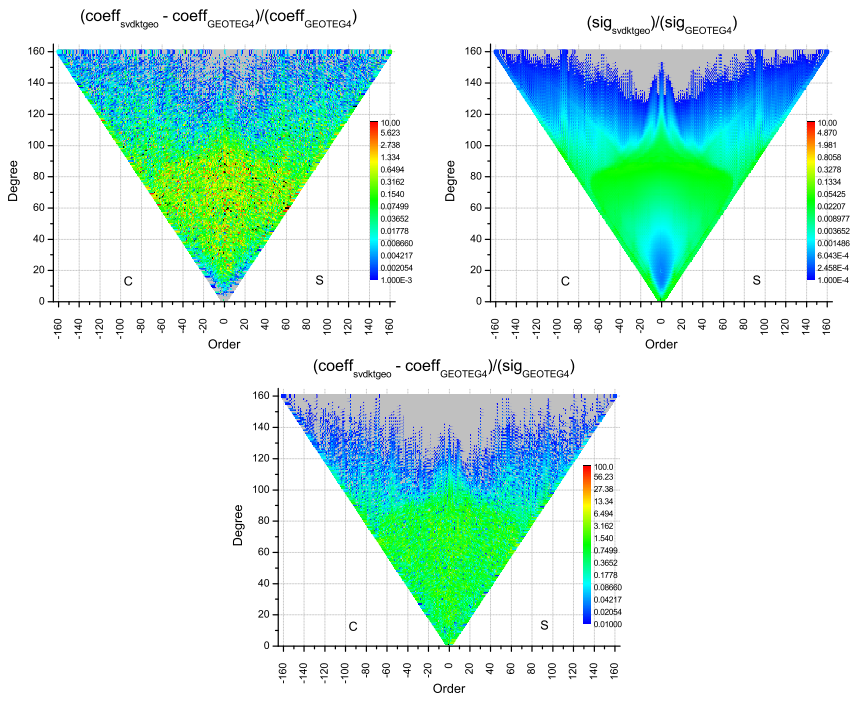


Figure C.5: KSV SVD Case T Discard 18560 Singular Values Comparison to TEG4

also illustrating how much more information is being added in each of the successive cases, T, X and Y, see Figures C.5, C.6, and C.7 respectively. In Figure C.5 the maximum "realistic" ratio of KSV SVD geo-coefficient information to TEG4 is somewhere between 65 and 100 percent throughout a significant portion of the upper left plot and drops off markedly in the 100 to 120 harmonic degree zone. Ratios above 100 percent may be nonsensical and perhaps indicate that some inflation is getting through during the inversion, however it could be (welcomed) gravity signal. The variance (sigma) ratios show a very similar, but smoother pattern. Areas of grey are to be interpreted as regions where there is zero added information.

