Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

RAINALD EHRIG   ULRICH NOWAK

Algorithmen- und Softwareoptimierung für die Satellitenbahn– und Schwerefeldmodellierung

ZIB-Report 00-01 (Januar 2000)

Algorithmen- und Softwareoptimierung für die Satellitenbahn– und Schwerefeldmodellierung

Rainald Ehrig, Ulrich Nowak

ehrig@zib.de, nowak@zib.de

Zusammenfassung

Dieser Report enthält die Ergebnisse der Untersuchungen, die gemäß dem Forschungs– und Entwicklungsvertrag **Gravity** zwischen dem GeoForschungsZentrum Potsdam und dem Konrad–Zuse–Zentrum Berlin vorgenommen wurden.

Inhaltsverzeichnis

1 Einleitung

Zwischen dem GeoForschungszentrum Potsdam (GFZ), Aufgabenbereich Kinematik und Dynamik der Erde, und dem Konrad–Zuse–Zentrum Berlin wurde am 7. Juli 1999 ein Forschungs– und Entwicklungsvertrag mit dem Titel Gravity abgeschlossen. Die damit vereinbarte wissenschaftliche Kooperation hat die folgenden Ziele:

- die am GFZ vorhandenen Algorithmen und Softwaremodule auf ihre Effizienz hinsichtlich Nutzung der Rechnerresourcen zu untersuchen und Lösungen für einen schnelleren Datendurchsatz zu entwickeln und zu implementieren,
- Methoden zur Regularisierung und Lösung schlecht konditionierter Normalgleichungssysteme (für Schwerefeldkoeffizienten) kritisch zu untersuchen und eine mathematisch objektive Strategie der Regularisierung zu entwickeln, und
- insbesondere im Hinblick auf die Anforderungen bei GRACE, verschiedene Bahnintegrationsverfahren hinsichtlich ihrer numerischen Genauigkeit und Einsatzmöglichkeiten zu untersuchen.

Hintergrund dieser Problemstellungen sind die für 2000 und 2001 geplanten Geopotentialmissionen CHAMP und GRACE, für die das GFZ die wissenschaftlichen Auswertungen übernehmen wird. Diese Missionen sollen das globale Erdschwerefeld und dessen räumliche und zeitliche Änderung sehr viel genauer als bisher vermessen. Die extreme Beobachtungsgenauigkeit erfordert eine erhöhte Rechen– und Modellierungsgenauigkeit, die große Zahl der Beobachtungen (100 000/Tag) und die der relevanten Schwerefeldparameter (10 000 – 30 000) die Verwendung hocheffizienter Verfahren. Die entscheidenden mathematisch–numerischen Aufgaben, die hierbei auftreten, sind die folgenden:

- numerische Satellitenbahnintegration,
- Aufstellen und Akkumulation der Normalgleichungssysteme für die Schwerefeldparameter,
- Regularisierung und Lösung der Normalgleichungssysteme.

Der vorliegende Bericht enthält die Ergebnisse, die für diese drei Teilaspekte bisher erhalten wurden.

2 Untersuchung der vorhandenen Algorithmen und Softwaremodule auf Effizienz

In diesem Abschnitt werden die grundlegenden numerischen Verfahren bei der Aufstellung und Lösung der Normalgleichungssysteme hinsichtlich Effizienz und Robustheit analysiert. Weitherhin werden verschiedene sequentielle und parallele Varianten vorgeschlagen.
Alle numerischen Methoden, die zur Schwerefeldinversion benötigt werden, beruhen im Kern auf Basisroutinen der linearen Algebra. Für derartige Aufgabenstellungen existieren äußerst zuverlässige und vielseitige Bibliotheken. Hier sind insbesondere die frei verfügbaren Bibliotheken BLAS [1] und LAPACK [2] zu nennen. Die BLAS–Blibliothek umfaßt grundlegende Operationen mit Vektoren und Matrizen. Darauf aufbauend bietet die LAPACK–Bibliothek eine vielfältige Sammlung von Programmen etwa zur Faktorisierung von Matrizen, Eigenwertbestimmung, SVD–Analyse und anderen üblichen Aufgaben der linearen Algebra. Diese Routinen werden zudem von vielen Herstellern von Workstations oder PCs in maschinennah optimierter Form als Teil von Entwicklungsumgebungen angeboten. Dadurch sind Programme, die sich solcher Bibliothek bedienen, fast unbeschränkt übertragbar und gewährleisten gleichzeitig optimale Effizienz. Auf Mehrprozessor–Anlagen sind diese Programme oft auch in parallelisierter Implementierung vorhanden, so daß der Benutzer ohne eigene Aufwendungen in Richtung einer Parallelisierung eine weitere erhebliche Beschleunigung erwarten kann. Daher sollten nach Möglichkeit alle Programme auf der Basis von BLAS– und LAPACK–Routinen formuliert werden. Dies erhöht zudem die Fehlersicherheit, da diese Programme extrem zuverlässig arbeiten. Es sei erwähnt, daß es auch andere gleichwertige, meist aber kommerzielle Bibliotheken gibt, etwa die NAG–Library [3], die IMSL–Library [4] oder von IBM die ESSL–Routinen [5].
Im folgenden werden daher für die wesentlichen Schritte bei der Aufstellung und Lösung der Normalgleichungssysteme Implementationen auf Basis dieser Bibliotheken vorgeschlagen. Basierend hierauf werden danach auch Möglichkeiten für parallele Realisierungen dieser Verfahren diskutiert.
Alle Programme wurden an einem aktuellen regularisierten Normalgleichungssystem des GFZ der Größe $n = 3754$ getestet. Der Regularisierungsparameter wurde hierbei $\alpha = 100$ gesetzt, dieser wird in Abschnitt 3 ausführlich diskutiert.

2.1 Matrix–Block–Reduktion

Die Beobachtungen während einer Schwerefeldmission ergeben ein Normalgleichungssystem, daß neben den Schwerefeldparametern als Unbekannten auch eine größere Zahl Bahnparameter enthält, die vorab eliminiert werden können. Bei geeigneter Numerierung der Unbekannten kann diese Situation wie folgt beschrieben werden.

Es liegt vor eine symmetrische positiv definite Matrix

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} ,$$

ein entsprechend aufgeteilter Vektor

$$b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} ,$$

sowie das Skalarprodukt $l^T l$. Dieses Skalarprodukt ist später wichtig, um Abschätzungen über das Residuum zu ermöglichen. Da A symmetrisch ist, gilt $A_{12} = A_{21}^T$. Der Rang von A sei n, der von A_{11} m. Hieraus sind nun die folgenden Größen zu berechnen:

$$\begin{aligned} \tilde{A}_{22} &= A_{22} - A_{21} A_{11}^{-1} A_{12} \ , \\ \tilde{b}_2 &= b_2 - A_{21} A_{11}^{-1} b_1 \ , \\ \tilde{l}^T \tilde{l} &= l^T l - b_1^T A_{11}^{-1} b_1 \ . \end{aligned}$$

Zunächst sei angenommen, A liege als untere Dreiecksmatrix vor. Damit hat man im ersten Schritt die Cholesky–Zerlegung $A_{11} = L\,L^T$ zu berechnen, L ist dann ebenfalls eine untere $(m \times m)$–Dreiecksmatrix. Hierfür dient die LAPACK-Routine `dpotrf`. Nun gilt:

$$\begin{aligned} \tilde{A}_{22} &= A_{22} - A_{21}(L\,L^T)^{-1} A_{12} \\ &= A_{22} - A_{21} L^{-T} L^{-1} A_{21}^T \\ &= A_{22} - A_{21} L^{-T} (A_{21} L^{-T})^T \end{aligned}$$

Damit wird ersichtlich, daß im nächsten Schritt die $(n - m \times m)$–Matrix $A_{21}L^{-T}$ benötigt wird. Diese läßt sich effizient mit der LAPACK–Routine `dtrsm` berechnen, die auch die Dreiecksgestalt von L berücksichtigt.
Die anschließend nowendige Berechnug von $A_{22} - A_{21}L^{-T}(A_{21}L^{-T})^T$ läßt sich als Rang–k Update von A_{22} interpretieren, hierbei ist k der Spaltenrang von $A_{21}L^{-T}$, also dem Rang von A_{11}, d.h. gleich m. Für solche Updates steht nun die LAPACK–Routine `dsyrk` zu Verfügung. Damit ist die Berechnung von $\tilde{A}_{22}$ vollständig.
Die Berechnung von $\tilde{b}_2$ läßt sich folgendermaßen formulieren:

$$\begin{aligned} \tilde{b}_2 &= b_2 - A_{21} A_{11}^{-1} b_1 \\ &= b_2 - (A_{21} L^{-T})\, L^{-1} b_1 \ . \end{aligned}$$

Demzufolge muß man erst $L^{-1}b_1$ berechnen, hierzu eignet sich die LAPACK–Routine `dtrsv`, danach erhält man durch eine Matrix–Vektor Multiplikation mit der bereits vorliegenden Matrix $A_{21}L^{-T}$ dann $\tilde{b}_2$.

Das Skalarprodukt $\tilde{l}^T\tilde{l}$ schließlich ergibt sich sehr einfach aus

$$\begin{aligned}\tilde{l}^T\tilde{l} &= l^T l - b_1^T A_{11}^{-1} b_1 \\ &= l^T l - b_1 L^{-T} L^{-1} b_1 \\ &= l^T l - (L^{-1}b_1)^T (L^{-1}b_1) \ ,\end{aligned}$$

d.h. es ist lediglich das Skalarprodukt von $L^{-1}b_1$ mit sich selber zu berechnen, dieser Vektor wurde aber bereits bei der Berechnung von $\tilde{b}_2$ erhalten.
Die hier skizzierte Vorgehensweise ist im Programm `redul.f` realisiert. Wegen $m \ll n$ wird die meiste Rechenzeit des Verfahrens während der Berechnung des Matrix–Updates benötigt werden, da nur hier auf der „großen“ Matrix A_{22} operiert wird. Bei der Bildung eines Matrixproduktes der Gestalt $A\,A^T$ wird aber nur auf den Zeilen der Matrix A gearbeitet, dies ist der Performance vermutlich abträglich.

	orig. Bibl.	tuned BLAS3	Perf.–Lib.	ATLAS
Cholesky dec.	0.10	0.11	0.09	0.05
$A_{21}L^{-T}$	6.46	3.30	2.35	1.05
$\tilde{A}_{22}$	61.51	24.54	16.12	8.47
$L^{-1}b_1$	0.00	0.00	0.00	0.00
$\tilde{b}_2$ und $\tilde{l}^T\tilde{l}$	0.05	0.08	0.06	0.05

Tab. 1 : Rechenzeiten der einzelnen Schritte des Programms `redul.f`

Das Programm wurde auf einer Workstation vom Typ SUN Ultra–2 getestet. Um die möglichen Performance–Gewinne durch die Benutzung maschinennah optimierter Kerne zu demonstrieren, wurden die Rechnungen einerseits mit Hilfe der selbst übersetzten BLAS– und LAPACK–Codes ausgeführt. Andererseits wurde die sogenannte tuned–BLAS Library [6] eingestzt, die frei verfügbar ist und für RISC–Prozessoren optimierte BLAS3–Routinen (also Matrix–Matrix Operationen) enthält. Weiterhin wurden die Rechnungen auf Basis der SUN–Performance Library [7] durchgeführt, die die vom Hardware–Hersteller optimierte BLAS– und LAPACK–Bibliothek enthält. Ähnliche Programmbibliotheken gibt es auch für Hardware anderer Hersteller. Die neueste Entwicklung in Richtung maschinen–optimierter Kerne für die lineare Algebra ist das ATLAS–System (Automatically Tuned Linear Algebra Software) von Whaley und Dongarra [8]. Hierbei werden während der Installation der Programme alle maschinenspezifischen Größen vermessen, so daß eine optimale Anpassung des Codes an die Eigenschaften des Prozessors erfolgt. Für die Verwendung der verschiedenen Bibliotheken sind an den Programmen in der Regel keinerlei Anpassungen notwendig.
In Tab. 1 sind die Rechenzeiten (in Sekunden) der wesentlichen Schritte des Programms `redul.f` dargestellt, hierbei wurde willkürlich $m = 375$ gesetzt.

Das Update von A_{22} benötigt tatsächlich die meiste Rechenzeit. Offensichtlich wird allein durch die Verwendung der tuned–BLAS3 Bibliothek die Berechnung des Matrix–Updates um den Faktor 5 beschleunigt, eine weitere Steigerung der Performance etwa um ein Drittel ergibt die Verwendung der SUN–Perfomance Library, die des ATLAS–Systems reduziert die Rechenzeit noch einmal um die Hälfte.
Um die Bildung eines Matrixprodukts der Form $A\,A^T$ zu vermeiden, geht man besser von einer Darstellung der Matrix als obere Dreiecksmatrix aus. Wiederum wird zuerst die Cholesky–Zerlegung von A_{11} berechnet, diesmal in der Form $A_{11} = U^T U$, U ist nun entsprechend auch obere Dreiecksmatrix. Damit lassen sich nun die benötigten Größen wie folgt ableiten:

$$\begin{aligned}\tilde{A}_{22} &= A_{22} - (U^{-T} A_{12})^T\, U^{-T} A_{12}\ ,\\ \tilde{b}_2 &= b_2 - (U^{-T} A_{12})^T\, U^{-T} b_1\ ,\\ \tilde{l}^T \tilde{l} &= l^T l - (U^{-T} b_1)^T\, (U^{-T} b_1)\ .\end{aligned}$$

Offensichtlich ist das Matrix–Update jetzt von der Form $A^T A$, somit wird dabei nur auf den Spalten von A (bzw. von $U^{-T} A_{12}$) gearbeitet. Die Reihenfolge der Operationen sowie die verwendeten LAPACK–Routinen bleiben bei dieser Implementation unverändert, dies ist im Programm `reduu.f` realisiert. Die erzielten Ergebnisse finden sich in Tab. 2.

	orig. Bibl.	tuned BLAS3	Perf.–Lib.	ATLAS
Cholesky dec.	0.12	0.10	0.08	0.05
$A_{21} L^{-T}$	7.53	3.16	3.36	1.01
$\tilde{A}_{22}$	65.26	25.64	17.00	8.13
$L^{-1} b_1$	0.00	0.00	0.00	0.00
$\tilde{b}_2$ und $\tilde{l}^T \tilde{l}$	0.05	0.06	0.06	0.04

Tab. 2 : Rechenzeiten der einzelnen Schritte des Programms `reduu.f`

Die Rechenzeiten sind, entgegen den Erwartungen, im wesentlichen unabhängig davon, welche Darstellung der Matrix gewählt wird. Der Grund hierfür liegt in den bereits vom Compiler vorgenommenen Optimierungen. Lediglich bei Verwendung von LAPACK–Unterprogrammen, die ohne Optimierung übersetzt wurden, ergeben sich sehr deutliche Unterschiede in den Rechenzeiten.
Abschließend sei angemerkt, daß die vorgeschlagene Methodik keine zusätzlichen Hilfsfelder benötigt, d.h. alle Rechnungen werden auf der Matrix selber vorgenommen. Eine weitere Effizienzsteigerung auf sequentiellen Rechnern ist vermutlich schwierig.

2.2 Lösung der Normalgleichungssysteme

Die Lösung der regularisierten Normalgleichungssysteme ist der entscheidende und zeitaufwendigste Schritt bei der Berechnung der Schwerefeldparameter. Daher soll im folgenden hierauf besonders eingegangen werden.
Die klassische Methode zur direkten Lösung einer voll–besetzten symmetrisch positiv definiten Matrix der Form A^TA, A $(m \times n)$–Matrix A, $m > n$, ist die Cholesky–Zerlegung. Innerhalb der LAPACK–Bibliothek steht hierfür die Routine `dpotrf` zur Verfügung. Diese liefert entweder eine untere Dreiecksmatrix L für die gilt $A^TA = L\,L^T$, oder eine obere Dreiecksmatrix U mit $A^TA = U^TU$. Da die Berechnung von L generell effizienter ist, da dann mehr Spaltenoperationen durchgeführt werden können, soll nur hier diese Variante berücksichtigt werden. Nach der Faktorisierung liefert die Routine `dpotrs` dann die Lösung des Normalgleichungssystems.
Geht man von dem linearen statistischen Modell $Ax = b + \epsilon$ und für ϵ von einer Varianz σ aus, also nur von Fehlern in den Beobachtungen b, so erhält man für die Korrelationsmatrix der Komponenten der Lösung x

$$V_x = \sigma^2\,(A^TA)^{-1} = \sigma^2\,L^{-T}L^{-1}\,,$$

σ^2 kann hierbei durch $\|b - Ax\|_2^2/(m-n)$ approximiert werden. Somit muß für die Berechnung der Kovarianzmatrix die Dreiecksmatrix L invertiert werden. Dies wird durch der LAPACK–Routine `dtrtri` ermöglicht. Daraus sind die Standardabweichungen für x_i vermittels der Identität $(L^{-T}L^{-1})_{ii} = \sum_{k=i}^{n}(L^{-1})_{ki}$ erhältlich. Für die Berechnung der vollständigen Kovarianzmatrix ist dagegen die explizite Matrix–Multiplikation $L^{-T}L^{-1}$ notwendig, dies kann mit dem LAPACK–Unterprogramm `dlauum` durchgeführt werden.

	orig. Bibl.	tuned BLAS3	Perf.–Lib.	ATLAS
`dpotrf`	320.04	106.41	65.64	41.63
`dpotrs`	0.53	1.03	1.77	0.78
`dtrtri`	318.83	108.46	78.01	39.63
`dlauum`	98.80	101.72	64.50	39.37

Tab. 3 : Rechenzeiten der einzelnen Schritte des Programms `chol.f`

Damit sind alle notwendigen Schritte als Aufrufe geeigneter Routinen realisiert, Tab. 3 enthält die jeweiligen Rechenzeiten des entsprechenden Programms `chol.f` bei Verwendung der verschiedenen BLAS– bzw. LAPACK–Bibliothek. Offensichtlich ist hier auch wieder die wesentliche Verringerung der Rechenzeit bei Verwendung optimierter Bibliotheken.
Zur Erhöhung der Genauigkeit der Lösung eines linearen Gleichungssystems als auch zur Abschätzung der bereits erreichten Genauigkeit kann eine iterative Verfeinerung herangezogen werden. Hierbei werden aus einer Nähe-

rungslösung x_0 und dem folgenden Schema neue, unter gewissen Voraussetzungen bessere Lösungen x_k berechnet:

$$\begin{aligned} &\text{für } k = 1, 2, \ldots : \\ &\qquad r_k = b - Ax_k \\ &\qquad \delta_k = A^{-1} r_k \\ &\qquad x_{k+1} = x_k + \delta_k \;. \end{aligned}$$

Diese Methodik wird im Detail gerade in Hinsicht auf Normalgleichungssysteme in Björck [9] diskutiert. Realisiert ist sie in der LAPACK–Routine `dposvx`, die neben einer möglicherweise genaueren Lösung x die folgenden Größen und Fehlerschranken liefert:

- die Norm der Matrix A,
- eine Abschätzung der Konditionszahl der Matrix A in der 1–Norm,
- eine Abschätzung für den sog. Vorwärtsfehler der Lösung x, d.h. eine obere Schranke für $(x_i - \tilde{x}_i)/x_i$, wobei $\tilde{x}$ die exakte Lösung darstellt.
- eine Abschätzung des sog. Rückwärtsfehlers der Lösung x, d.h. die kleinste relative Änderung in der Matrix A, die x zur exakten Lösung machen würde.