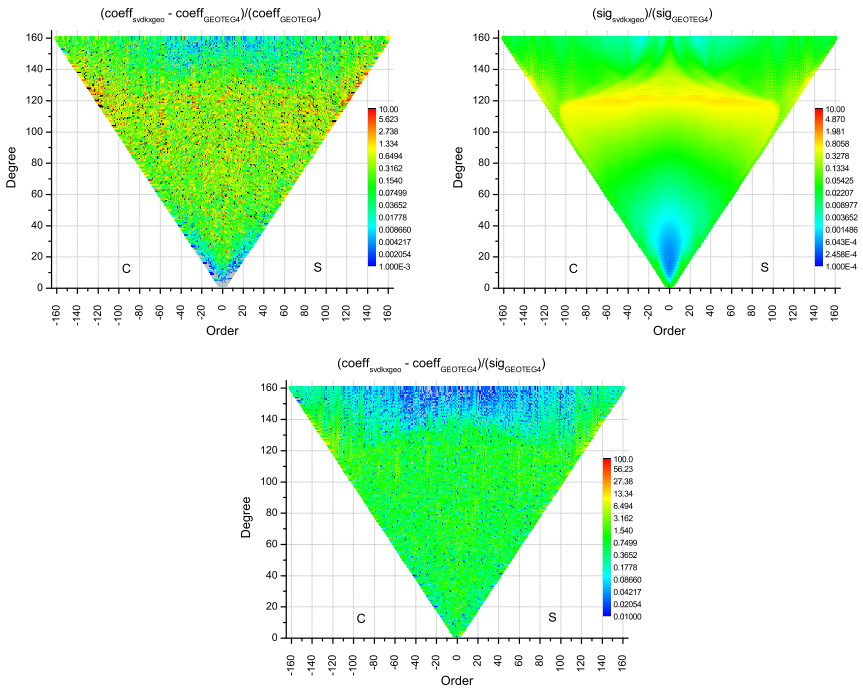


Figure C.6: KSV SVD Case X Discard 10917 Singular Values Comparison to TEG4

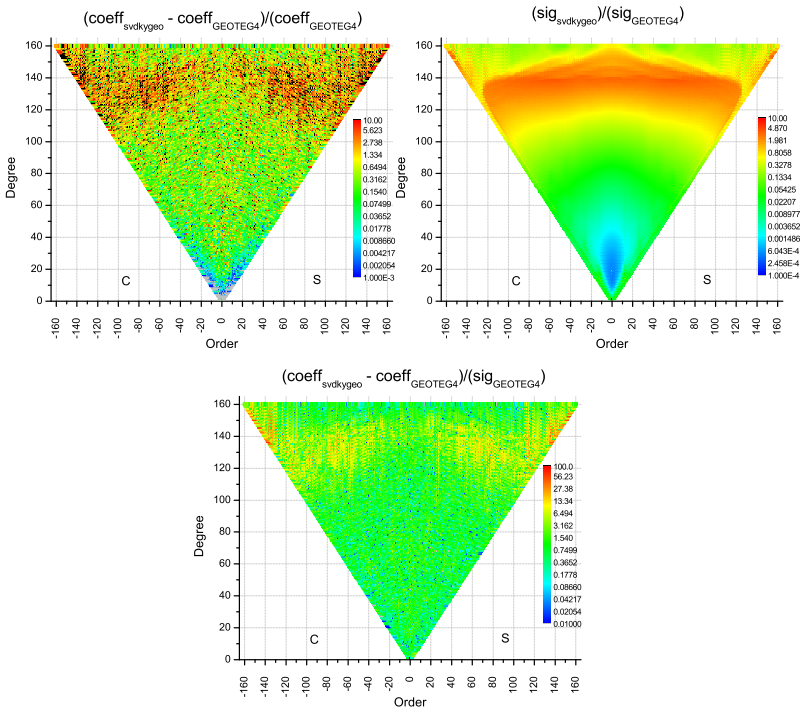


Figure C.7: KSV SVD Case Y Discard 5917 Singular Values Comparison to TEG4

It is interesting to note, that by adding 2812 singular values to the best KSV SVD solution (keeping a total of 7357, a region still well inside the "flat zone" of Figure C.1), seems to improve the field markedly with very little trade-off in stabilization, Figure C.5. The beneficial effect of adding another 7643 singular values (15000 total retained) is shown in triangle plot, Figure C.6. Notice that the significant drop in error variance is delayed further to the region of about degree 120. This may indicate that more information has been extracted out of the orthogonalized information matrix (R). There is also low trade-off in stabilization to solution. Figure C.7 shows the case where maybe too many singular values are kept (20000) and too much inflation is creeping through and hurting the solution. Much of the SVD error variance is larger than that of TEG4 (also seen in Figure C.4), because significant portions of its triangle plot are well above ratios of unity. One might also say, that the drop-off of SVD error variance is too much delayed, and harmful inflation is allowed through. Said in another way, the point at which the SVD formal error goes above that of TEG4, may correspond to the maximum number of singular values that should be included into the inversion for solution. It may also be interpreted, that because the ratio of the differences in coefficients to the TEG4 variances hovers around unity for most of the bottom triangle plot, this particular SVD solution (case Y) is over optimistic and too biased.

In any case, it is evident that curvature of the KSV SVD alpha function (of Figure C.1) should be taken into account along with Kaula's rule when searching for the best gravity solution stabilized by a truncated SVD inversion. Finally, it appears that after the point at which there is "little" change in the difference to

Kaula's rule, as seen in the alpha versus retained singular values curve, is the region where the **best** solution may be found for a particular ill-conditioned orthogonalized information matrix R.