Eine weitere Möglichkeit ist die Skalierung der Matrix A. Dies bedeutet, daß statt dem Gleichungssystem $Ax = b$ das System $DAD\,D^{-1}x = Db$ gelöst wird. Dabei ist D eine i.A. diagonale Skalierungsmatrix. Durch die Skalierung soll erreicht werden, daß die Matrixelemente eine in etwa gleiche Größenordnung haben. Eine sinnvolle Wahl für D ist in vielen Fällen die Diagonale von A selber.

	ohne Skalierung	mit Skalierung
Matrixnorm	$1.68 \cdot 10^{24}$	$9.28 \cdot 10^{1}$
Schätzung f. Konditionszahl	$2.75 \cdot 10^{7}$	$2.27 \cdot 10^{7}$
Vorwärtsfehler	$8.34 \cdot 10^{-6}$	$3.42 \cdot 10^{-5}$
Rückwärtsfehler	$7.40 \cdot 10^{-14}$	$3.15 \cdot 10^{-14}$
Zahl der Iterationen	3	4
Reduktion der Norm von des Residuums bis	$1.66 \cdot 10^{0}$ $6.36 \cdot 10^{-1}$	$1.47 \cdot 10^{-9}$ $5.01 \cdot 10^{-10}$

Tab. 4 : Ergebnisse von iterativer Verfeinerung und Skalierung

Beide Optionen wurden in das Programm `chol.f` übernommen, wahlweise kann dort ein Unterprogramm `refine` aufgerufen werden, das eine iterative

Verfeinerung durchführt bzw. eine Skalierung des Gleichungssystems vorgenommen werden. Für das hier behandelte Normalgleichungssystem erhält man damit die Ergebnisse in Tab. 4.
Obwohl nun die Norm des Residuums in beiden Fällen durch die iterative Verfeinerung etwa um einen Faktor 2.5 verringert wird, ist der Einfluß auf die Komponenten der Lösung im Vergleich zur abgeschätzten Standardabweichung viel zu klein, um eine Anwendung von Skalierung oder iterativer Verfeinerung zu rechtfertigen. Während die Standardabweichungen nämlich zwischen 10^{-11} und 10^{-9} liegen, bleibt die maximale Änderung einer Komponente des Lösungsvektors durch Anwendung der iterativen Skalierung unter $2 \cdot 10^{-19}$. Der Einfluß der Skalierung ist noch geringer. Selbst wenn die Schätzung der Standardabweichungen unrealistisch sein sollte, so ist der Einfluß des Regularisierungsparameters α um Größenordnungen stärker, als die der Verfeinerung. Dies werden die Ausführungen im Kap. 3 belegen. Aus diesem Grund finden beide Methoden im folgenden keine Anwendung mehr.

2.3 Parallele Lösung der Normalgleichungssysteme

Die Verwendung optimierter Bibliotheken beschleunig die Lösung der Systeme bei der Schwerefeldinversion bereits erheblich. Da aber bei den zukünftigen Geopotentialmissionen die Zahl der Schwerefeldparameter sehr viel größer sein wird, ist es wünschenswert, noch schnellere Methoden zu Verfügung zu haben. Aus diesem Grund wurde auch eine parallele Implementation der Lösung der Normalgleichungssysteme auf der Basis des Cholesky–Verfahrens entwickelt.
Entsprechend den Bibliotheken BLAS und LAPACK existieren auch für parallele Anwendungen Sammlungen von Unterprogrammen für Anwendungen der linearen Algebra. Diese sind jedoch weit weniger verbreitet als die sequentiellen Implementation und ihre Benutzung ist zum Teil außerst umständlich. Im wesentlichen ist dies zum einen die ScaLAPACK–Bibliothek [10], die eine parallele Weiterentwicklung von LAPACK darstellt. Frei verfügbar ist weiterhin nur die PLAPACK–Bibliothek [11]. Diese stellt für den Benutzer Schnittstellen auf einer sehr hohen Abstraktionsebene zur Verfügung und verbirgt dadurch Details der Datenverteilung und Kommunikation. Daher sind Standardaufgaben der linearen Algebra mit dieser Bibliothek sehr viel einfacher zu implementieren. Da außerdem für die hier interessierende Cholesky–Zerlegung Implementationen auf Basis von PLAPACK–Routinen die besten Rechenzeitergebnisse erzielten, wurde PLAPACK als Grundlage für eine parallele Version des Programms `chol.f` verwendet. Es sei angemerkt, daß die parallele Variante des Cholesky–Verfahrens innerhalb PLAPACK gerade auch in Hinsicht auf Anwendungen der Satellitengeodäsie entwickelt wurde, s. etwa Baker [12]. PLAPACK benutzt für die Kommunikation die Message–Passing Bibliothek MPI [13] und ist dadurch auf praktisch allen Parallelrechnern einsetzbar.

Die Implementierung der parallelen Programme wurde auf der CRAY T3E des Konrad–Zuse-Zentrums vorgenommen. Diese hat 384 Prozessoren, davon 48 Prozessoren mit einer Taktrate von 600 MHz und 512 MB lokalem Speicher, 88 Prozesoren mit 600 MHz und 128 MB sowie 264 Prozessoren mit 450 MHz und 128 MB. Abb. 1 enthält die die Rechenzeiten nur auf den 600 MHz–Prozessoren. Da der Speicher eines einzigen Prozessors zu klein ist, können keine Rechenzeiten auf einem Prozessor angegeben werden.

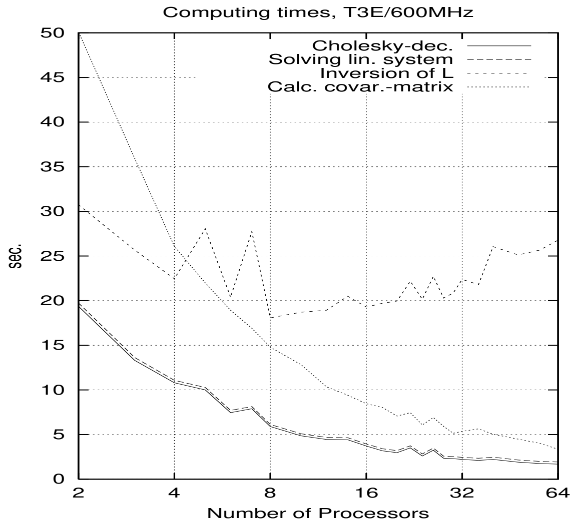


Abbildung 1: Rechenzeiten des Programms `parchol`

Bis auf die Inversion von L skalieren alle Programmteile offensichtlich zufriedenstellend. Die Berechnung von L^{-1} in der gegenwärtigen Implementierung hat noch einen hohen Kommunikations–Aufwand, dieser kann aber ggf. durch eine detaillierte Analyse noch erheblich gesenkt werden. Damit ist nun die Möglichkeit gegeben, den Datendurchsatz gegebenenfalls durch den Einsatz der parallelen Versionen beträchtlich zu erhöhen.

2.4 Eigenwertberechnung

Eine Eigenwert– und Eigenvektorbestimmung kann problemlos mit Hilfe der LAPACK–Routine `dsyev` vorgenommen werden, die speziell für symmetrische Matrizen geeignet ist. Dies ist im Programm `eigen.f` realisiert. In Tab. 5 sind die Rechenzeiten wieder unter Verwendung verschieden optimierter Bibliotheken aufgeführt. Werden nur die Eigenwerte berechnet, so hat es nur geringen Einfluß, ob die Matrix als obere oder untere Dreiecksmatrix übergeben wird. Für die Berechnung der Eigenvektoren ist aber offensichtlich die Darstellung als untere Dreiecksmatrix erheblich günstiger. Etwa überraschend ist die Tatsache, daß die normalen LAPACK– und BLAS–Bibliotheken schneller sind als die tuned BLAS3–Bibliothek und diese wiederum deutlich schneller als die SUN Perfomance Library ist.

	orig. Bibl.	tuned BLAS3	Perf.–Lib.	ATLAS
nur EW, obere M.	538.43	779.56	1116.46	653.90
nur EW, untere M.	534.39	773.77	1128.95	613.29
EW und EV, obere M.	3883.64	4031.43	5247.67	2254.67
EW und EV, untere M.	2697.88	2861.99	3461.50	2280.34

Tab. 5 : Rechenzeit für die Eigenwertbestimmung mit `dsyev`

Für die Eigenwert– und Eigenvektorberechnung steht in der neuen LAPACK Version 3.0 zusätzlich die Routine `dsyevd` zur Verfügung, diese arbeitet nach dem sog. divide and conquer Prinzip. Hiermit erhält man die folgenden Resultate.

	orig. Bibl.	tuned BLAS3	Perf.–Lib.	ATLAS
nur EW, obere M.	582.03	625.36	1253.34	496.18
nur EW, untere M.	568.80	632.18	1168.35	496.84
EW und EV, obere M.	1880.17	1367.62	1569.08	819.93
EW und EV, untere M.	2141.51	1346.52	1637.39	816.29

Tab. 6 : Rechenzeit für die Eigenwertbestimmung mit `dsyevd`

Diese Routine ist zwar um einiges schneller als `dsyev`, die Ergebnisse sind aber etwas ungenauer. Aus der Matrix V der Eigenvektoren und den Eigenwerten Λ kann gemäß $A = V \Lambda V^T$ nämlich A zurückberechnet werden. Tab. 7 zeigt (für die skalierte Matrix $L^{-T} A^T P A L^{-1}$, L ist die Kaula–Matrix) typische Ergebnisse dieser Rückberechnung. Offensichtlich ist diese, wohl auch infolge der schlechten Kondition der Matrix bzw. der Existenz von Eigenwerten über einen weiten Bereich von Zehnerpotenzen, nicht fehlerfrei möglich. Insbesondere muß hervorgehoben werden, daß die Berechnung gerade der kleinsten Eigenwerte recht ungenau ist, schon die Verwendung verschiedener Compiler–Optionen ergibt hierbei zum Teil signifikante Abweichungen. Weiter sei darauf hingewiesen, daß diese Resultate zum Teil auch von der verwendeten Hardware abhängig sind.

	Δ_1	Δ_2	Δ_3	Δ_4
`dsyev`	0.072	65.97	0.0087	40.27
`dsyevd`	60.77	$3.40 \cdot 10^5$	0.0234	$1.20 \cdot 10^5$

Tab. 7 : Δ–Werte für `dsyev` und `dsyevd`

Hierbei bedeuten:

$$\begin{aligned}
\Delta_1 &= \sum_{i,j=1}^{n} |a_{ij} - (V\Lambda V^T)_{ij}| \\
\Delta_2 &= \sum_{i,j=1}^{n} |(a_{ij} - (V\Lambda V^T)_{ij})/a_{ij}| \\
\Delta_3 &= \max |a_{ij} - (V\Lambda V^T)_{ij}| \\
\Delta_4 &= \max |(a_{ij} - (V\Lambda V^T)_{ij})/a_{ij}| \;.
\end{aligned}$$

Für die nicht regularisierte Matrix ($\alpha = 0$) ergeben sich mit `dsyev` numerisch die folgenden 4 Eigenwerte < 0:

$$\begin{array}{l}
-\,4.23 \cdot 10^{15} \\
-\,2.24 \cdot 10^{15} \\
-\,2.00 \cdot 10^{14} \\
-\,9.51 \cdot 10^{13},
\end{array}$$

alle anderen liegen zwischen 10^{11} und 10^{24}. Da diese Matrix ein Normalgleichungssystem sein soll, ist die Existenz negativer Eigenwerte bereits ein deutliches Indiz dafür, daß eine Regularisierung unumgänglich ist. In Abb. 2 ist die Verteilung der Eigenwerte über einer logarithmischen Skala (also ohne die 4 negativen Eigenwerte) aufgetragen.
Für die mit $\alpha = 100$ regularisierte Matrix erhält man keine negativen Eigenwerte mehr, sie liegen zwischen 10^{17} und 10^{24}. In Abb. 2 ist diese Verteilung, ebenfalls logarithmisch, aufgetragen.

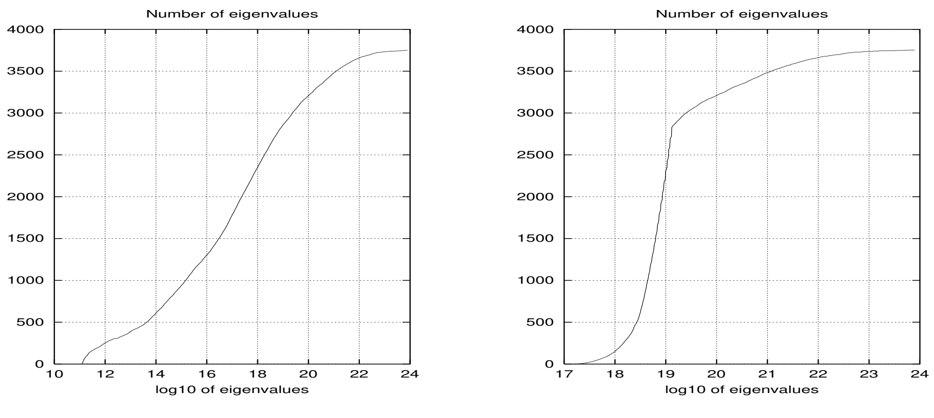


Abbildung 2: Eigenwertverteilungen für $\alpha = 0$ und $\alpha = 100$

3 Untersuchung und Vergleich von Methoden zur Regularisierung schlecht konditionierter Normalgleichungssysteme

3.1 Grundlagen der Regularisierung der Schwerefeldinversion

In diesem Abschnitt sollen die spezifischen Voraussetzungen für die Regularisierung der Systeme bei der Schwerefeldinversion untersucht werden. Anschließend werden verschiedene Verfahren zur Wahl eines optimalen Regularisierungsparameters getestet und kritisch verglichen. Im folgenden soll aber keine detaillierte Einführung in die Theorie und Anwendung der Regularisierung gegeben werden, hierfür sei auf die Literatur, etwa Hanke und Hansen [14], Hansen [15, 16], Engl, Hanke und Neubauer [17] verwiesen.
Es sei

$$Ax = b, \quad A \in \mathbb{R}^{m\times n}, \quad x \in \mathbb{R}^n, \quad b \in \mathbb{R}^m, \quad m > n \; , \tag{1}$$

ein lineares Ausgleichssystem mit einer schlecht konditionierten Matrix A. Sowohl A als auch die rechte Seite b seien fehlerbehaftet. Um eine Näherungslösung eines solchen Systems zu erhalten, muß es durch ein weniger schlecht konditioniertes System ersetzt, d.h. regularisiert werden. Die Regularisierung besteht in der Einführung eines Operators L und der Ersetzung des linearen Systems durch das Minimierungsproblem

$$\min_{x\in\mathbb{R}^n} \{\|Ax - b\|^2 + \alpha\|Lx\|^2\} \; . \tag{2}$$

Hierbei ist α der sog. Regularisierungsparameter, $\|\cdot\|$ bedeutet die Euklidische Norm. Der Operator L soll sicherstellen, daß die Lösung innerhalb eines Unterraums gesucht wird, innerhalb dem eine Näherungslösung akzeptiert werden kann, gilt $L = I$, so liegt ein Regularisierungsproblem in Standardform vor. Für die Lösung x_α von (2) gilt

$$(A^T A - \alpha L^T L)x_\alpha = A^T b \; , \tag{3}$$

dies ist die übliche Formulierung der Tichonov–Regularisierung. Das Minimierungsproblem (2) kann äquivalent formuliert werden als

$$\min_{x\in\mathbb{R}^n} \left\| \begin{pmatrix} A \\ \sqrt{\alpha}L \end{pmatrix} x - \begin{pmatrix} b \\ 0 \end{pmatrix} \right\| \; , \tag{4}$$

d.h. x_α ist die beste Lösung des überbestimmten Gleichungssystems

$$\begin{pmatrix} A \\ \sqrt{\alpha}L \end{pmatrix} x = \begin{pmatrix} b \\ 0 \end{pmatrix} \; . \tag{5}$$

Aus diesem Grund nennt man die in das ursprüngliche lineare System (1) zusätzlich eingeführten Gleichungen $\sqrt{\alpha}Lx = 0$ auch Pseudobeobachtungen.

Im Kontext der Schwerefeldbestimmung ergeben sich nun einige Besonderheiten. Zum einen führt man eine Diagonalmatrix P ein, die die unterschiedliche Gewichtung der Datensätze berücksichtigt. Da man a priori weiß, daß sich die Varianz der Koeffizienten, in Abhängigkeit vom Grad l der Koeffizienten etwa wie $7 \cdot 10^{-6}/l^2$, z.B. Reigber [18], verhält, kann man dies dazu verwenden, einen entsprechenden Operator L zu definieren, L ist ebenfalls eine Diagonalmatrix (Kaula–Matrix). Weiterhin ist die Bestimmung der Schwerefeldparameter eigentlich ein nichtlineares Problem, d.h. $F(x) = b$. Eine Linearisierung ergibt

$$b = F(x_0) + \frac{\partial F}{\partial x} dx = b_0 + A\, dx \ . \tag{6}$$

mit $dx = x - x_0$, d.h. es wird eine Näherungslösung des Gleichungssystems $l := b - b_0 = A\, dx$, bzw. unter Berücksichtigung von P, von

$$\sqrt{P} A\, dx = \sqrt{P} l \tag{7}$$

gesucht. x_0 ist hierbei ein Standard–Schwerefeldmodell niederer Ordnung. Damit ergibt sich als Minimierungsaufgabe

$$\min_{x \in \mathbb{R}^n} \{ \|l - A\, dx\|_P^2 + \alpha \|x\|_L^2 \} \ . \tag{8}$$

Hierzu äquivalent sind die der Formulierung (3) entsprechenden Gleichungen

$$\begin{aligned} (A^T P A + \alpha L^T L) x_\alpha &= A^T P (l + A x_0) \ , \\ (A^T P A + \alpha L^T L)\, dx_\alpha &= A^T P l - \alpha L^T L x_0 \ . \end{aligned} \tag{9}$$

Anstelle (4) und (5) erhält man jetzt:

$$\min_{x \in \mathbb{R}^n} \left\| \begin{pmatrix} \sqrt{P} A \\ \sqrt{\alpha} L \end{pmatrix} dx - \begin{pmatrix} \sqrt{P} l \\ -\sqrt{\alpha} L x_0 \end{pmatrix} \right\| \ ,$$

$$\begin{pmatrix} \sqrt{P} A \\ \sqrt{\alpha} L \end{pmatrix} dx = \begin{pmatrix} \sqrt{P} l \\ -\sqrt{\alpha} L x_0 \end{pmatrix} \ .$$

Die Norm des Residuums der letzten Gleichung ist gegeben durch

$$\|A\, dx_\alpha - l\|_P^2 + \alpha \|x_\alpha\|_L^2 \ ,$$

also gleich der Zielfunktion der Minimierungsaufgabe (8). Demgegenüber gilt für das Residuum von (7)

$$\|r_\alpha\|_P^2 := \|l - A\, dx_\alpha\|_P^2 = l^T P l - dx_\alpha^T A^T P l - \alpha (L dx_\alpha)^T L (dx_\alpha + x_0) \ .$$

Im folgenden wird unter Residuum immer diese Größe verstanden.

Wie bereits angemerkt, ist $A^T PA$ ebenfalls fehlerbehaftet und die Eigenwertberechnung ergibt 4 negative Eigenwerte. Eine Voraussetzung zur Lösbarkeit des Normalgleichungssystems ist aber eine positiv–definite Matrix $A^T PA + \alpha L^T L$, daraus läßt sich also ein Minimalwert für α ableiten. Hierfür transformiert man das Regularisierungproblem (9) vermittels der Definition $\tilde{A} := AL^{-1}$ in Standardform:

$$(\tilde{A}^T P\tilde{A} + \alpha)L\, dx_\alpha = \tilde{A}^T Pl - \alpha L x_0 \; .$$

Für $\tilde{A}^T P\tilde{A}$ erhält man als kleinsten Eigenwert $-5.38 \cdot 10^{-2}$. Demzufolge muß etwa $\alpha \geq 0.054$ gelten, um das System überhaupt mit Hife einer Cholesky–Zerlegung lösen zu können. Es wird sich sich aber zeigen, daß mit einem solchen α keine sinnvolle Regularisierung erreicht wird.
Die hier skizzierte Methode der Tichonov–Regularisierung ist sicherlich das am häufigsten verwendete Verfahren zur Lösung schlecht konditionierter Normalgleichungssystem. Eine mathematische Analyse ist auf der Grundlage einer SVD–Zerlegung der Matrix A möglich, auf diese wird im Abschnitt 3.9 näher eingegangen. Basierend auf der SVD lassen sich nun die verschiedensten Regularisierungmethoden definieren, z.B. Truncated SVD oder GSVD, Damped SVD oder GSVD, Hansen [15, 16], Bouman [19], exponential filtering, Calvetti, Lewis und Reichel [20]. Diese Methoden sind aber nur dann anwendbar, wenn eine SVD–Zerlegung von A oder eine Eigenwert–Zerlegung von $A^T A$ zugänglich ist. Eine weitere Klasse von Regularisierungsverfahren basiert auf der Tatsache, daß A und b oft beide fehlerbehaftet sind. Als Minimierungsaufgabe ergibt sich als Konsequenz daher die sog. „total least squares (TLS)“–Formulierung:

$$\min_{\tilde{A},\tilde{b},x} \|(A, b) - (\tilde{A}, \tilde{b})\|_F \quad \text{mit: } \tilde{b} = \tilde{A}x \; ,$$

Golub, Hansen und O'Leary [21]. Es kann aber gezeigt werden, daß für Probleme in Standardform oder solche, die in diese transformiert werden können, TLS und Tichonov–Regularisierung im wesentlichen äquivalent sind. Eine weitere Klasse von Regularisierungs–Methoden sind iterative Verfahren. Diese basieren auf bekannten Methoden zur Lösung von Gleichungssystemen auf der Grundlage von Krylov–Verfahren, s. Hansen [15] für eine Übersicht. Dies ist zum einen das stationäre Landweber–Verfahren und die daraus abgeleitete sog. ν–Methode, zum anderen das CG–Verfahren, insbesondere in der robusten LSQR–Implementation. Besonders attraktiv an diesen Verfahren scheint auf den ersten Blick, daß sie keinen Regularisierungsparameter benötigen, sondern die inhärente Regularisierungseigenschaften des iterativen Alorithmus nutzen. Dieser Vorteil wird aber aufgewogen dadurch, daß kein eindeutiges Abbruchkriterium zu Verfügung steht. Tatsächlich übernimmt der Iterationsindex die Rolle des Regularisierungsparameters und eine Entscheidung für einen spezifischen Index kann nur anhand der gleichen Kriterien getroffen werden, die auch zur Wahl eines optimalen α zur Verfügung stehen. Weiterhin sind iterative Verfahren effizient und robust nur für dünn besetzte Verfahren einzusetzen, bei der Anwendung des CG–Verfahrens auf positiv–definite Systeme hängt die Konvergenz zudem entscheidend davon ab, daß niemals das Matrixprodukt A^TA benutzt wird, sondern immer nur A selber. Dies ist aber bei der Schwerfeldbestimmung nicht möglich. Eine Konvergenz wäre für ein CG–Verfahren daher nur zu erwarten, wenn die Kondition von A^TA durch ein geeignete Präkonditionierung entscheidend verbessert würde. Allerdings bedingt ein Präkonditionierer, etwa ein sog. sparse approximate inverse preconditioner (SPAI), Huckle [22], weitere Parameter, die in gewissen Bereichen frei gewählt werden können und daher eine noch geringere Eindeutigkeit der Lösung ergeben.

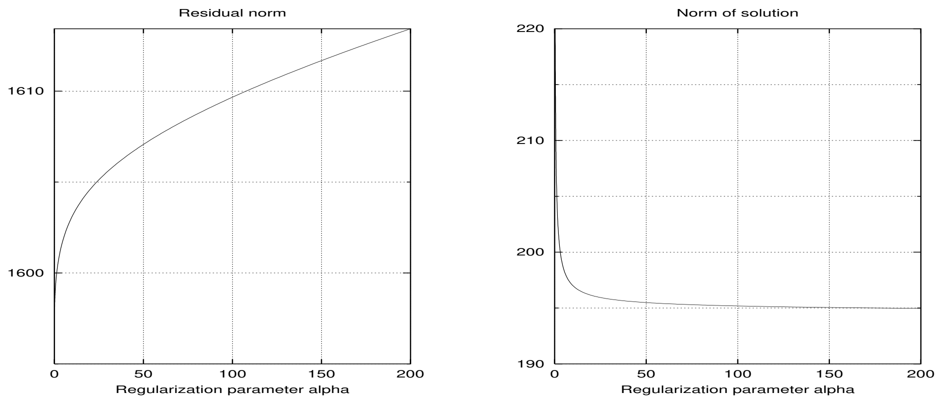


Abbildung 3: Norm des Residuums und der Lösung

Im Rahmen der Schwerefeldbestimmung bleibt daher der einzig praktikable Ansatz zur Regularisierung eine Tichonov–Regularisierung wie in (9). Akzeptiert man nun die spezielle Wahl der Matrizen P und L, so bleibt als freier Parameter nur noch α. Daher wird man nach einer objektiven und mathematisch fundierten Methode suchen, die ein „optimales“ α liefert. Im folgenden werden einige Methoden dargestellt, die unter Grundlegung verschiedener theoretischer Ansätze versuchen, ein solches α zu ermitteln. Alle Berechnungen wurden in einem in etwa logarithmisch verteilten Bereich von 770 α–Werten zwischen 0.054 und 10^6 vorgenommen. Dabei ist 0.054 in Übereinstimmung mit der Eigenwertanalyse tatsächlich der kleinste Wert, für den eine Cholesky–Zerlegung möglich ist. In Abb. 3 sind die Norm des Residuums und der Lösung, also $\|l - A\,dx_\alpha\|_P$ und $\|x_\alpha\|_L$, dargestellt.
Da mit x_0 bereits ein Schwerefeldmodell niederer Ordnung vorliegt, ist auch die Abweichung $x_\alpha - x_0$, d.h. $\|dx_\alpha\|$, von Interesse, diese Größe ist in Abb. 4 dargestellt. Diese Funktion hat ein Minimum bei $\alpha = 79$, hier ist also die Übereinstimmung zwischen x_0 und x_α am größten.
Es sei hervorgehoben, daß diese Untersuchungen nur möglich waren mit Hilfe

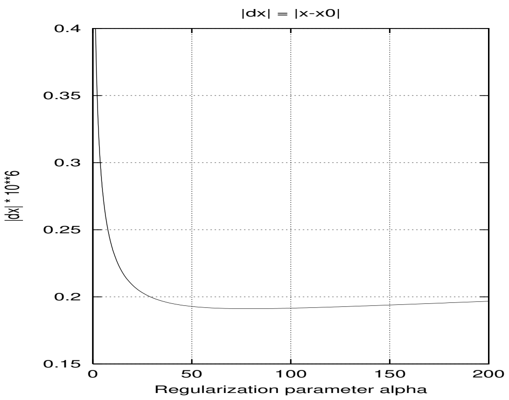


Abbildung 4: Norm von dx_α

der im Kapitel 2 dargestellten effektiven Algorithmen. Die einzelnen Methoden in den folgenden Abschnitten werden der Einfachheit halber an dem Minimierungsproblem (3) dargestellt.

3.2 L–Kurve Kriterium

Eine der am häufigsten verwendeten Methoden zur Bestimmung eines Regularisierungsparamters α ist das L–Kurve Kriterium, Hansen [15, 23], Hansen und O'Leary [24]. Die zugrunde liegende Idee ist die eines Ausgleich zwischen dem Fehler, der durch die Regularisierung eingeführt und dem, der durch die Fehler in den Daten selber hervorgerufen wird. Ist α zu groß, wird also zu stark regularisiert, so wird $\|Ax_\alpha - b\|$ zu groß sein. Ist andrerseits α zu klein, wird also nur schwach regularisiert, so wird die Lösung dominiert von Fehlern in den Daten und $\|x_\alpha\|$ zu groß sein, s. als Illustration hierzu auch Abb. 3. Demnach sollte zur Beurteilung der Lösung sowohl die Norm des Residuums als die der Lösung herangezogen werden. Bei der Methode der L–Kurve werden nun diese beiden Größen in doppelt–logarithmischer Skala aufgetragen. Auf Grundlage einer SVD–Zerlegung der Matrix A kann nun gezeigt werden, daß diese Darstellung die Form eines L besitzt und der Punkt mit der stärksten Krümmung der Kurve das optimale α repräsentiert. Die theoretischen Voraussetzungen hierfür sind aber die folgenden:

- Nur die rechte Seite b ist fehlerbehaftet, d.h. $b = \bar{b} + \epsilon$.
- Die Kovarianzmatrix des Störungsvektors ϵ ist ein Vielfaches der Einheitsmatrix.
- Es gilt: $\|\epsilon\| < \|\bar{b}\|$.
- Die SVD–Koeffizienten $|u_i^T \bar{b}|$ fallen rascher als die Singulärwerte σ_i (diskrete Picard–Bedingung).

Ungeachtet dieser restriktiven Voraussetzungen hat sich diese Methode auch in verschiedenen Anwendungsbereichen u.a. mit stark korrelierten Fehlern in den Komponenten der rechten Seite bewährt. Es sei hervorgehoben, daß im Fall der Schwerefeldinversion die ersten beiden Bedingungen nicht erfüllt sind, die anderen beiden können nicht überprüft werden.
Bei der Realisierung dieser Methode für die Schwerefeldbestimmung hat man auf die spezielle Form des Minimierungsproblems (8) zu achten. Aufzutragen ist demnach $\log \|l - A\,dx_\alpha\|$ gegen $\log \|x_\alpha\|$, s. Abb. 5.

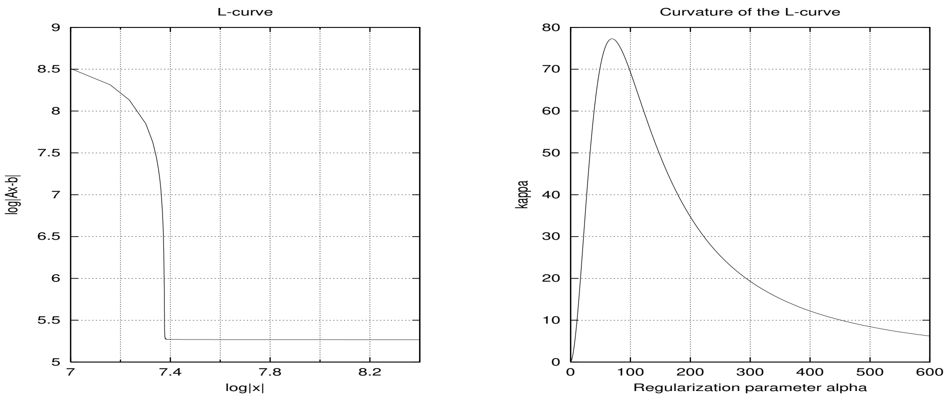


Abbildung 5: L–Kurve und deren Krümmung

Um aus dieser Darstellung nun das optimale α zu erhalten, muß die Krümmung dieser Kurve berechnet werden. Sei hierzu

$$\eta = \|x_\alpha\|^2 , \qquad \rho = \|b - Ax_\alpha\|^2 ,$$
$$\hat{\eta} = \log \eta , \qquad \hat{\rho} = \log \rho .$$

und $\hat{\rho}', \hat{\rho}'', \hat{\eta}', \hat{\eta}''$ die Ableitungen nach α. Dann ist die Krümmung gegeben durch

$$\kappa(\alpha) = 2 \frac{\hat{\rho}'\hat{\eta}'' - \hat{\rho}''\hat{\eta}'}{((\hat{\rho}')^2 + (\hat{\eta}')^2)^{3/2}} .$$

Hat man die Berechnungen für hinreichend viele α–Werte durchgeführt, so läßt sich durch Approximation der Ableitungen durch finite Differenzen der Ort der maximalen Krümmung bestimmen, s. Abb. 5.
Allerdings gibt es auch andere Möglichkeiten zur Bestimmung des Maximums von $\kappa(\alpha)$. Mit Hilfe der SVD–Zerlegung läßt sich nämlich die einfache Beziehung

$$\rho' = -\alpha\eta'$$

ableiten. Diese Relation ist zudem ein guter Test für die Konsistenz der numerisch ermittelten Werte für ρ und η. Im vorliegenden Fall ist sie mit einer

Genauigkeit von etwa 0.1 % erfüllt. Auch die Verwendung der 2. Ableitungen läßt sich vermeiden, man kann zeigen, daß

$$\kappa = 2\frac{\eta\rho}{\eta'}\frac{\alpha\eta\rho + 2\sqrt{\alpha}\eta\rho + \alpha^2\eta\eta'}{(\alpha\eta^2+\rho^2)^{3/2}}$$

gilt. Von Regińska [25] wurde weiterhin abgeleitet, daß das α für das $\kappa(\alpha)$ das Maximum annimmt, gleichzeitig das Minimum von $\|x_\alpha\| \cdot \|b - Ax_\alpha\|$ ist; allerdings kann diese Funktion auch ein Minimum haben, ohne daß die $(\|x_\alpha\|, \|b - Ax_\alpha\|)$–Darstellung die Form einer L–Kurve hat, s. etwa Floberghagen und Bouman [26].

Unabhängig von den Details der Methode erhält man im Fall der Schwerefeldinversion mit der Methode der L–Kurve ein optimales $\alpha = 69$. Bei der Beurteilung dieses Wertes ist aber Vorsicht geboten. Zum einen ist der Wertebereich, in dem sich $\|x_\alpha\|$ und $\|b - Ax_\alpha\|$ bewegen, recht klein. Bei allen in der Literatur, Hansen [23, 24, 27] angegebenen Testbeispielen und Anwendungen für die L–Kurve variieren beide Funktionen über mehrere Zehnerpotenzen. Tatsächlich entspricht die dem Augenschein nach wohldefinierte „Ecke“ der L–Kurve einem größerem Wertebereich für α. Innerhalb von $\alpha = 16$ und 420 ändern sich beide Normen nur um weniger als 0.1 %, innerhalb von 56 und 88 um weniger als 0.01 %.

Abschließend sei angemerkt, daß die Methode der L–Kurve verschiedentlich kritisiert wurde, s. insbesondere Hanke [28], Vogel [29]. Im wesentlichen wurde von diesen Autoren gezeigt, daß für verschiedene diskretisierte Integralgleichungen dieser Algorithmus weder für eine feiner werdende Diskretisierung noch für eine gegen 0 gehende Störung der rechten Seite das korrekte asymptotische Verhalten besitzt.

3.3 Generalized cross–validation

Die Methode der generalized cross–validation (GCV) basiert auf statistischen Verfahren. Sei eine beliebige Gleichung aus $Ax = b$ gestrichen. Das optimal regularisierte reduzierte System sollte dann am besten geeignet sein, das fehlende b_i zu rekonstruieren. Eine exakte Herleitung findet sich in Golub, Heath und Wahba [30], Wahba [31], s. auch Hanke und Hansen [14], Björck [9]. Als Funktional, dessen Minimum in Abhängigkeit von α gesucht ist, ergibt sich dabei:

$$G(\alpha) = \frac{\|b - Ax_\alpha\|^2}{(\mathrm{tr}\,[I_m - A(A^TA + \alpha L^TL)^{-1}A^T])^2} \quad .$$

Der Spurausdruck läßt sich vereinfachen vermittels

$$\mathrm{tr}\left[I_m - A(A^TA + \alpha L^TL)^{-1}A^T\right] = m - n + \alpha\,\mathrm{tr}\left[(A^TA + \alpha L^TL)^{-1}\right] \quad .$$

Dies ergibt speziell für die Schwerefeldinversion:

$$G(\alpha) = \frac{\|l - A\,dx_\alpha\|_P^2}{\left(m - n + \alpha\,\mathrm{tr}\,[(A^TPA + \alpha L^TL)^{-1}]\right)^2} \quad .$$

Die numerische Auswertung des Spuraudrucks im Nenner ist effizient auf folgende Weise möglich. Sei $\Lambda\Lambda^T$ die Cholesky–Zerlegung von $A^TPA+\alpha L^TL$. Dann gilt:

$$\begin{aligned}
\mathrm{tr}\left[(A^TPA + \alpha L^TL)^{-1}\right] &= \mathrm{tr}\left[(\Lambda\Lambda^T)^{-1}\right] \\
&= \sum_{i=1}^{n}(\Lambda^{-T}\Lambda^{-1})_{ii} \\
&= \sum_{i=1}^{n}\sum_{k=1}^{n}(\Lambda^{-1})_{ki}^2 \; .
\end{aligned}$$

Es muß also für jedes α die Dreiecksmatrix Λ invertiert werden.
Das Ergebnis dieser Berechnungen ist für die Schwerefeldinversion in Abb. 6 gezeigt. Offensichtlich weist die Kurve kein Minimum auf, außer für $\alpha \to 0$. Ähnliche Beobachtungen sind bereits von anderen Autoren berichtet worden. Danach scheint die GCV–Methode dann zu versagen, wenn die Fehler in der rechten Seit hoch korreliert sind. Dagegen kann die Methode der L–Kurve solche ausfiltern, auch wenn sie die diskrete Picard–Bedingung verletzen; für eine nähere Diskussion hierüber s. Hansen [23]. Das GCV–Verfahren ist also nicht geeignet, ein optimales α für die Bestimmung der Schwerefeldparameter zu liefern.

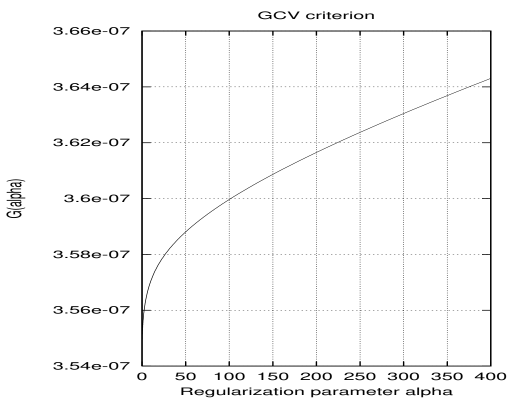


Abbildung 6: GCV–Funktion $G(\alpha)$

3.4 Quasioptimale Lösungen

Die Methode der Quasioptimalität beruht auf der Annahme, daß die Funktion $\|x_\alpha - x^*\|$, mit x^* der exakten Lösung des ungestörten Problems, bei dem optimalen α ein Minimum besitzt. Daher wird das α gesucht, das

$$\Psi(\alpha) = \left\| \alpha \frac{d}{d\alpha} x_\alpha \right\|$$

minimiert. Man kann zeigen, s. insbesondere HANSEN [23], daß unter geeigneten Voraussetzungen, u.a. wieder der diskreten Picard–Bedingung, näherungsweise gilt:

$$\Psi^2(\alpha) = \|x_0^* - x_\alpha^*\|_L^2 + \|x_\alpha^* - x_\alpha\|_L^2 \ .$$

Hierbei ist x_α^* die Lösung des regularisierten, aber ungestörten Problems. Demnach versucht dieses Verfahren in ähnlicher Weise wie die Methode der L–Kurve einen Kompromiß zwischen Minimierung des durch die Regularisierung eingeführten Fehlers $x_0^* - x_\alpha^*$ und Minimierung des Störungsfehlers $x_\alpha^* - x_\alpha$ zu finden.