Bibliography

[1] Vallado, *Fundamentals of Astrodynamics and Applications*, The McGraw Hill Companies, Inc., 1997.

[2] Vetter, J., Nerem, R. S., Cefola, P., and Hagar, H., "A Historical Survey of Earth Gravitational Models used in Astrodynamics from Sputnik and TRANSIT to GPS and TOPEX," *AAS/AIAA Astrodynamics Specialist Conference*, Vol. AAS 93-620, Aug. 16-19 1993.

[3] Stewart, R., Fu, L., and Lefebvre, M., "Science Opportunities for the Topex Poseidon Mission," Publication 86-18, JPL, NASA Jet Propulsion Laboratory, Pasadena, California, 1986.

[4] Council, N. R., "Satellite Gravity and the Geosphere," National Academy Press, 2101 Constitution Ave., NW.

[5] Gunter, B. C., *The Treatment of Processing Error in the Estimation of the Earth's Gravity Field*, Ph.D. thesis, The University of Texas at Austin, Center for Space Research, 2004.

[6] Yuan, D. N., "Large Linear System Solver(LISS) Program Document," Tech. rep., Center for Space Research, The University of Texas at Austin, 1991.

[7] Watkins, D. S., *Fundamentals of Matrix Computations*, John Wiley and Sons, Inc., 1991.

[8] Demmel, J., Heath, M., and van der Vorst, H., "Parallel Numerical Linear Algebra," *Acta Numerica*, Vol. Computer Sciences Division Tech Report, No. UCB//CSD-92-703, U.C. Berkeley, October 1992.

[9] Hansen, P., *Rank-Deficient and Discrete Ill-Posed Problems*, Ph.D. thesis, UNI C, Technical University of Denmark, Lyngby, Denmark, 1995.

[10] Hansen, P., "Analysis of Discrete Ill-Posed Problems by Means of the L-curve," *SIAM Review*, Vol. 34, 1992, pp. 561–580.

[11] Hansen, P., "The Use of the L-curve in the Regularization of Discrete Ill-Posed Problems." *SIAM J. Sci. Comput.*, Vol. 14, 1993, pp. 1487–1503.

[12] Ehrig, R. and Nowak, U., "Algorithmen und Softwareoptimierung fuer die Satellitebahn und Schwerefeldmodellierung," Tech. Rep. 00-1, Konrad-Zuse-Zentrum fuer Informationstechnik Berlin, Takustrasse 7, D-14195, Berlin-Dahlem, Germany 2000, Januar.

[13] Reigber, C., "Gravity Field Recovery from Satellite Tracking Data," *Theory of Satellite Geodesy and Gravity Field Determination*, Vol. Springer, Berlin, 1989, pp. 197–234.

[14] Golub, G., Heath, M., and Wahba, G., "Generalized Cross-Validation as a Method for Choosing a Good Ridge Parameter," *Technometrics*, Vol. 21, 1979, pp. 215–223.

[15] Hanke, M. and Raus, T., "A General Heuristic for Choosing the Regularization Parameter in Ill-Posed Problems," *SIAM J. Sci. Comput.*, Vol. 17, 1996, pp. 956–972.

[16] Neumaier, A., "Solving Ill-Conditioned and Singular Linear Systems: A Tutorial on Regularization," *SIAM Review*, Vol. 40, 1998, pp. 636–666.

[17] Bouman, J., *The Normal Matrix in Gravity Field Determination with Satellite Methods. Its Stabilization, Its Information Content and Its Use in Error Propagation*, Master's thesis, Delft University of Technology, 1993.

[18] Bouman, J., "Quality Assessment of Geopotential Models by Means of Redundancy Decomposition," *DEOS Progress Letters*, Vol. 97, 1997, pp. 49–54.

[19] Bouman, J., "Quality of Regularization Methods," Tech. Rep. 98.2, Delft University of Technology, 1998.

[20] Floberghagen, R. and Bouman, J., "On the Information Content and Regularization of Lunar Gravity Field Solutions." *DEOS Progress Letters*, Vol. 98, 1998, pp. 1–19.

[21] Koop, R. and Bouman, J., "Quality Differences between Tikhonov Regularization and Generalized Biased Estimation in Gradiometric Analysis," *DEOS Progress Letters*, Vol. 97, 1997, pp. 42–48.