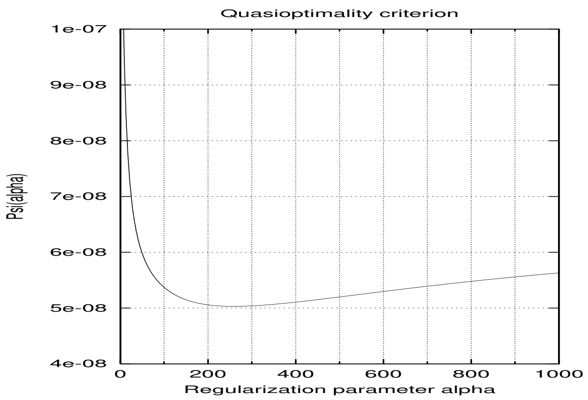


Abbildung 7: Quasioptimale Lösungen: $\Psi(\alpha)$

Sei x_{α_1} die Lösung des Minimierungsproblems (2). Weiterhin bezeichne x_{α_2} die Lösung des Problems

$$\min_{x\in\mathbb{R}^n} \{\|Ax - b\|^2 + \alpha\|L(x - x_{\alpha_1})\|^2\} \ .$$

Dann läßt sich x_{α_2} schreiben als:

$$\begin{aligned} x_{\alpha_2} &= x_{\alpha_1} + \alpha(A^TA + \alpha L^TL)^{-1}L^TLx_{\alpha_1} \\ &= x_{\alpha_1} - \alpha \left. \frac{d}{d\alpha} x_\alpha \right|_{\alpha=\alpha_1} \ . \end{aligned}$$

Speziell für die Schwerefeldinversion ergibt sich somit als Zielfunktion, deren Minimum gesucht ist

$$\Psi(\alpha) = \alpha \|(A^T PA + \alpha L^T L)^{-1} L^T L x_{\alpha_1}\| \,.$$

Für jedes α muß also die Cholesky–Zerlegung von $A^T PA + \alpha L^T L$ nacheinander auf zwei Vektoren angewendet werden. Die Resultate der entsprechenden Berechnungen zeigt Abb. 7.
Die Funktion $\Psi(\alpha)$ hat ein Minimum bei $\alpha = 260$, d.h. die Methode quasioptimaler Lösungen ergibt ein erheblich größeres α als das Kriterium der L–Kurve. Ein ähnliche Beobachtung wurde auch von Bouman [19] gemacht bei der Simulation von fluggravimetrischen Messungen, ohne daß dafür eine theoretische Erklärung gegeben werden kann.

3.5 Hanke–Raus-Verfahren

Wie bereits bei dem Verfahren der L–Kurve erwähnt, ist es eine theoretisch erwünschte Eigenschaft einer bestimmten Methode, daß für einen kleiner werdenden Fehler der rechten Seite, die mit dem jeweils optimalen α regularisierte Lösung gegen die exakte Lösung des ungestörten Problems konvergiert. Solche sog. ordnungs–optimalen Methoden wurden entwickelt von Raus [32], Engl und Gfrerer [33] und Gfrerer [34]. Davon ausgehend entwickelten Hanke und Raus [35] heuristische Regeln zur Bestimmung eines optimalen α für verschiedene Regularisierungsverfahren.

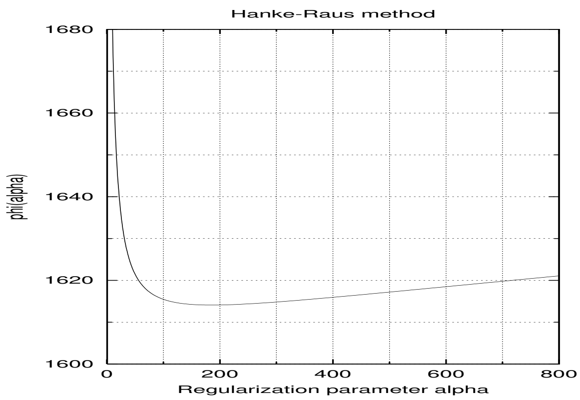


Abbildung 8: Hanke–Raus Verfahren: $\varphi(\alpha)$

Für die Tichonov–Regularisierung erhält man auf diese Weise als Zielfunktion, deren Minimum gesucht ist, den Ausdruck

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

Hierbei sind x_{α_1} und x_{α_2} die gleichen Größen wie sie bereits bei der Methode quasioptimaler Lösungen verwendet wurden. Im Rahmen der Schwerefeldbestimmung hat damit man folgende Funktion auszuwerten

$$\begin{aligned}\varphi(\alpha) &= \sqrt{1+\frac{1}{\alpha}}\cdot \\ &\quad \left[l^T Pl - dx_{\alpha_1}^T A^T Pl - dx_{\alpha_2}^T (A^T PA + \alpha L^T L)(dx_{\alpha_2} - dx_{\alpha_1})\right]^{1/2} .\end{aligned}$$

Die Methode ist vom Aufwand her offensichtlich äquivalent zum Verfahren quasioptimaler Lösungen; die Ergebnisse sind in Abb. 8 dargestellt. Als Minimum von $\varphi(\alpha)$ erhält man $\alpha = 186$, aber auch hier ist das Minimum nicht sehr scharf definiert.

3.6 Generalized–maximum–likelihood Verfahren

Neumaier [36] definierte ebenfalls ausgehend von statistischen Annahmen über die Verteilung und Korrelation der Fehler eine Klasse von Verfahren zur Bestimmung eines optimalen α. Diese Klasse enhält als Spezialfälle zum einen das GCV–Verfahren, zum anderen ein sog. generalized maximum likelihood–Ansatz (GML). Dieser liefert, unter einschränkenden Voraussetzungen, die klassische maximum likelihood–Abschätzung für α. Zielfunktion ist hierbei das Funktional

$$\mathrm{GML}(\alpha) = \frac{1}{\alpha} b^T (b - Ax_\alpha) \sqrt[n]{\det(A^T A + \alpha L^T L)} .$$

Im Fall der Schwerefeldinversion kann dieser Ausdruck umgeformt werden zu

$$\mathrm{GML}(\alpha) = \frac{1}{\alpha}(\|r_\alpha\|_P^2 + \alpha\|x_\alpha\|_L^2) \sqrt[n]{\det(A^T PA + \alpha L^T L)} .$$

Dieser Ausdruck enthält gerade den Term $\|r_\alpha\|_P^2 + \alpha\|x_\alpha\|_P^2$, der für ein festes α durch die Tichonov–Regularisierung minimiert wird; seine numerische Auswertung ist nicht schwierig. Sei wieder $\Lambda\Lambda^T$ die Cholesky–Zerlegung von $A^T PA + \alpha L^T L$. Dann gilt für die Determinante

$$\det(A^T PA + \alpha L^T L) = \det(\Lambda)^2 = \left(\prod_{i=1}^{n} \lambda_{ii}\right)^2 .$$

Die berechnete Funktion $\mathrm{GML}(\alpha)$ ist in Abb. 9 dargestellt.
Das im Vergleich zu den anderen Methoden relativ scharf ausgeprägte Minimum liegt hier bei $\alpha = 44$, ist also kleiner als bei allen anderen Verfahren.

3.7 Mean square error matrix Analyse

Die mean square error (MSE) Matrix wurde von verschiedenen Autoren innerhalb der Schwerefeldbestimmung eingeführt, um die verschiedenen Fehlereinflüsse zu unterscheiden und ihren Einfluß auf die Lösung zu quantifizieren,

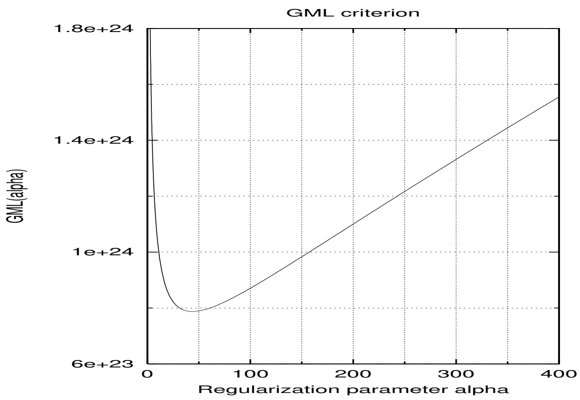


Abbildung 9: GML–Verfahren

s. hierzu insbesondere BOUMAN [19, 37, 38], BOUMAN UND KOOP [39, 40] sowie FLOBERGHAGEN UND BOUMAN [26]. Ziel ist hierbei die Beurteilung der Qualität der Lösung für ein festes α a posteriori. Wiederum ist der Ausgangspunkt eine Annahme über die Fehlerverteilung in den Messungen, also der rechten Seite b:

$$\begin{aligned} b &= \bar{b} + \epsilon\,, \\ E\{\epsilon\} &= 0\,, \\ E\{\epsilon\epsilon^T\} &= P^{-1} \ \text{(Kaula–Regel)}\,. \end{aligned}$$

Hierbei kennzeichnet $E\{\cdot\}$ den Erwartungswert. Damit läßt sich für den Erwartungswert der regularisierten Lösung ableiten:

$$\begin{aligned} E\{x_\alpha\} &= (A^TPA + \alpha L^TL)^{-1}A^TPE\{b\} \\ &= (A^TPA + \alpha L^TL)^{-1}A^TPAx_0^*\,. \end{aligned}$$

Damit ist der Einfluß durch die Regularisierung abschätzbar als

$$\delta x := E\{x_\alpha - x_0^*\} = -\alpha(A^TPA + \alpha L^TL)^{-1}L^TLx_0^*\,.$$

Durch Fehlerfortpflanzung, angewandt auf $x_\alpha = (A^TPA + \alpha L^TL)^{-1}A^TPb$, erhält man nun für die Kovarianzmatrix Q_x der regularisierten Lösung

$$Q_x = (A^TPA + \alpha L^TL)^{-1}A^TPA(A^TPA + \alpha L^TL)^{-1}\,.$$

Der gesamte Fehler, gesehen als Summe der Fehler in den Beobachtungen und der durch die Regularisierung eingeführten, wird nun definiert durch die sog. mean square error Matrix

$$\mathrm{MSEM} = Q_x + \delta x \delta x^T\,.$$

Mit Hilfe dieser Matrix wurde verschiedentlich versucht, den Einfluß der Meßfehler und der Regularisierung im Detail zu analysieren, s. die Arbeiten von BOUMAN ET AL. [19, 37, 38, 39, 40, 26]. Der mittlere Fehler („mean square error" MSE) der Lösung ist nun definiert als die Spur dieser Matrix und kann interpretiert werden als Erwartungswert der Differenz zwischen der wahren Lösung x_0^* und x_α:

$$\mathrm{MSE} = \mathrm{tr}\, Q_x + \mathrm{tr}\,(\delta x \delta x^T)\ .$$

Der erste Term der rechten Seite dieses Ausdrucks, also die Summe der Varianzen für die Lösungskomponenten, wird kleiner mit größer werdendem α. Der zweite Term als Maß des Einflusses der Regularisierung ist 0 für $\alpha = 0$ und wächst dann kontinierlich. Ziel ist nun, die Summe MSE(α) zu minimieren. Dies ist aber direkt nicht möglich, da er offensichtlich x_0^* selber enthält. Damit ist nur eine Abschätzung der MSE möglich, indem man stattdessen x_α verwendet, obwohl dann der Einfluß der Regularisierung systematisch unterschätzt wird, s. XU [41].
Die numerische Auwswertung des Spurausdrucks erfordert einen höheren Rechenaufwand als die bisher vorgestellten Verfahren. Zur Berechnung von Q_x ist die folgende Umformung vorteilhaft.

$$\begin{aligned} Q_x &= (A^TPA + \alpha L^TL)^{-1} A^TPA (A^TPA + \alpha L^TL)^{-1} \\ &= (A^TPA + \alpha L^TL)^{-1} \\ &\quad - \alpha\, (A^TPA + \alpha L^TL)^{-1} L^TL (A^TPA + \alpha L^TL)^{-1}\ . \end{aligned}$$

Damit ergibt sich für $\mathrm{tr}\, Q_x$:

$$\begin{aligned} \mathrm{tr}\, Q_x &= \mathrm{tr}\left[(A^TPA + \alpha L^TL)^{-1}\right] \\ &\quad - \alpha\, \mathrm{tr}\left\{[L(A^TPA + \alpha L^TL)^{-1}]^T [L(A^TPA + \alpha L^TL)^{-1}]\right\} \\ &= \sum_{i=1}^{n} \left[(ATPA + \alpha L^TL)^{-1}\right]_{ii} \\ &\quad - \alpha \sum_{k=1}^{n} (L^TL)_k \sum_{i=1}^{n} \left[(A^TPA + \alpha L^TL)^{-1}\right]_{ki}^2\ . \end{aligned}$$

Sei $\Lambda\Lambda^T$ wieder die Cholesky–Zerlegung von $A^TPA + \alpha L^TL$, die Matrixelemente von Λ^{-1} seien $\tilde{\lambda}_{ij}$. Damit erhält man

$$\mathrm{tr}\, Q_x = \sum_{i=1}^{n} \sum_{k=1}^{n} \tilde{\lambda}_{ki}^2 - \alpha \sum_{k=1}^{n} (L^TL)_k \sum_{i=1}^{n} \left(\sum_{j=1}^{n} \tilde{\lambda}_{jk} \tilde{\lambda}_{ji} \right)^2\ .$$

Damit ist zur Berechnung von MSE(α) für jedes α die volle Invertierung der Matrix Λ notwendig. $\mathrm{tr}(\delta x \delta x^T)$ läßt sich schreiben als

$$\mathrm{tr}\,(\delta x \delta x^T) = \|(A^TPA + \alpha L^TL)^{-1} L^TLx_0^*\|^2\ .$$

Setzt man für x_0^* näherungsweise x_α, so wird hieraus

$$\mathrm{tr}\,(\delta x \delta x^T) = \|dx_{\alpha_2} - dx_{\alpha_1}\|^2 \,, \tag{10}$$

dies ist genau das Funktional, das bei der Methode quasioptimaler Lösungen, s. Abschnitt 3.4, minimiert wird. Die Rechnungen für die Schwerefeldinversion ergeben hiermit die in Abb. 10 dargestellten Resultate.

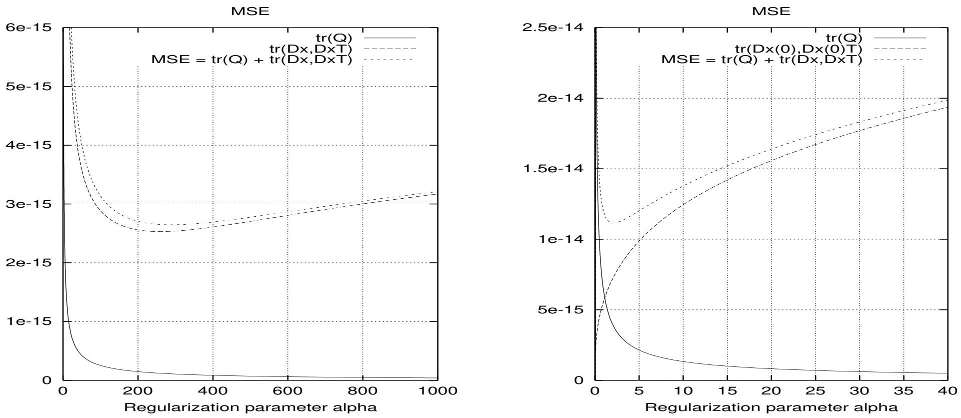


Abbildung 10: mean square error - MSE mit x_α und x_0

Obwohl die MSE–Funktion ein Minumum bei $\alpha = 290$ besitzt, ist diese Methode nicht geeignet, ein optimales α zu liefern, da die Funktion $\mathrm{tr}(\delta x, \delta x^T)$ offensichtlich nicht den korrekten Verlauf für $\alpha \to 0$ aufweist. Daher ist es sinnvoller, zur Berechnung diese Beitrages ein festes x zu verwenden. Eine Möglichkeit ist z.B. das vorliegende Schwerefeldmodell niederer Ordnung, also x_0, zu verwenden. Damit erhält man die ebenfalls in Abb. 10 gezeigten Ergebnisse. Nun besitzen beide Funktionen zwar das theoretisch richtige Aussehen, aber das Minimum bei $\alpha = 2.1$ ergibt sicherlich keinen optimalen Regularisierungsparameter.

Statt x_0, das wohl nur eine grobe Näherung ist, kann natürlich jedes beliebige x_α, nun aber zu einem festen α–Wert, genommen werden. Sei nun α^* derjenige Wert des Regularisierungsparameters, für den die Tichonov–Regularisierung die beste Näherungslösung liefert. Dann sollte die Funktion $\mathrm{tr}(Q_x + \delta x_{\alpha^*} \delta x_{\alpha^*}{}^T)$ bei α^* ihr Minimum aufweisen, d.h. α^* sollte auf diese Weise rekonstruierbar sein. Daher wurden für alle α–Werte die Funktionen $\delta x \delta x^T$ nach (10) berechnet. In Abb. 11 sind die Ergebnisse für 4 ausgewählte α–Werte dargestellt.

Eine andere Möglichkeit der Beurteilung dieser Ergebnisse besteht darin, dasjenige α^* zu suchen, für das das Minimum der MSE–Funktion unter Verwendung von $\mathrm{tr}(\delta x_{\alpha^*} \delta x_{\alpha^*}{}^T)$ den kleinsten Wert annimmt. Aus diesem Grund ist in Abb. 12 das Minimum und der ensprechende α–Wert der MSE–Funktionen aufgetragen.

Die Ergebnise der Berechnungen lassen aber beide Interpretationen offensichtlich nicht zu. Einerseits wird das Minimum von MSE(α) mit wachsenden Wert für α immer kleiner. Zum anderen existiert kein ausgezeichneter Wert α^*, bei dem $\mathrm{tr}(Q_x + \delta x_{\alpha^*} \delta x_{\alpha^*}{}^T)$ tatsächlich sein Minimum hat. Die Differenz zwischen α^* und dem zugehörigen Minimum wird mit wachsendem Wert für α immer grösser. Beide Auslegungen führen also zu widersprüchlichen Resultaten. Offenbar ist die Methode der Analyse der MSE–Matrix nicht geeignet, ein optimales α zu identifizieren.

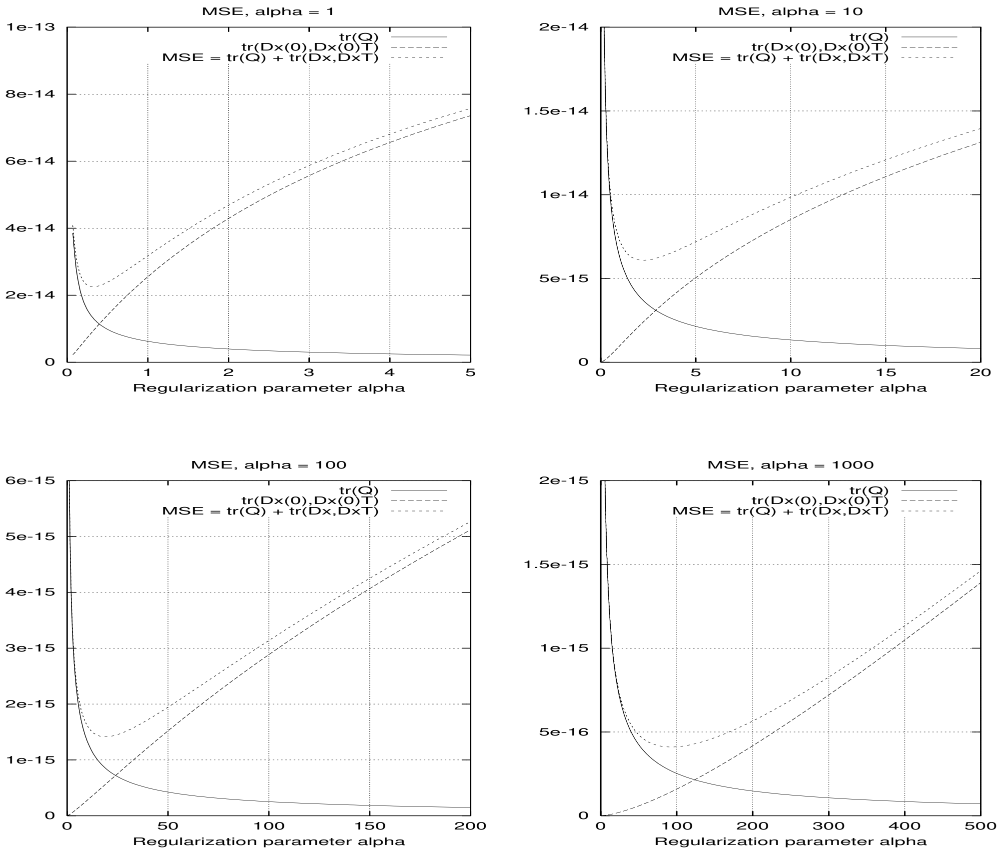


Abbildung 11: MSE–Funktionen für $\alpha = 1,\, 10,\, 100,\, 1000$

3.8 Vergleichende Bewertung der Methoden

Von den vorgestellten 6 Verfahren liefern 4 einen eindeutig bestimmten Wert für α. 2 Verfahren, GCV und MSE–Analyse ergeben $\alpha \to 0$ bzw. $\alpha \to \infty$, sind also für den Fall der Schwerefeldinversion nicht anwendbar. Tab. 8 faßt die Ergebnisse noch einmal zusammen.

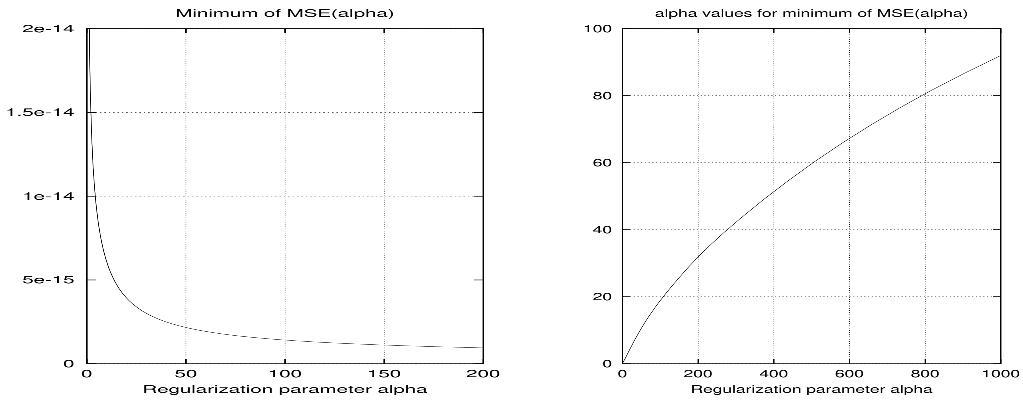


Abbildung 12: Minimum der MSE–Funktionen u. zugehörige α–Werte

Offensichtlich liefert jede Methode ein anderes α. Eine Beurteilung kann aber nur in Hinsicht auf die Unterschiede in den jeweils optimalen x_α erfolgen. Daher sind als Beispiel in Abb. 13 und 14 die absoluten und relativen Differenzen zwischen den Lösungen x_{69} und x_{100} bzw. x_{260} und x_{100} dargestellt. Im Zusammenhang hiermit muß die vermittels

$$V_x = \sigma^2 \, (A^T A)^{-1}$$

berechenbare Standardabweichung, Abb. 15, der Lösung gesehen werden.

Methode	optimales α
L–Kurve	69
Quasioptimale Lösungen	260
Hanke–Raus Verfahren	186
GML	44

Tab. 8 : Optimale α-Werte aus verschiedenen Methoden

Aus den in Abb. 13 und 14 dargestellten Daten folgt, daß für $\alpha = 69$ und $\alpha = 100$ sich in weniger als 3% der Komponenten des Lösungsvektors Unterschiede ergeben, die größer als die entsprechenden Standardabweichungen sind. Für $\alpha = 260$ und $\alpha = 100$ erhält man aber schon in etwa 20% eine signifikante Differenz. Tatsächlich unterscheiden sich im Bereich von $\alpha = 63$ bis $\alpha = 156$ die Lösungen x_α in weniger als 5% der Komponenten um mehr als die Standardabweichung; für 10% signifikante Abweichungen erhält man sogar einen Bereich von $\alpha = 55$ bis $\alpha = 182$. Demnach ist sinnlos, eine relative Genauigkeit eines optimalen α–Wertes kleiner als ± 40 (für ein α in der Größenordnung von etwa 100) zu fordern. Dennoch ergibt sich aus Abb. 14,

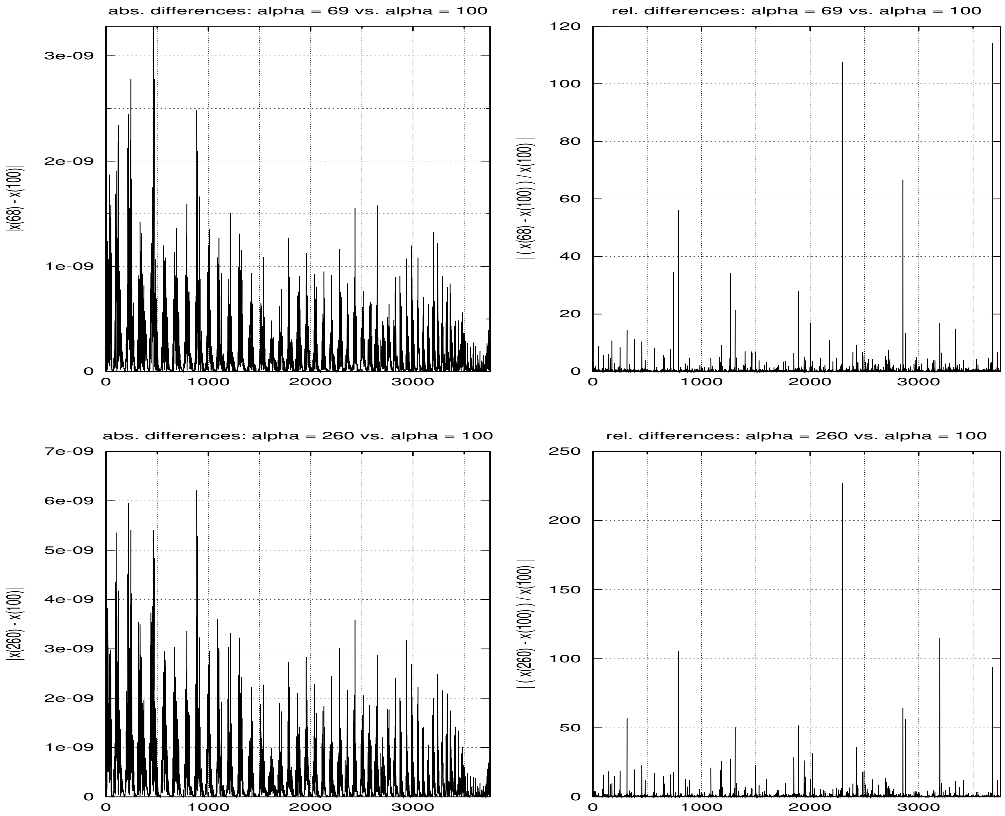


Abbildung 13: abs. und rel. Differenzen zwischen x_{100} und x_{69} bzw. x_{260}

.

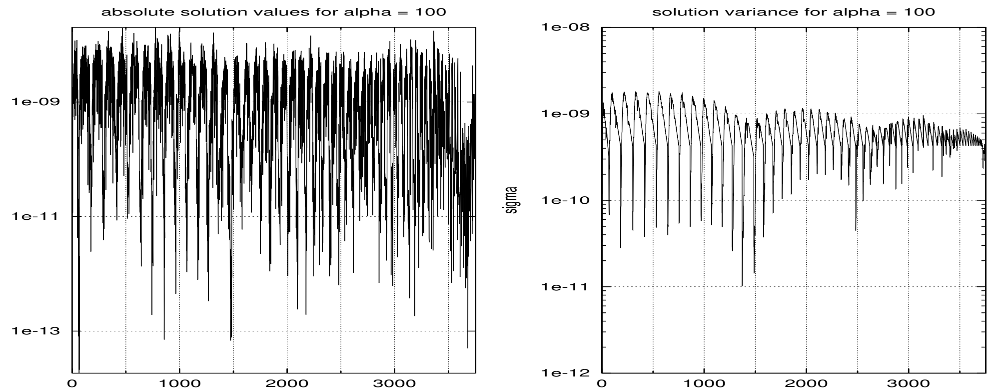


Abbildung 14: Absolutwerte und Standardabweichung von x_{100}

daß die Verfahren zur Ermittlung eines optimalen α auch unter Berücksichtigung der überhaupt möglichen Lösungsgenauigkeit deutlich unterschiedliche Resultate ergeben.
Auch in der Literatur wurden des öfteren verschiedene Ansätze zur Bestimmung eines optimalen Regularisierungsparameters in praktischen Problemstellungen verglichen. Von HANKE UND HANSEN [14] wurden umfangreiche Tests anhand von zwei Anwendungen aus der Sonnenphysik („helioseismology") und der Bild–Rekonstruktion durchgeführt, und zwar sowohl für die Tichonov–Regularisierung als auch für iterative Methoden. Die Resultate ergeben ein sehr uneinheitliches Bild. Für das Problem aus der Sonnenphysik wurden mit der Tichonov–Regularisierung die besten Resultate erhalten mit GCV oder dem L–Kurve Kriterium.
Beide Verfahren sind bezüglich Robustheit und Genauigkeit den anderen Methoden überlegen, wobei bei normal–verteilten Fehlern in der rechten Seite GCV am besten abschneidet. Bei Verwendung iterativer Verfahren ist die Situation aber erheblich weniger eindeutig. Im Fall der Bildrekonstruktion werden die besten Resultate für die Tichonov–Regularisierung mit der Methode von GFRERER UND RAUS [34, 32], ähnlich dem Hanke–Raus Verfahren in Abschnitt 3.5, der Methode quasioptimaler Lösungen und GCV erhalten, die Anwendung der L–Kurve führt dagegen zu deutlich größeren Fehlern. Wiederum ergeben sich bei Verwendung iterativer Verfahren andere Resultate.
HANSEN [23] verglich am Beispiel einer Fredholm–Integralgleichung GCV und L–Kurve Kriterium. Nach diesen Ergebnissen sind beide Verfahren bei unkorrelierten Fehlern in der rechten Seite des Gleichungssystems nahezu äquivalent, dagegen ist die L–Kurve deutlich überlegen, wenn starke Korrelationen dieser Fehler vorhanden sind.
NEUMAIER [36] überprüfte für eine ganze Klasse von Regularisierungsverfahren und verschiedene Fehler–Verteilungen die Approximationseigenschaften. GML ist demnach vor GCV das optimale und gleichzeitig am wenigsten aufwendige Verfahren. Nach WAHBA [42] ist unter gewissen Voraussetzungen dagegen GCV robuster als GML.
FLOBERGHAGEN UND BOUMAN [26] schließlich untersuchten bei der Rekonstruktion des Schwerefeldes des Mondes verschiedene Methoden zur Bestimmung eines optimalen Regularisierungsparameters, insbesondere auch das L–Kurve Kriterium und quasioptimale Lösungen. Allerdings stand ihnen kein Wert für das Residuum $\|r_\alpha\|_P$ zu Verfügung, lediglich eine Abschätzung über die Matrixnorm. Daher sind die Ergebnisse bezüglich der L–Kurve sicherlich nur Näherungen, da hier der genaue Wert des Residuums unabdingbar ist. Die Autoren konstatieren, daß keine Methode geeignet ist, einen zuverlässigen Wert für α zu liefern, die ermittelten Werte hierfür sind wesentlich zu klein. Sie betonen auch, daß in diesem Fall die Zahl der Entwicklungskoeffizienten außerordentlich groß ist im Vergleich zu den Verfügung stehenden Informationen.
Zusammenfassend ergibt sich, daß kein eindeutig vorzuziehendes Verfahren

zur Bestimmung eines optimalen Wertes für α existiert. Dies wird durch die Rechnungen, die hier für die Schwerefeldinversion vorgenommen wurden, bestätigt. Offensichtlich wird aber die Chance, mit verschiedenen Methoden zu einem etwa gleichen Resultat zu gelangen, umso kleiner, je schlechter konditioniert das Gleichungssystem und je grösser die Fehler in der rechten Seite bzw. der Matrix selber sind. Die Tatsache, daß das GCV–Verfahren im vorliegenden Fall versagt, ist insbesondere ein starkes Indiz für hoch korrelierte Fehler.
Um die Möglichkeiten und Grenzen der Regularisierung genauer abzuschätzen, wird im folgenden daher die Regularisierung auf Basis einer Eigenwert–Zerlegung von AT^PA analysiert.

3.9 Eigenwert–Zerlegung und Regularisierung

Wie bereits erwähnt, können zahreiche Eigenschaften der Regularisierung anhand einer SVD–Zerlegung der Matrix A untersucht werden. Dies ist eine Zerlegung der Gestalt

$$A = U\Sigma V^T = \sum_{i=1}^{n} u_i \sigma_i v_i^T \ .$$

Hierbei sind $U = (u_1, \ldots, u_n)$ und $V = (v_1, \ldots, v_n)$ orthogonale Matrizen, d.h. $U^TU = V^TV = I_n$, und $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_n)$, $\sigma_1 \geq \cdots \geq \sigma_n \geq 0$, die Singulärwerte von A. Die Spalten von U und V sind die linken bzw. rechten Singulärvektoren. Hat man eine SVD–Zerlegung vorliegen, so ist die Lösung des nicht regularisierten Gleichungssystems (1) gegeben durch

$$x^* = \sum_{i=1}^{n} \frac{u_i^T b}{\sigma_i} v_i \ .$$

Diese Gleichung illustriert den starken Einfluß der kleinsten σ–Werte auf die Lösung. Eine regularisierte Lösung läßt sich demgegenüber schreiben als

$$x_{reg} = \sum_{i=1}^{n} f_i \frac{u_i^T b}{\sigma_i} v_i \ ,$$

hierbei sind f_i die Filterfaktoren, für diejenigen der Tichonov–Regularisierung gilt z.B. $f_i = \sigma_i^2/(\sigma_i^2 + \alpha)$. Diese Faktoren bewirken offensichtlich eine Dämpfung der Beiträge der kleinsten σ_i. Diese Dämpfung wird wirksam etwa ab $\sigma_i = \sqrt{\alpha}$.
Im Fall der Schwerefeldbestimmung liegt nun nur A^TA, bzw. A^TPA vor. Sei nun $A = U\Sigma V^T$ die SVD–Zerlegung von A, so folgt

$$A^TA = V\Sigma U^T U\Sigma V^T = V\Sigma^2 V^T = V\Lambda V^T = \sum_{i=1}^{n} v_i \lambda_i v_i^T \ .$$

Dies ist gerade eine Eigenwert– oder Schur–Zerlegung von $A^T A$, V ist die Matrix, deren Spalten die orthonormalen Eigenvektoren sind, $\Lambda = \Sigma^2$ ist die Diagonalmatrix der Eigenvektoren von $A^T A$, diese sind also die Quadrate der Singulärwerte von A. Das Gleichungssystem $A^T A x = A^T b$ läßt sich demnach schreiben als $V \Lambda V^T x = A^T b$. Somit ist die nicht regularisierte Lösung x^* gegeben durch

$$x^* = V \Lambda^{-1} V^T A^T b = \sum_{i=1}^{n} \frac{v_i^T A^T b}{\lambda_i} v_i \ .$$

Die Regularisierung dämpft nun entsprechend den Beitrag der kleinsten Eigenwerte von $A^T A$ auf die Lösung:

$$x_{reg} = \sum_{i=1}^{n} f_i \frac{v_i^T A^T b}{\lambda_i} v_i \ .$$

Die Filterfaktoren der Tichonov–Regularisierung sind nun gegeben durch

$$f_i = \frac{\lambda_i}{\lambda_i + \alpha} \ .$$

Mit Hilfe von Filterfaktoren lassen sich nun auch andere Regularisierungsmethoden definieren, z.B. truncated SVD, Hansen [15, 16], Bouman [19]:

$$f_i = \begin{cases} 0, & \lambda_i < \alpha \\ 1, & \lambda_i \geq \alpha \end{cases} \ ,$$

oder exponential filtering, Calvetti, Lewis und Reichel [20]:

$$f_i = 1 - \exp\left(-\frac{\lambda_i}{\alpha}\right) \ .$$

Die Filterfaktoren von truncated SVD und Tichonov–Regularisierung können dabei als rationale Approximationen an die Faktoren zu exponential filtering interpretiert werden, s. [20].
Um diese Regularisierungverfahren auf die Schwerefeldinversion anwenden zu können, ist von der Formulierung

$$A^T P A x = A^T P (l + A x_0) = A^T P \tilde{l}$$

auszugehen, um der Minimierungsbedingung (8) entsprechende Lösungen zu erhalten. Der Einfluß der Kaula–Matrix wird berücksichtigt durch die Transformation in Standardform vermittels $\tilde{A} = A L^{-1}$:

$$\tilde{A}^T P \tilde{A} \, L x = \tilde{A}^T P \tilde{l} \ .$$

Man berechnet daher zuerst die Schur–Zerlegung

$$V \Lambda V^T = \tilde{A}^T P \tilde{A} \ .$$

Sei F die Diagonalmatrix der Regularisierungsfaktoren f_i. Dann gilt nun für eine regularisierte Lösung

$$x_{reg} = L^{-1} V F \Lambda^{-1} V^T \tilde{A}^T P \tilde{l} = L^{-1} \sum_{i=1}^{n} \frac{f_i v_i^T L^{-T} A^T P \tilde{l}}{\lambda_i} v_i \ .$$

Das Residuum von x_{reg} ist definiert durch

$$\|r_{reg}\|_P^2 = \tilde{l}^T P \tilde{l} - 2 x_{reg}^T A^T P \tilde{l} + x_{reg}^T A^T P A x_{reg} \ .$$

Der Ausdruck $x_{reg}^T A^T P A x_{reg}$ kann hierbei effizient folgendermaßen berechnet werden. Zuerst einmal gilt

$$\begin{aligned} A^T P A x_{reg} &= L^T V \Lambda V^T L \, L^{-1} V F \Lambda^{-1} V^T \tilde{A}^T P \tilde{l} \\ &= L^T V F V^T \tilde{A}^T P \tilde{l} \ . \end{aligned}$$

Damit ergibt sich für das Skalarprodukt

$$\begin{aligned} x_{reg}^T A^T P A x_{reg} &= \left[L^{-1} V F \Lambda^{-1} V^T \tilde{A}^T P \tilde{l} \right]^T \left[L^T V F V^T \tilde{A}^T P \tilde{l} \right] \\ &= \left[\Lambda^{-1} F V^T \tilde{A}^T P \tilde{l} \right]^T \left[F V^T \tilde{A}^T P \tilde{l} \right] \\ &= \sum_{i=1}^{n} \frac{(f_i v_i^T \tilde{A}^T P \tilde{l})^2}{\lambda_i} \ . \end{aligned}$$

Da der Vektor $V^T \tilde{A}^T P \tilde{l}$ auch zur Berechnung von x_{reg} benötigt wird, ergibt sich die Norm des Residuums so ohne weiteren Aufwand. Ist eine Schur–Zerlegung von $\tilde{A}^T P \tilde{A}$ gegeben, so erfordert die Berechnung einer regularisierten Lösung damit im wesentlichen nur zwei Matrix–Vektor Multiplikationen, einmal mit V^T und einmal mit V. Sie ist daher dann problemlos für einen variablen Satz von Regularisierungsvarianten durchführbar. Der Gesamtaufwand zur Berechnung von Lösungen zu vielen verschiedenen α–Werten kann auf diese Weise sogar drastisch reduziert werden, da eine Eigenwert–Zerlegung nur etwa 20–mal mehr Zeit benötigt als eine Cholesky–Faktorisierung, s. auch Abschnitt 2.2 und 2.4.