[22] Koop, R. and Bouman, J., "Stabilization of Global Gravity Field Solutions by combining Satellite Gradiometry and Airborne Gravimetry." *DEOS Progress Letters*, Vol. 98, 1998, pp. 43–45.

[23] Golub and van Loan, *Matrix Computations*, The Johns Hopkins University Press, 3rd ed., 1996.

[24] Lawson, C. and Hason, R., *Solving Least Squares Problems*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1974, 340 pages.

[25] McCord, S. and Breckheimer, P. J., "Graphical Approach to the Solution of Least Square Problems Using Singular Value Analysis," Tech. rep., Jet Propulsion Laboratory, Pasedena, California, 1989.

[26] Lightsey, E. G., *Development and Flight Demonstration of a GPS Receiver for Space*, Ph.D. thesis, Stanford, 1997.

[27] Cicci, D., *Optimal A Priori Covariance Selection for the Solution of Ill-Conditioned Nonlinear Inverse Problems*, Ph.d. dissertation, The University of Texas at Austin, Austin, Texas, 1987.

[28] Cicci, D., "Improving Gravity Field Determination in Ill-Conditioned Inverse Problems," *Computer and Geosciences*, Vol. 18, No. 5, 1992, pp. 509–516.

[29] Lerch, F. J., Iz, H. B., and Chan, J. C., "Gravity Model Solution based upon SLR Data Using Eigenvalue Analysis: Alternative Methodology," *Contributions of Space Geodesy to Geodynamics: Earth Dynamics*, Vol. 24, 1993, pp. 213–219.

[30] Ahn, K. S., "Application of Singular Value Decomposition to Gravity Field Model Development using Satellite Data," Tech. Rep. 956689, Center for Space Research, The University of Texas at Austin, 1996.

[31] Inc., I., *IMSL User's Manual*, Houston, Texas, 1987.

[32] Dongarra, J. and et al., *LINPACK User's Guide*, Society for Industrial and Applied Mathematics, Philadelphia, siam publications ed., 1979.

[33] Gunter, B. C., *Aesop User's Guide*, University of Texas at Austin,Center for Space Research (CSR), Austin, Texas, csr technical memorandum ed., 2004.

[34] Press, W., Vetterling, W., Teukolsky, S., and Flannery, B., *Numerical Recipes in C*, Cambridge University Press, The Pitt Building, Trumptington Street, Cambridge CB2 1RP, second edition ed., 1992.

[35] Reilly, J., "EE731 Lecture Notes: Matrix Computations for Signal Processing," *Department of Electrical and Computer Engineering, McMaster University*, , No. 3, February 2000, pp. 1–21.

[36] Brent, R. and Luk, F., "The Solution of Singular Value and Symmetric Eigenvalue Problems on Multiprocessor Arrays." *SIAM J. Sci. Stat. Comput.*, Vol. 6, 1985, pp. 69–84.

[37] van de Geijn, R., Morrow, G., Gunnels, J., Howard, J., Joffrain, T., Overfelt, J., Riviere, B., Alpatov, P., Baker, G., Edwards, C., and Wu, J., *Using PLAPACK*, The MIT Press, 1997.

[38] Braun, T. D., Maciejewski, A. A., and Siegel, H. J., "A Parallel Algorithm for Singular Value Decomposition as Applied to Failure Tolerant Manipulators," Tech. rep., School of Electrical and Computer Engineering Purdue University, West Lafayette, IN 47907-1285 USA.

[39] Berry, M., Mezher, D., Phillipe, B., and Sameh, A., "Parallel Computation of the Singular Value Decomposition," *Institut National de Recherche en Informatique et en Automatique (INRIA)*, , No. 4694, January 2003, pp. 54 pages.

[40] Pan, Y. and Hamdi, M., "Singular Value Decomposition on Processor Arrays with a Pipelined Bus System," *Journal of Network and Computer Applications*, Vol. 19, 1996, pp. 235–248.

[41] Jessup, E. and Sorensen, D. C., "A Parallel Algorithm for Computing the Singular Value Decomposition of a Matrix," *SIAM J. Matrix Anal. Appl.*, Vol. 15, No. 2, March 1994, pp. 530–548.