Die Realisierung dieses Verfahrens liefert nun bei Verwendung der Filterfaktoren der Tichonov–Regularisierung exakt die gleichen Ergebnisse wie die Lösung der entsprechend regularisierten Normalgleichungssysteme, es lassen sich nun aber auch andere Filterfaktoren anwenden. Da relevante Filterfaktoren die Größenordnung der Eigenwerte haben, wurden die Normen der Lösungen und der Residuen für $\alpha = \lambda_i$, $i = 1, \dots, n$ für die drei angeführten Regularisierungen berechnet. Da die Unterschiede in den Ergebnissen gering sind, sollen sie hier nur als L–Kurven dargestellt werden, s. Abb. 15. In dieser Abb. sind die „hochaufgelösten Ecken“ der L–Kurven für die drei Filterfaktoren gezeigt, man vergleiche den Maßstab mit dem in Abb. 5.

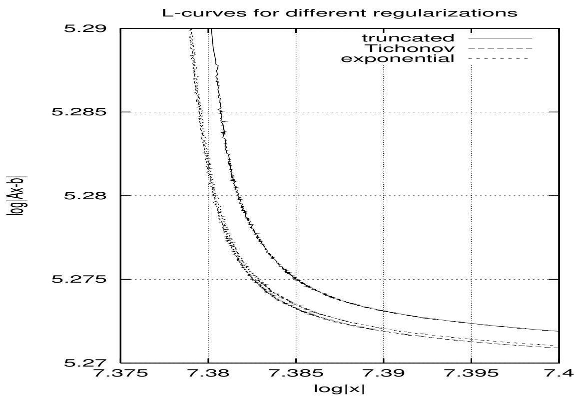


Abbildung 15: L–Kurven für verschiedene Regularisierungen

Die aus diesen Resultaten sich ergebenden optimalen α–Werte sind $\alpha \approx 70$ für die truncated Eigenwert–Regularisierung (infolge der unstetigen Filterfaktoren nicht exakt lokalisierbar), $\alpha = 69$ für die Tichonov–Regularisierung und $\alpha = 73$ für die Regularisierung durch exponential filtering. Die sich hiermit ergebenden optimalen Filterfaktoren sind in Abb. 16 dargestellt. Hierbei ist zusätzlich die auf 1 normierte Eigenwert–Verteilung aufgetragen.

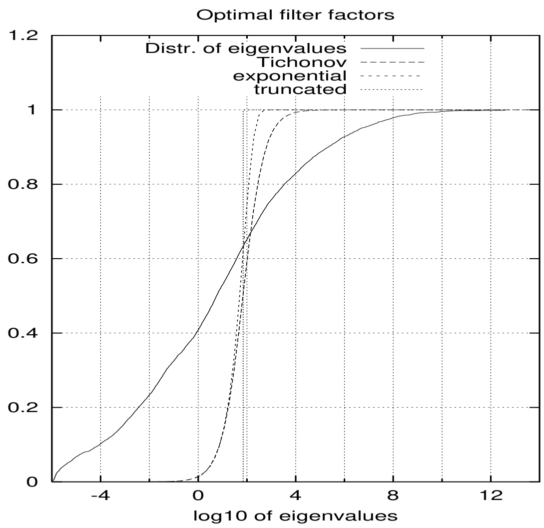


Abbildung 16: Filterfaktoren für verschiedene Regularisierungen

Aus Abb. 16 läßt sich unmittelbar ablesen, daß durch die ermittelten optimalen Filterfaktoren etwa 60% aller Eigenvektoren von $\tilde{A}^T P \tilde{A}$ bzw. $A^T P A$ praktisch unberücksichtigt bleiben, d.h. die Schwerefeldparameter werden etwa aus nur etwa 40% aller Eigenwerte und –vektoren berechnet. Setzt man

$\alpha = 100$, so ist sogar nur etwa 35% der in dieser Matrix enthaltenen „Information“ relevant für die Lösung, dies ist ein außerordentlich geringer Anteil. Es erscheint daher unwahrscheinlich, daß auf Grundlage der vorliegenden Daten eine Entwicklung des Schwerefelds der Erde in die derzeit aktuelle Zahl von Koeffizienten (3754) zuverlässig möglich ist.
Dies soll abschließend durch vergleichende Rechnungen mit einer verringerten Zahl von Koeffizienten belegt werden.

3.10 Reduktion der Zahl der Koeffizienten

Gegeben sei wieder das Gleichungssystem (1), also $Ax = b$. Eine Reduktion der Zahl der Unbekannten ist äquivalent zu der Streichung einer Spalte in der Matrix A, bzw. der Streichung einer Spalte und einer Zeile in A^TPA.
Als heuristisches Kriterium zur Verringerung der Zahl der Koeffizienten diente die Größe des Eintrags in die Kaula–Matrix L^TL. Das bedeutet, daß alle Spalten und Zeilen aus A^TA bzw. A^TPA gestrichen werden, wenn die entsprechende Komponente von L^TL eine gewisse Grenze überschreitet. Diese Streichung kann auch dergestalt interpretiert werden, daß der zugehörige Eintrag in L^TL als unendlich groß angenommen wird. In gleicher Weise müssen gegebenenfalls Komponenten in A^Tb, L und in x_0 gestrichen werden. Diese Vorgehensweise soll natürlich nur zur Demonstration dienen.
Die obere Grenze des Eintrags in die Kaula–Matrix wurde nun sukzessive verringert, dies ergab ein immer weiter reduziertes System. Für jedes dieser Systeme wurde über eine Eigenwert–Zerlegung mit Hilfe des Verfahrens der L–Kurve der optimale Regularisierungsparameter α bestimmt. In den Abb. 17 und 18 sind die Resultate zusammen mit denen für ein festes $\alpha = 100$ dargestellt.

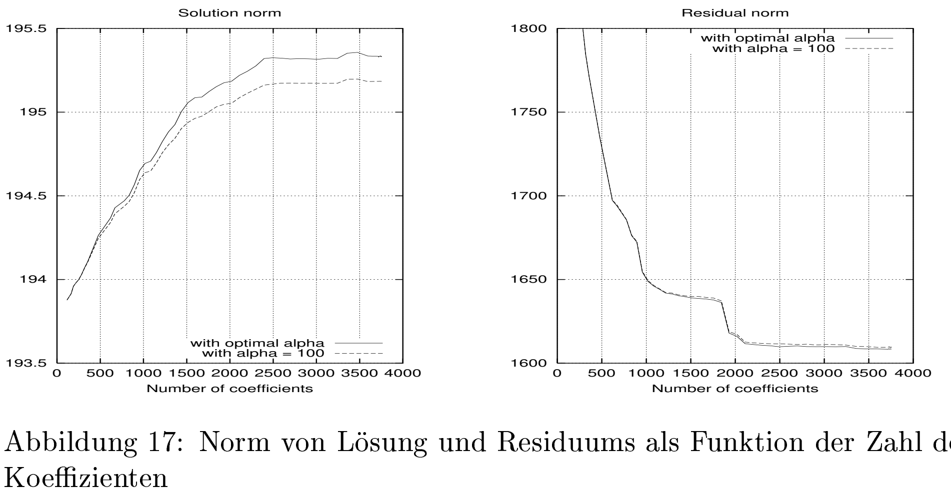


Abbildung 17: Norm von Lösung und Residuums als Funktion der Zahl der Koeffizienten

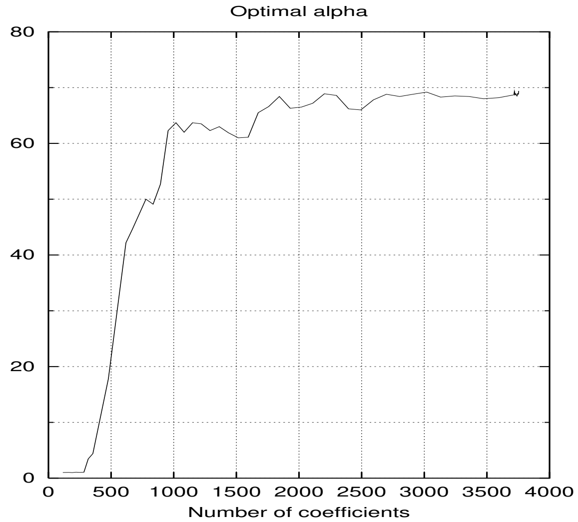
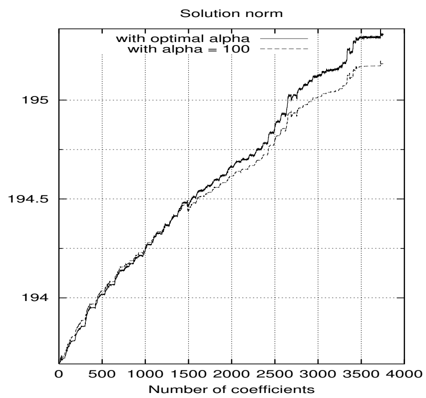


Abbildung 18: Optimale α–Werte als Funktion der Zahl der Koeffizienten

Aus den Abbildungen ergibt sich, daß die Zahl der Koeffizienten auf etwa 2250 reduziert werden kann, ohne die Norm des Residuums merklich größer werden zu lassen, bzw. die der Lösung wesentlich zu verringern. Auch der optimale Regularisierungsparameter α bleibt in diesem Bereich offensichtlich nahezu gleich. Als Beispiel sei das System mit 2395 Unbekannten betrachtet. Dieses ergibt sich bei einem maximalen Eintrag in die Kaula–Matrix von $5.67 \cdot 10^{16}$. Als optimalen Regularisierungsparameter berechnet man $\alpha = 66.2$. Der kleinste Eigenwert der unregularisierten Matrix ist 0.0039, ist also positiv, der Anteil der tatsächlich berücksichtigten Eigenwerte und –vektoren beträgt etwa 60 %. Die L–Kurve und deren Krümmung sind praktisch identisch mit der aus Abb. 5. Zusammen ergibt sich hieraus, daß durch diese 2395 Koeffizienten das System ebenso genau gelöst wird wie durch das Gesamtsystem mit 3754 Unbekannten. Aus Abb. 17 folgt auch, daß der Einfluß von α bei mehr als 2250 Koeffizienten größer ist als der durch die Zahl der Koeffizienten selber. Angemerkt sei noch, daß eine Bestimmung eines optimalen α durch das GCV–Verfahren, s. Abschnitt 3.3, auch für das System mit 2395 Unbekannten nicht möglich ist, die Funktion $G(\alpha)$ ist der in Abb. 6 sehr ähnlich.

Eine andere Möglichkeit der Koeffizientenreduktion besteht darin, sukzessive das System um eine Unbekannte zu verringern, unabhängig von der Größe des zugehörigen Eintrages in die Kaula–Matrix. Die Durchführung dieses Verfahrens ergibt die Ergebnisse in Abb. 19 und 20. Wiederum veranschaulicht der treppenartige Verlauf der Norm von Lösung und Residuum die Tatsache, daß nur eine Teilmenge der Koeffizienten einen signifikanten Beitrag zur Lösung liefert. Allerdings wird dies hier weniger deutlich als bei der Reduktion nach

Größe des entsprechenden Elements in der Kaula–Matrix. Außerdem zeigt die Abhängigkeit des optimalen α von der Zahl der Koeffizienten, s. Abb. 20, daß für die Lösung wesentliche Komponenten für $n < 1500$ fehlen, da dann α deutlich größer wird, anstatt für $n \to 0$ kontinuierlich kleiner zu werden, etwa wie in Abb. 18.



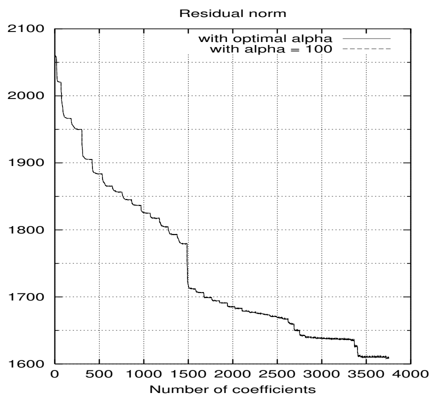


Abbildung 19: Norm von Lösung und Residuums als Funktion der Zahl der Koeffizienten

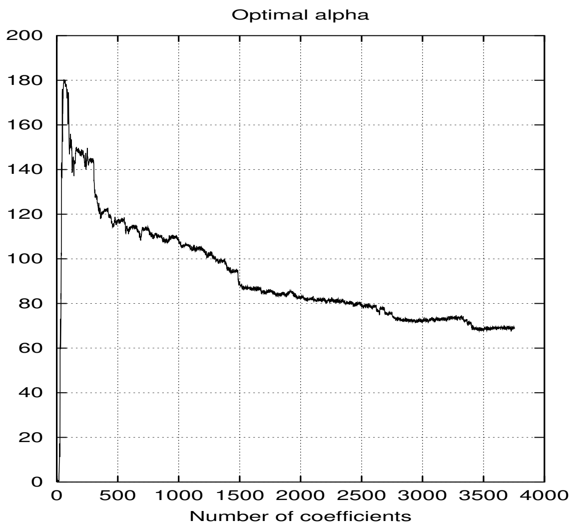


Abbildung 20: Optimale α–Werte als Funktion der Zahl der Koeffizienten

3.11 Schlußfolgerungen

Wie bereits dargelegt, kann die aktuelle Zahl von Schwerefeldparametern $n = 3754$ aus den gegebenen Daten nicht zuverlässig bestimmt werden. Dies wird deutlich zum einen durch die uneinheitlichen Resultate der verschiedenen Methoden zur Bestimmung eines optimalen Regularisierungsparameters. Zum anderen zeigt eine detaillierte Analyse der vorliegenden Daten, daß durch die Regularisierung etwa 60 % der Information in der Matrix verworfen wird. Demgegenüber können mit einer deutlich geringeren Zahl von Parametern bezüglich der Norm des Residuums ebenso gute Lösungen erzeugt werden. Alle Folgerungen sind natürlich auf den vorliegenden Datensatz beschränkt.

Letztendlich geht es bei der zugrundeliegenden Problemstellung darum, aus vermessenen Satellitenbahnen das Schwerefeld der Erde zu rekonstruieren. Um nun zu einer unbeeinflußten Abschätzung über die notwendige und sinnvolle Zahl von Schwerefeldparametern zu gelangen, wird folgende Vorgehensweise vorgeschlagen. Damit soll ermittelt werden, wie exakt die Parameter des Schwerefeldes bekannt sein müssen, um die Satellitenbahn mit einem vorgegebenen Fehler berechnen zu können.

Hierfür werden aus Lösungen des Schwerefeldsystems mit einer unterschiedlichen Zahl von Parametern und variablem Regularisierungsparameter realistische Satellitenbahnen berechnet. Dies sollte unter Vernachlässigung aller anderen die Satellitenbahn beeinflussenden Kräfte (Mond, Sonnenwind, Atmosphäre) mit Hilfe eines hochgenauen Integrators unter Kontrolle des globalen Integrationsfehlers durchgeführt werden.

Parametersätze, die hierbei Lösungen liefern, deren Unterschiede in den berechneten Bahnen geringer sind als die Genauigkeit der Bahnvermessung, müssen demzufolge als ununterscheidbar angesehen werden, auch wenn signifikante Unterschiede in einzelnen Koeffizienten vorhanden sind. Nur solche Parametersätze bzw. Regularisierungsparameter, die auch unter diesen idealisierten Verhältnissen unterscheidbare Satellitenbahnen ergeben, können als verschiedene Modellierungen des Schwerefelds betrachtet werden. Diese Vorgehensweise sollte es ermöglichen, die bei der aktuellen Präzision der Bahnmessungen sinnvolle Genauigkeit der Bestimmung der Schwerefeldparameter festzulegen.

Die Frage des hierzu notwendigen Integrators wird im nächsten Kapitel eingehend diskutiert.

4 Integration einfacher Satellitenbahnen

Bedingt durch die immer höhere Meßgenauigkeit bei der Beobachtung von Satellitenbahnen werden auch die Genauigkeitsanforderungen bei der numerischen Integration von Satellitenbahnmodellen immer größer. In der hier vorgestellten Untersuchung soll anhand zweier sehr einfacher Modellprobleme diese Problematik näher analysiert werden.
Es werden im wesentlichen je ein Verfahren aus den drei Verfahrensklassen Mehrschrittverfahren, Runge–Kutta–Verfahren und Extrapolationsverfahren betrachtet. Dabei zählen diese Verfahren der gegenwärtigen Lehrmeinung nach zu den besten ihrer jeweiligen Klasse. Auf die Untersuchung von Verfahren, die speziell für Systeme 2. Ordung konstruiert sind wurde bewußt verzichtet, da sie i.a. nur dann Vorteile bieten, wenn die Gleichung eine spezielle Form, etwa

$$y''(t) = f(t, y)$$

besitzt oder andere besondere Eigenschaften aufweist. Für die hier relevante Problemklasse ist dies nicht der Fall.

4.1 Kepler–Problem

Als erstes einfaches Testproblem wird die Bahnberechnung eines erdnahen Satelliten betrachtet, der sich auf einer idealen Keplerbahn bewegt. In kartesischen Koordinaten $p = (p_x, p_y, p_z)^T$ lauten die entsprechenden Differentialgleichungen

$$\begin{aligned} p_x'' &= -G_m p_x / r^3 \\ p_y'' &= -G_m p_y / r^3 \\ p_z'' &= -G_m p_z / r^3 \end{aligned}$$

mit

$$r = \sqrt{p_x^2 + p_y^2 + p_z^2} \ .$$

Als Parameterwert wird $G_m = 3.986009 \cdot 10^{14}$ gesetzt (Erdmasse). Als System 1. Ordnung erhält man

$$\begin{aligned} p_x' &= v_x \\ p_y' &= v_y \\ p_z' &= v_z \\ v_x' &= -G_m p_x / r^3 \\ v_y' &= -G_m p_y / r^3 \\ v_z' &= -G_m p_z / r^3 \ . \end{aligned}$$

In Kurzform läßt sich dieses System schreiben als

$$y' = f(t, y) \ , \ \ y \in \mathbb{R}^6 \ ,$$

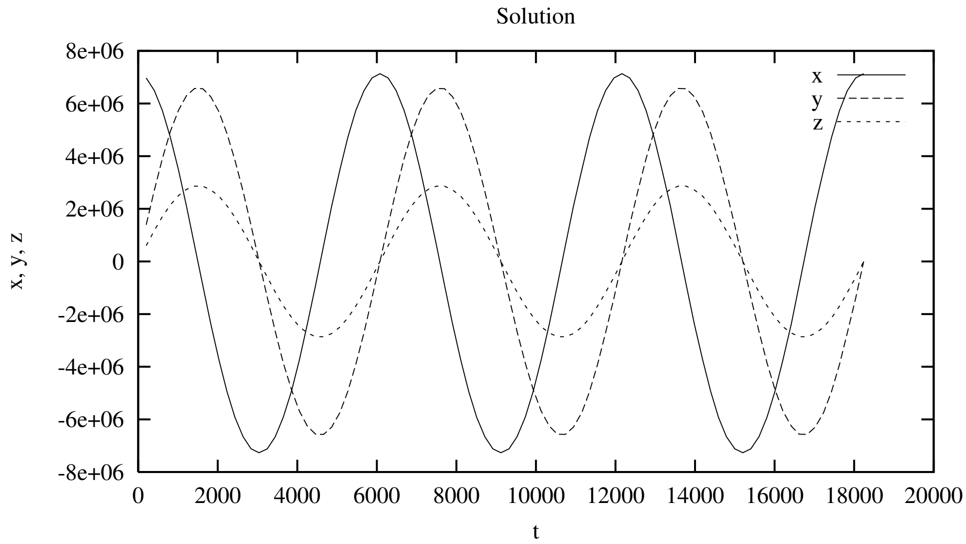


Abbildung 21: Lösung des einfachen Keplerproblems

die ersten 3 Komponenten werden mit dem Positionsvektor $p = (p_x, p_y, p_z)^T$, die letzten 3 Komponenten mit dem Geschwindigkeitsvektor $v = (v_x, v_y, v_z)^T$ identifiziert.
Um eine Keplerellipse mit Halbachse $a = 7200$ km und Exzentrizität $e = 0.01$ zu erhalten, werden folgende Anfangswerte gewählt:

$$p_x^0 = 7.128 \cdot 10^6 \ , \quad p_y^0 = 0 \ , \quad p_z^0 = 0 \ ,$$

$$v_x^0 = 0 \ , \quad v_y^0 = 6.890... \cdot 10^3 \ , \quad v_z^0 = 3.0 \cdot 10^3 \ .$$

Damit ergibt sich eine Umlaufzeit von $t_{Per} = 6080.08...$ Sekunden.
Als ersten Testfall (KEP1A) sei die numerische Integration über 3 Umläufe hinweg betrachtet. In Abb. 21 sind die berechneten Positionen p_x, p_y, p_z in einem Zeit–Ort–Diagramm dargestellt. Als Integrationsverfahren wurde die explizite Mittelpunktsregel mit Extrapolation verwendet – auch als Gragg–Bulirsch–Stoer–Verfahren bekannt. Es wird eine feste Zahl von Extrapolationsstufen (hier 2) und eine konstante Schrittweite vorgegeben. Durch zweifache Extrapolation besitzt das Verfahren (GBS3F) die lokale Fehlerordnung 7. Pro Umlauf werden 30 Integrationsschritte durchgeführt, so daß die Gesamtschrittzahl 90 beträgt, d.h. die Schrittweite ist etwa $H = 202$ Sekunden.

4.2 Fehlerkriterien

Zur Beurteilung der numerischen Integration wird im folgenden meist eine gewichtete l_2–Norm des „wahren“ Fehlers betrachtet. Zu einem Zeitpunkt t_j

seien die Komponenten $i = 1, \ldots, 6$ der numerischen Lösung $y_i^N(t_j)$ und einer sehr genauen bzw. exakten Referenzlösung $y_i^R(t_j)$ gegeben. Man definiert dann als Fehler zum Zeitpunkt t_j die Größe

$$\varepsilon(t_j) = \varepsilon_{Rms}(t_j) = ||y^N(t_j) - y^R(t_j)|| = \sqrt{\frac{1}{6}\sum_{i=1}^{6}\left(\frac{y_i^N(t_j) - y_i^R(t_j)}{\max(|y_i^R(t_j)|, y_i^S(t_j))}\right)^2} \,. \tag{11}$$

Hierbei sind die $y_i^S(t_j)$ geeignet zu wählende Schwellwerte für eine Fehlerbetrachtung im relativen Sinn. Mit der Wahl

$$y_i^S = \max(|p_x^0|, |p_y^0|, |p_z^0|) \;, \quad i = 1, 2, 3$$

und

$$y_i^S = \max(|v_x^0|, |v_y^0|, |v_z^0|) \;, \quad i = 4, 5, 6 \;,$$

für alle betrachteten Zeitpunkte, liefert diese Norm Werte, die sehr nahe an problemorientierten Fehlerkriterien, wie etwa der relativen Positions– bzw. Geschwindigkeitsabweichung liegen:

$$\varepsilon_P(t_j) = \sqrt{\frac{\Delta p_x^2 + \Delta p_y^2 + \Delta p_z^2}{p_x^2 + p_y^2 + p_z^2}}$$

bzw.

$$\varepsilon_V(t_j) = \sqrt{\frac{\Delta v_x^2 + \Delta v_y^2 + \Delta v_z^2}{v_x^2 + v_y^2 + v_z^2}} \;,$$

mit

$$\begin{aligned}
\Delta p_x &= y_1^N(t_j) - y_1^R(t_j) \;, \quad p_x = y_1^R(t_j) \\
\Delta p_y &= y_2^N(t_j) - y_2^R(t_j) \;, \quad p_y = y_2^R(t_j) \\
\Delta p_z &= y_3^N(t_j) - y_3^R(t_j) \;, \quad p_z = y_3^R(t_j) \\
\Delta v_x &= y_4^N(t_j) - y_4^R(t_j) \;, \quad v_x = y_4^R(t_j) \\
\Delta v_y &= y_5^N(t_j) - y_5^R(t_j) \;, \quad v_y = y_5^R(t_j) \\
\Delta v_z &= y_6^N(t_j) - y_6^R(t_j) \;, \quad v_z = y_6^R(t_j) \;.
\end{aligned}$$

In Abb. 22 sind dieses 3 Fehlermaße für die Integration des Problems mit der Mittelpunktsregel dargestellt. Offensichtlich liegen sie sehr nahe beieinander, so daß im folgenden sich auf die Darstellung des Fehlermaßes (11) beschränkt werden soll.

Um nur eine Fehlerkennzahl pro Integration zu erhalten, bildet man aus den Fehlern an allen berechneten Zeitpunkten t_j, $j = 1, \ldots, m$ den folgenden Mittelwert

$$\bar{\varepsilon} = \sqrt{\frac{1}{m}\sum_{j=1}^{m}\varepsilon(t_j)} \;.$$

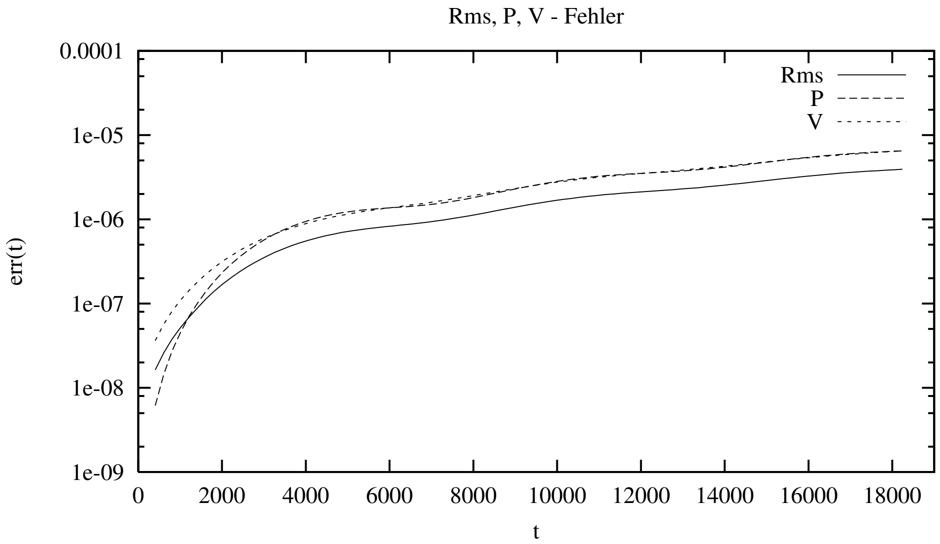


Abbildung 22: Fehler bei Lösung des Poblems KEP1A

4.3 Erster Vergleich

Für einen ersten Vergleich wird nun das Problem mit 3 state–of–the–art Integratoren analysiert. Das gemeinsames Kennzeichen dieser Verfahren ist eine lokale Fehlerkontrolle in dem Sinn, daß der Fehler eines Integrationsschrittes, in einer Fehlernorm entsprechend (11), durch automatische Anpassung der Schrittweite unter einer vorgegebenen Schranke gehalten wird. Es werden folgende Integratoren betrachtet:

- LSODE – Mehrschrittverfahren vom Adams–Typ mit Anpassung von Ordnung und Schrittweite, s. Hindmarsh [43, 44]
- ODEX – Extrapolationsverfahren basierend auf der expliziten Mittelpunktsregel mit Anpassung von Ordnung und Schrittweite, s. Bulirsch und Stoer [45] sowie Hairer, Nørset und Wanner [46]
- DOP853 – explizites Runge–Kutta–Verfahren mit Anpassung der Schrittweite und fester Ordnung, s. Dormand und Prince [47, 46].

Alle Verfahren sind in Fortran 77 implementiert und verwenden doppelt genaue Gleitkommazahlen, d.h. die relative Maschinengenauigkeit ist etwa $\varepsilon = 1 \cdot 10^{-16}$ (SUN Ultra 1 Creator). Die Ergebnisse, d.h. benötigter Aufwand gemessen in Anzahl der Integrationsschritte (Step) und Anzahl der Funktionsauswertungen (Fcn) und tatsächlich erreichte Genauigkeit $\bar{\varepsilon}$ in Abhängigkeit von der vorgegebenen Fehlerschranke *Rtol* sind in Tab. 9 zusammengefaßt.
Die Daten der Tab. 9 sowie die der Abb. 23–25 spiegeln die aus vielen Vergleichen bekannten allgemeinen Einschätzungen der Verfahrenstypen Mehr-

	LSODE			ODEX			DOP853		
Rtol	Step	Fcn	$\bar{\varepsilon}$	Step	Fcn	$\bar{\varepsilon}$	Step	Fcn	$\bar{\varepsilon}$
10^{-4}	59	99	$7.2 \cdot 10^{-2}$	14	376	$1.0 \cdot 10^{-2}$	17	257	$3.5 \cdot 10^{-4}$
10^{-5}	120	146	$1.5 \cdot 10^{-2}$	14	506	$1.3 \cdot 10^{-3}$	22	332	$3.7 \cdot 10^{-5}$
10^{-6}	168	175	$2.8 \cdot 10^{-3}$	16	650	$9.7 \cdot 10^{-5}$	28	422	$4.1 \cdot 10^{-6}$
10^{-7}	250	270	$5.2 \cdot 10^{-5}$	15	797	$1.3 \cdot 10^{-5}$	36	542	$4.7 \cdot 10^{-7}$
10^{-8}	313	326	$1.5 \cdot 10^{-5}$	15	979	$1.4 \cdot 10^{-6}$	46	692	$5.2 \cdot 10^{-8}$
10^{-9}	343	356	$2.3 \cdot 10^{-6}$	16	1168	$1.2 \cdot 10^{-7}$	60	902	$5.6 \cdot 10^{-9}$
10^{-10}	502	535	$2.0 \cdot 10^{-7}$	19	1377	$1.0 \cdot 10^{-8}$	78	1172	$5.9 \cdot 10^{-10}$
10^{-11}	602	614	$1.1 \cdot 10^{-8}$	18	1582	$1.2 \cdot 10^{-9}$	102	1532	$6.2 \cdot 10^{-11}$
10^{-12}	628	644	$3.7 \cdot 10^{-9}$	19	1797	$1.3 \cdot 10^{-10}$	135	2027	$6.4 \cdot 10^{-12}$
10^{-13}	769	815	$2.3 \cdot 10^{-11}$	21	2077	$9.0 \cdot 10^{-12}$	179	2687	$6.6 \cdot 10^{-13}$
10^{-14}	981	1028	$5.7 \cdot 10^{-12}$	25	2427	$2.0 \cdot 10^{-13}$	237	3557	$1.2 \cdot 10^{-13}$
10^{-15}	1145	1199	$1.2 \cdot 10^{-12}$	32	3044	$4.9 \cdot 10^{-13}$	314	4712	$1.1 \cdot 10^{-13}$
10^{-16}	1427	1524	$1.2 \cdot 10^{-13}$	52	4436	$9.8 \cdot 10^{-14}$	418	6272	$1.2 \cdot 10^{-13}$

Tabelle 9: Aufwandsvergleich I für Problem KEP1A

schrittverfahren, Runge–Kutta–Verfahren und Extrapolationsverfahren wieder. Das Mehrschrittverfahren LSODE ergibt vergleichsweise die geringste Genauigkeit bei gleichen Anforderungen, braucht aber auch die wenigsten Funktionsauswertungen, um eine spezifische Genauigkeit zu erreichen. Extrapolationsverfahren brauchen viele Funktionsauswertungen insbesondere für moderate verlangte/erreichte Genauigkeiten. Da jedoch die Zahl der Schritte sehr gering ist, ist der Overhead sehr klein und somit ist für „billige" Funktionen (wie hier) ODEX ab einer Genauigkeit von ca. 10^{-6} das schnellste Verfahren.
Bei einer etwas genaueren Analyse des Integrationsverhaltens zeigt sich weiterhin, daß alle Verfahren der sehr gleichförmigen Dynamik des Problems gerecht

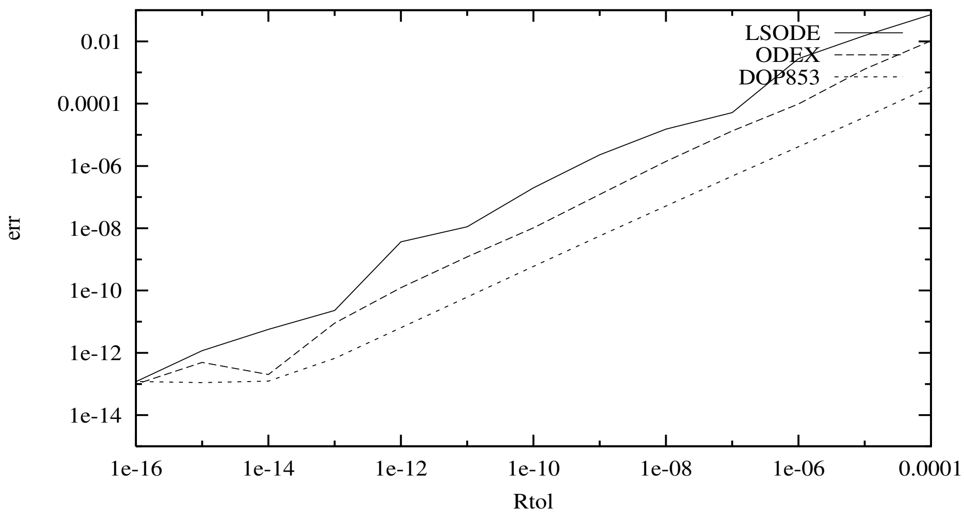


Abbildung 23: Erreichte Genauigkeit über verlangter Genauigkeit (DP)

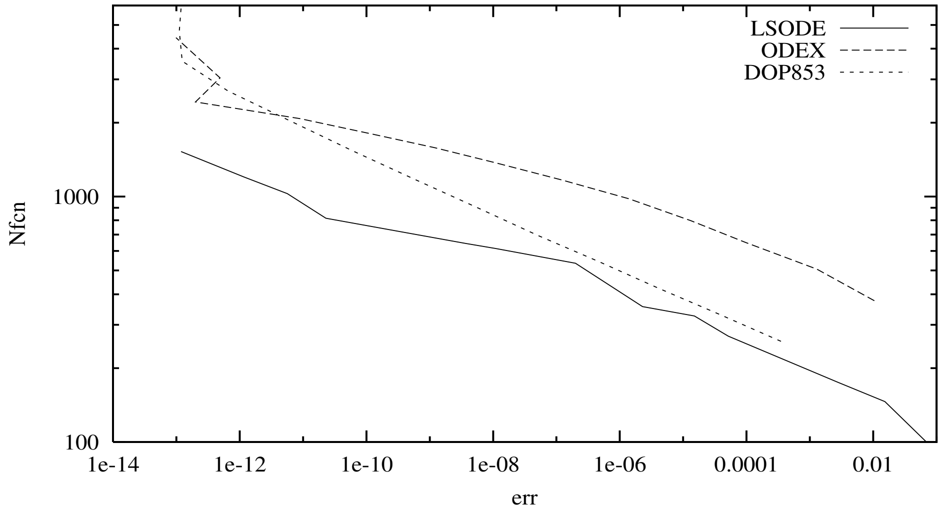


Abbildung 24: Anzahl F–Auswertungen über erreichter Genauigkeit (DP)

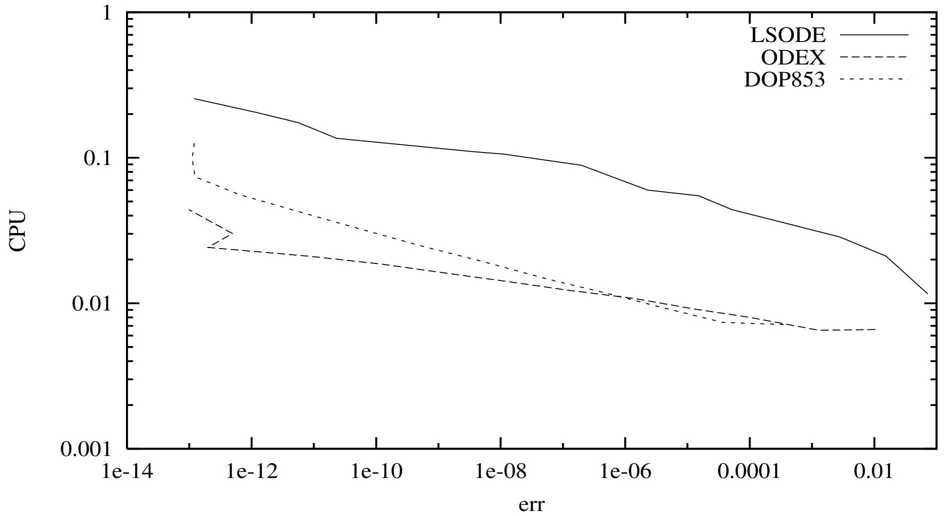


Abbildung 25: Benötigte CPU–Zeit über erreichter Genauigkeit (DP)

werden und sich nach einer kurzen Anlaufphase fast konstante Schrittweiten und Ordnungen einstellen. Für eine verlangte Genauigkeit $Rtol = 10^{-6}$ wählt DOP853 eine Schrittweite $H \approx 805$, LSODE wählt $H \approx 110$ und Ordnung $k \approx 4$, ODEX wählt $H \approx 1800$ bei $k \approx 7$. Bei $Rtol = 10^{-12}$ ergibt sich $H \approx 140$ (DOP853), $H \approx 30$, $k \approx 6$ (LSODE) und $H \approx 1480$, $k \approx 17$ (ODEX).