[42] Jessup and Sorensen, D., "A Parallel Algorithm for Computing the Singular Value Decomposition of a Matrix," Tech. Report ANL/MCS-TM-102, Argonne National Laboratory, Argonne, IL, 1987.

[43] Bischof, C., "Computing the Singluar Value Decomposition on a Distributed System of Vector Processors," *Parallel Computing*, Vol. 11, 1989, pp. 171–186.

[44] Shroff, G. and Schreiber, R., "On the Convergence of the Cyclic Jacobi Method for Parallel Block Orderings," *SIAM J. Mat. Anal. Appl.*, Vol. 10, 1989, pp. 326–346.

[45] Tapley, Born, and Schutz, *Statistical Orbit Determination (in press)*, Academic Press Inc., 1999.

[46] Gropp, W., Lusk, E., and Skjellum, A., *Using MPI Portable Parallel Programming with the Message-Passing Interface*, The MIT Press, Cambridge, Massachusetts, London, England, 1999.

[47] Flynn, M. K. and Rudd, K. W., "Parallel Architectures," *ACM Computer Surveys*, Vol. 28, No. 1, March 1996, pp. 67–70.

[48] Dongarra, J. J., van de Geijn, R. A., and Walker, D. W., "Scalability Issues Affecting the Design of a Dense Linear Algebra Library," *Journal of Parallel and Distributed Computing*, Vol. 22, No. 3, September 1994, pp. 523–537.

[49] Gustafson, J., Montry, G., and Benner, R., "Development of Parallel Methods for a 1024-Processor Hypercube," *SIAM Journal of Scientific and Statistical Computing*, Vol. 9, No. 4, 1988, pp. 609–638.

[50] Saad, Y. and Schultz, M. H., "Data Communication in Parallel Architectures," *Parallel Computing*, Vol. 11, No. 2, Aug. 1989, pp. 131–150.

[51] van de Geijn, R., Payne, D., Shuler, L., and Watts, J., *A Streetguide to Collective Communication and its Application*, 1996.

[52] http://www.netlib.org/scalapack/.

[53] Edwards, C., Geng, P., Patra, A., and van de Geijn, R., "Parallel Matrix Distributions: Have we been doing it all wrong?" Technical Report TR-95-40, Department of Computer Sciences, The University of Texas at Austin, 1995.

[54] Bate, Mueller, and White, *Fundamentals of Astrodynamics*, Dover Publications, Inc., 1971.

[55] Gelb, A., *Applied Optimal Estimation*, M.I.T. Press, Cambridge, Mass., 1974.

[56] Tapley, B. D., "Statistical Orbit Determination Theory," *Recent Advances in Dynamical Astronomy*, 1972, pp. 396–425.

[57] Hinga, M. B. and van de Geijn, R. A., "Formal Methods of Covariance Calculations," *TBD*.

[58] Poole, S. R., "Second Variation Derivation of LUMVE," notes.

[59] Nerem, R., *Determination of the General Ocean Circulation Using Satellite Altimetry from a Simultaneous Solution for the Earth's Gravity Field*, Ph.D. thesis, University of Texas at Austin, Austin, Texas, 1989.

[60] Lerch, F., Nerem, R., Chinn, D., Shan, J., Patel, G., and Klosko, S., "New Error Calibration Tests for Gravity Models Using Subset Solutions with Independent Data: Applied to GEM-T3," *Geophysical Research Letters*, Vol. 20, No. 2, Feb. 5. 1993, pp. 249–252.

[61] Golub, G. and Kahan, W., "Calculating the Singular Values and Pseudo-inverse of a Matrix," *SIAM J. Num. Anal. Ser. B*, Vol. 2, No. 2, 1965, pp. 205–224.

[62] Belsley, D., Kuh, E., and Welsch, R. E., *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*, John Wiley and Sons, 1980.

[63] Xu, P. and Rummel, R., "Generalized Ridge Regression with Applications in Determination of Potential Fields," *manuscripta geodaetica*, Vol. 20, 1994, pp. 8–20.

[64] Bouman, J. and Keep, R., "Quality Differences between Tikhonov Regularization and Generalized Biased Estimation in Gradiometric Analysis," *DEOS Progress Letters*, , No. 1, 1997, pp. 42–48.

[65] Reigber, C., Balmino, G., Schwintzer, P., Biancale, R., Bode, A., Lemoine, J., Koenig, R., Loyer, S., Neumayer, H., Marty, J., Barthelmes, F., Perosanz, F., and Zhu, S. Y., "A High Quality Global Gravity Field Model from CHAMP GPS Tracking Data and Accelerometry (EIGEN-1S)," *Geophysical Research Letters*, Vol. 29(14), 2002.

[66] Tapley, B., Bettadpur, S., Chambers, D., Cheng, M., Choi, K., Gunter, B., Kang, Z., Kim, J., Nagel, P., Ries, J., Rim, H., Roesset, P., and Roundhill, I., "Gravity Field Determination from CHAMP using GPS Tracking and Accelerometer Data: Initial Results," *EOS Trans. AGU*, Vol. Fall Meet. Suppl. 82(47), No. Abstract G51A-0236, 2001.

[67] Lemoine, F. G., Kenyon, S. C., Factor, J. K., Trimmer, R., Pavlis, N. K., Chinn, D. S., Cox, C. M., Klosko, S. M., Luthcke, S. B., Torrence, M. H., Wang, Y. M., Williamson, R. G., Pavlis, E. C., Rapp, R. H., and Olson, T. R., "The Development of the Joint NASA GSFC and NIMA Geopotential Model EGM96," Tech. Rep. NASA/TP-1998-206861, NASA Goddard Space Flight Center, Greenbelt, Maryland, 20771 USA, 1998 July.

[68] Tapley, B., Schutz, B., Ries, J., and Rosborough, G. W., "The UT/Center for Space Research Orbit Determination Systems," *The XXVth Cospar Plenary Meeting, in Graz, Austria*, 1984.

[69] Demmel, J. and Kahan, W., "Accurate Singular Values of Bidiagonal Matrices," *SIAM J. Sci. Stat. Comput.*, Vol. 11, No. 5, 1990, pp. 873–912.

[70] Wilkinson, J., *The Algebraic Eigenvalue Problem*, Oxford University Press, 1965.

[71] Parlett, B., *The Symmetric Eigenvalue Problem*, Prentice Hall, Englewood Cliffs, New Jersey, 1980.

[72] Condi, D. F., "geoid calculation software," from personal communication.

[73] Schrama, E. J., "Error Propagation and Correlation Analysis of Covariance Matrices," Tech. Rep. Code 926, Goddard Space Flight Center, Greenbelt Md. 20771, June 26 2001.

[74] Stewart, G. W., "Perturbation Theory for the Singular Value Decomposition," Tech. rep., Dept. of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742, September 1990.

[75] Mirsky, L., "Symmetric Gauge Functions and Unitarily Invariant Norms," *Quarterly Journal of Mathematics*, Vol. 11, 1960, pp. 50–59.

[76] Weyl, H., "Das asymptotische Verteilungsgesetz der Eigenwert linearer partieller Differentialgleichungen (mit einer Anwendung auf der Theorie der Hohlraumstrahlung)." *Mathematische Annalen*, Vol. 71, 1912, pp. 441–479.

[77] Hinga, M. B., Poole R., S., Tapley, B. D., and van de Geijn, R. A., “Application of the SVD in Gravity Field Estimation,” *TBD*.

[78] Stewart, G., “On the Early History of the Singular Value Decomposition,” TR -92-31, Institute of Advanced Computer Sciences, Department of Computer Science, University of Maryland, College Park, Maryland, March 1992.

Vita

Mark Brandon Hinga was born in Denver, Colorado on 15 February 1968, the son of John P. Hinga and Agnes E. Hinga. He received the Bachelor of Science in Aerospace Engineering in 1990, then the Bachelor of Arts in Germanic Studies (*Magna Cum Laude*) and the Master of Science in Aerospace Engineering in 1994, all from the University of Colorado at Boulder. Just after graduation Mark worked/volunteered as an emergency medical technician in a Denver hospital. In 1995 he started employment as an astronautical and software engineer for Lockheed Martin Astronautics. After working on two classified projects he applied to the University of Texas at Austin for enrollment in their orbital mechanics program. He was accepted and started graduate studies in August 1998.

Permanent address: 3220 Duval Rd, Apt. 2224
Austin, Texas 78759-3531

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.