Offensichtlich ist bei einer Genauigkeit von ca. 10^{-13} die Grenze des mit doppelter Genauigkeit Möglichen erreicht. Um diese Schranke zu überwinden wird nun vierfache Genauigkeit verwendet. Der Testsatz umfaßt jetzt zusätzlich einige noch stringentere Fehlerschranken. Die Resultate sind in Tab. 10

	LSODE			ODEX			DOP853		
Rtol	Step	Fcn	$\bar{\varepsilon}$	Step	Fcn	$\bar{\varepsilon}$	Step	Fcn	$\bar{\varepsilon}$
10^{-4}	59	99	$7.2 \cdot 10^{-2}$	14	376	$1.0 \cdot 10^{-2}$	17	257	$3.5 \cdot 10^{-4}$
10^{-5}	120	146	$1.5 \cdot 10^{-2}$	14	506	$1.3 \cdot 10^{-3}$	22	332	$3.7 \cdot 10^{-5}$
10^{-6}	168	175	$2.8 \cdot 10^{-3}$	16	650	$9.7 \cdot 10^{-5}$	28	422	$4.1 \cdot 10^{-6}$
10^{-7}	250	270	$5.2 \cdot 10^{-5}$	15	797	$1.3 \cdot 10^{-5}$	36	542	$4.7 \cdot 10^{-7}$
10^{-8}	313	326	$1.5 \cdot 10^{-5}$	15	979	$1.4 \cdot 10^{-6}$	46	692	$5.2 \cdot 10^{-8}$
10^{-9}	343	356	$2.3 \cdot 10^{-6}$	16	1168	$1.2 \cdot 10^{-7}$	60	902	$5.6 \cdot 10^{-9}$
10^{-10}	502	535	$2.0 \cdot 10^{-7}$	19	1377	$1.0 \cdot 10^{-8}$	78	1172	$5.9 \cdot 10^{-10}$
10^{-11}	602	614	$1.1 \cdot 10^{-8}$	18	1582	$1.2 \cdot 10^{-9}$	102	1532	$6.2 \cdot 10^{-11}$
10^{-12}	628	644	$3.7 \cdot 10^{-9}$	19	1797	$1.2 \cdot 10^{-10}$	135	2027	$6.4 \cdot 10^{-12}$
10^{-13}	769	813	$1.7 \cdot 10^{-11}$	21	2077	$9.5 \cdot 10^{-12}$	179	2687	$6.6 \cdot 10^{-13}$
10^{-14}	943	1000	$7.6 \cdot 10^{-12}$	23	2379	$6.8 \cdot 10^{-13}$	237	3557	$6.7 \cdot 10^{-14}$
10^{-15}	1125	1161	$5.1 \cdot 10^{-13}$	26	2778	$4.6 \cdot 10^{-14}$	314	4712	$6.8 \cdot 10^{-15}$
10^{-16}	1436	1531	$3.6 \cdot 10^{-14}$	29	3189	$3.1 \cdot 10^{-15}$	417	6257	$6.9 \cdot 10^{-16}$
10^{-17}	1739	1861	$2.6 \cdot 10^{-15}$	33	3705	$2.0 \cdot 10^{-16}$	555	8327	$6.9 \cdot 10^{-17}$
10^{-18}	1985	2063	$3.1 \cdot 10^{-16}$	38	4302	$1.2 \cdot 10^{-17}$	738	11072	$6.9 \cdot 10^{-18}$
10^{-19}	2143	2233	$1.1 \cdot 10^{-16}$	43	4973	$6.8 \cdot 10^{-19}$	982	14732	$7.0 \cdot 10^{-19}$
10^{-20}	2865	3052	$4.1 \cdot 10^{-18}$	48	5728	$5.0 \cdot 10^{-20}$	1309	19637	$7.0 \cdot 10^{-20}$
10^{-21}	3829	4144	$1.8 \cdot 10^{-18}$	55	6631	$2.7 \cdot 10^{-21}$	1743	26147	$7.0 \cdot 10^{-21}$
10^{-22}	6938	7488	$1.5 \cdot 10^{-18}$	64	7762	$1.5 \cdot 10^{-22}$	2323	34847	$7.0 \cdot 10^{-22}$
10^{-23}	5133	5443	$7.8 \cdot 10^{-21}$	73	9023	$8.1 \cdot 10^{-24}$	3096	46442	$7.0 \cdot 10^{-23}$
10^{-24}	4370	4459	$9.5 \cdot 10^{-22}$	84	10442	$4.4 \cdot 10^{-25}$	4128	61922	$7.1 \cdot 10^{-24}$
10^{-25}	79864	89822	$2.2 \cdot 10^{-21}$	98	12214	$2.8 \cdot 10^{-26}$	5503	82547	$7.1 \cdot 10^{-25}$
10^{-26}	14319	15851	$3.3 \cdot 10^{-23}$	113	14183	$1.2 \cdot 10^{-27}$	7336	110042	$7.1 \cdot 10^{-26}$
10^{-27}	16915	18778	$5.5 \cdot 10^{-24}$	130	16442	$1.1 \cdot 10^{-28}$	9782	146732	$7.1 \cdot 10^{-27}$
10^{-28}	19300	21779	$9.7 \cdot 10^{-25}$	151	19181	$4.2 \cdot 10^{-29}$	13042	195632	$7.0 \cdot 10^{-28}$

Tabelle 10: Aufwandsvergleich II für Problem Kep1a

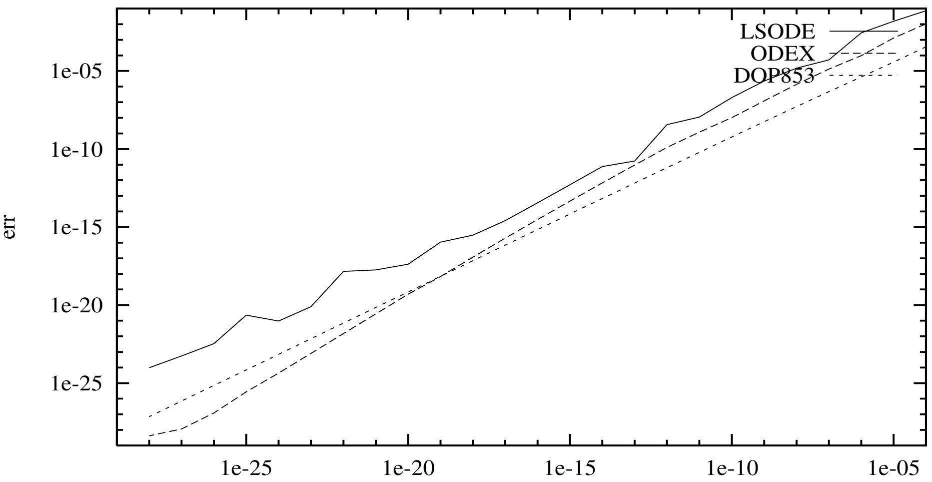


Abbildung 26: Erreichte Genauigkeit über verlangter Genauigkeit (QP)

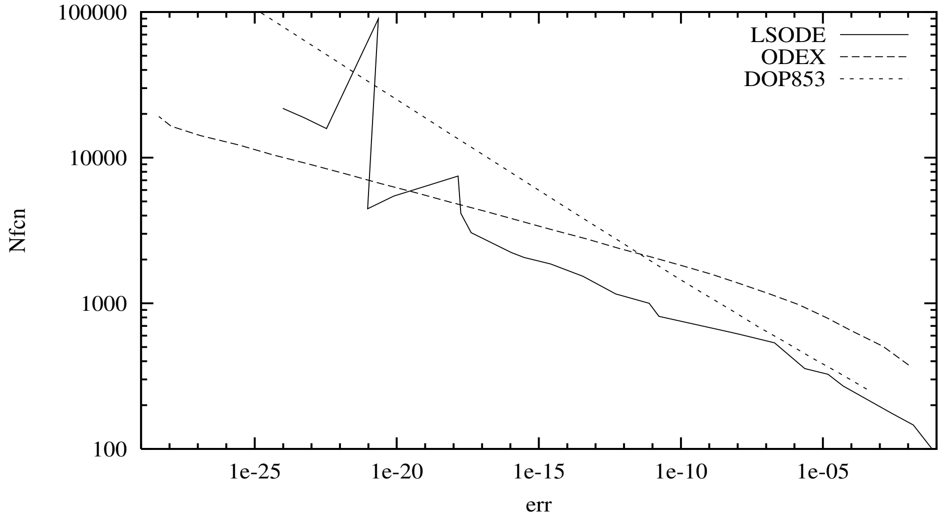


Abbildung 27: Anzahl F–Auswertungen über erreichter Genauigkeit (QP)

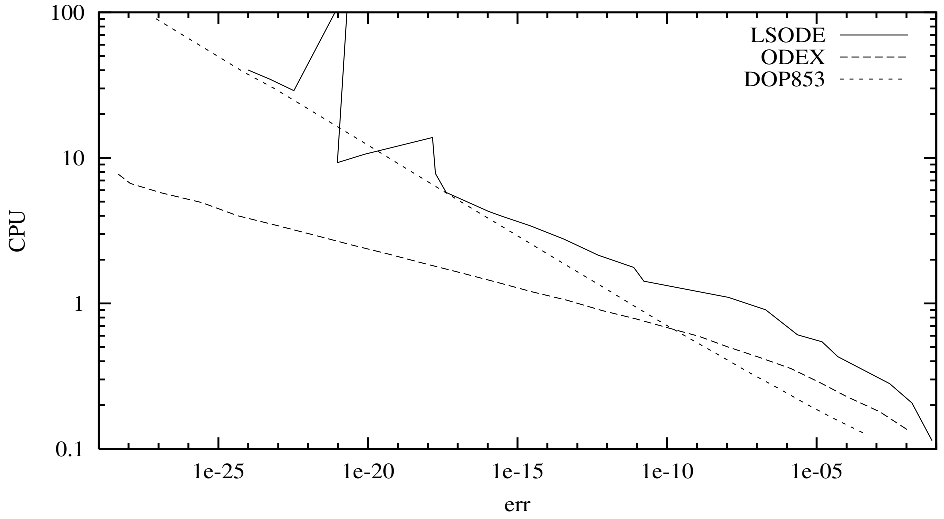


Abbildung 28: Benötigte CPU–Zeit über erreichter Genauigkeit (QP)

sowie den Abb. 24–26 zusammengefaßt. Dabei resultiert der Ausreißer von LSODE bei $Rtol = 10^{-25}$ aus einem „Festfressen“ der Ordnung bei $k = 6$ bzw. $k = 7$.
Für eine verlangte Genauigkeit $Rtol = 10^{-20}$ wählt DOP853 eine Schrittweite $H \approx 25$, LSODE wählt $H \approx 8$ und Ordnung $k \approx 8$, ODEX wählt $H \approx 590$ bei $k \approx 17$. Bei $Rtol = 10^{-24}$ ergibt sich $H \approx 4.4$ (DOP853), $H \approx 4.5$, $k \approx 9$ (LSODE) und $H \approx 230$, $k \approx 17$ (ODEX). Bei $Rtol = 10^{-28}$ ergibt sich $H \approx 1.4$ (DOP853) und $H \approx 122$, $k \approx 17$ (ODEX). Bei LSODE zeigen sich recht deutliche Schwankungen in Schrittweite und Ordnung. So ist $H \in [0.1, 3]$ und $k \in [7, 10]$.

	LSODE		ODEX		DOP853	
Rtol	DP	QP	DP	QP	DP	QP
10^{-4}	0.012	0.114	0.007	0.136	0.007	0.129
10^{-5}	0.021	0.207	0.007	0.180	0.007	0.162
10^{-6}	0.029	0.281	0.008	0.230	0.009	0.205
10^{-7}	0.044	0.430	0.009	0.285	0.012	0.264
10^{-8}	0.055	0.544	0.011	0.356	0.015	0.337
10^{-9}	0.060	0.608	0.012	0.426	0.019	0.439
10^{-10}	0.089	0.903	0.014	0.503	0.024	0.571
10^{-11}	0.106	1.100	0.016	0.592	0.032	0.746
10^{-12}	0.110	1.150	0.019	0.673	0.042	0.987
10^{-13}	0.136	1.420	0.021	0.780	0.056	1.310
10^{-14}	0.174	1.770	0.024	0.894	0.074	1.730
10^{-15}	0.205	2.140	0.030	1.050	0.099	2.300
10^{-16}	0.255	2.770	0.044	1.200	0.129	3.050
10^{-17}		3.420		1.400		4.050
10^{-18}		3.970		1.630		5.390
10^{-19}		4.290		1.890		7.180
10^{-20}		5.790		2.180		9.580
10^{-21}		7.820		2.530		12.800
10^{-22}		13.800		2.960		17.000
10^{-23}		10.600		3.450		22.700
10^{-24}		9.280		4.000		30.200
10^{-25}		150.000		4.940		38.900
10^{-26}		29.000		5.760		51.800
10^{-27}		34.600		6.670		69.200
10^{-28}		40.300		7.810		92.100

Tabelle 11: Vergleich der CPU–Zeiten für Kep1a

Bezüglich Genauigkeit und Effizienz ist nun ODEX deutlich das beste Verfahren. Daß das Verfahren mit fester, vergleichsweise niedriger Ordnung (DOP853) bei derartigen verlangten Genauigkeit ineffizient wird, ist zu erwarten, aber die Ergebnisse von LSODE überraschen doch. So ist nicht klar, warum die maximal mögliche Ordnung von $k = 12$ nicht erreicht wird.
In Tab. 11 sind die Rechenzeiten der Vergleichsläufe mit doppelter (DP) und vierfacher (QP) Genauigkeit zusammengestellt. Es verwundert, daß LSODE in QP nur etwa 10 mal langsamer als in DP ist, während bei ODEX der Faktor im Bereich von 33 und bei DOP853 im Bereich von 23 liegt. Trotzdem ist ODEX ab einer erreichten Genauigkeit von ca. 10^{-10} das schnellste Verfahren.

4.4 10 Perioden

Nun werde über 10 Umläufe hinweg integriert. In Abb. 29 ist das Fehlerverhalten der 3 Verfahren für eine verlangte Genauigkeit von $Rtol = 10^{-8}$ dargestellt. Dabei zeigt DOP853 ein hervorragendes Fehlerverhalten. Anscheinend kann dieses Verfahren von der Symmetrie des Problems profitieren. Die Fehler in der zweiten Hälfte eines jeden Umlaufs löschen bis dahin akkumulierte Fehler zum Teil wieder aus, so daß der Gesamtfehler ein wellenförmiges Ver-

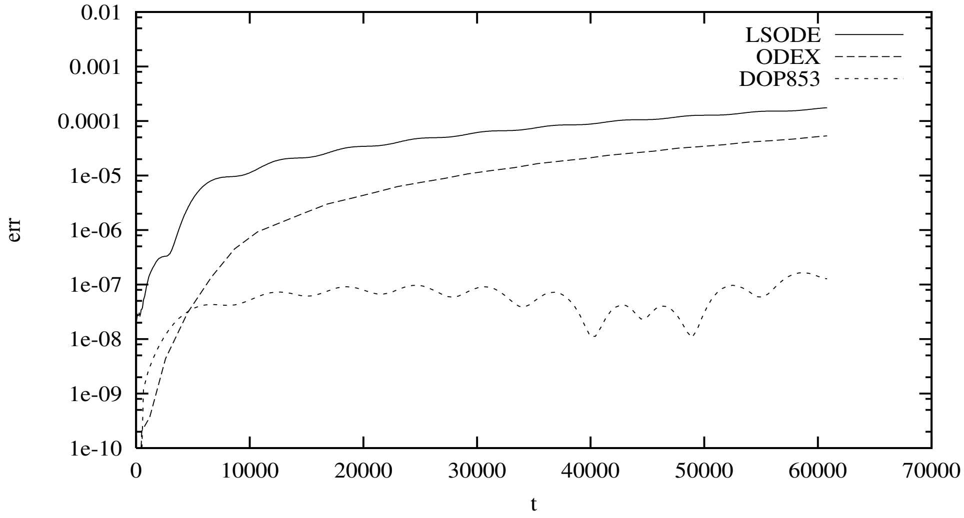


Abbildung 29: Integrationsfehler bei KEP1B (QP, Rtol=10^{-8})

halten zeigt. In geringem Maß tritt dies auch bei LSODE auf, bei ODEX praktisch nicht. Bei allen Verfahren ist allerdings zu erkennen, daß der Geauigkeitsverlust pro Umlauf im Laufe der Zeit etwas sinkt (negative 2. Ableitung der Fehlerkurve). Dies bedeutet allerdings nur, daß der Fehler kein in der Zeit exponentielles Wachstum aufweist. In Tab. 12 und 13 sind Aufwand und Genauigkeit einer ganzen Testreihe dargestellt. In Abb. 30, 31 und 32 sind die Resultate graphisch dargestellt.
Die Resultate des gleichen Experimentes mit $Rtol = 10^{-20}$ sind in Abb. 33 dargestellt. Hier ist das schlechte Abschneiden von LSODE auffällig.

	LSODE			ODEX			DOP853		
Rtol	Step	Fcn	$\bar{\varepsilon}$	Step	Fcn	$\bar{\varepsilon}$	Step	Fcn	$\bar{\varepsilon}$
10^{-8}	1032	1051	$8.5 \cdot 10^{-5}$	36	3036	$2.3 \cdot 10^{-5}$	141	2117	$7.0 \cdot 10^{-8}$
10^{-10}	1688	1759	$1.7 \cdot 10^{-6}$	48	4230	$1.4 \cdot 10^{-7}$	248	3722	$1.1 \cdot 10^{-9}$
10^{-12}	2002	2029	$4.1 \cdot 10^{-8}$	47	5487	$1.7 \cdot 10^{-9}$	439	6587	$1.6 \cdot 10^{-11}$
10^{-14}	3063	3226	$2.9 \cdot 10^{-11}$	62	7384	$9.2 \cdot 10^{-12}$	777	11657	$1.9 \cdot 10^{-13}$
10^{-16}	4662	4959	$4.0 \cdot 10^{-13}$	82	9978	$3.8 \cdot 10^{-14}$	1378	20672	$2.1 \cdot 10^{-15}$
10^{-18}	6475	6750	$3.8 \cdot 10^{-15}$	110	13568	$1.3 \cdot 10^{-16}$	2448	36722	$2.2 \cdot 10^{-17}$
10^{-20}	9249	9801	$4.7 \cdot 10^{-16}$	146	18370	$4.7 \cdot 10^{-19}$	4350	65252	$2.3 \cdot 10^{-19}$
10^{-22}	21877	23569	$2.0 \cdot 10^{-17}$	198	25048	$1.5 \cdot 10^{-21}$	7732	115982	$2.3 \cdot 10^{-21}$
10^{-24}	19618	20641	$5.3 \cdot 10^{-20}$	267	34049	$5.4 \cdot 10^{-24}$	13747	206207	$2.3 \cdot 10^{-23}$
10^{-26}	46807	51809	$4.9 \cdot 10^{-22}$	362	46274	$2.0 \cdot 10^{-26}$	24443	366647	$2.3 \cdot 10^{-25}$

Tabelle 12: Aufwandsvergleich für 10–Perioden–Problem KEP1B

Rtol	LSODE	ODEX	DOP853
10^{-8}	1.780	1.100	0.993
10^{-10}	3.000	1.550	1.740
10^{-12}	3.600	2.080	3.080
10^{-14}	5.700	2.810	5.460
10^{-16}	8.890	3.810	9.700
10^{-18}	12.700	5.200	17.200
10^{-20}	18.500	7.080	30.600
10^{-22}	43.200	9.690	54.900
10^{-24}	40.200	13.200	96.600
10^{-26}	95.300	18.100	172.000

Tabelle 13: CPU–Zeit–Vergleich für 10-Perioden–Problem KEP1B

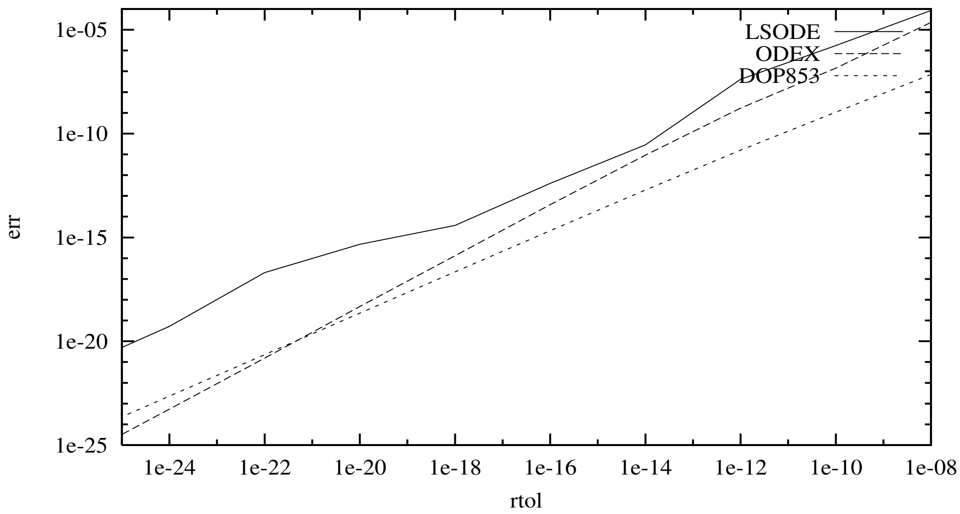


Abbildung 30: Erreichte Genauigkeit über verlangter Genauigkeit (QP)

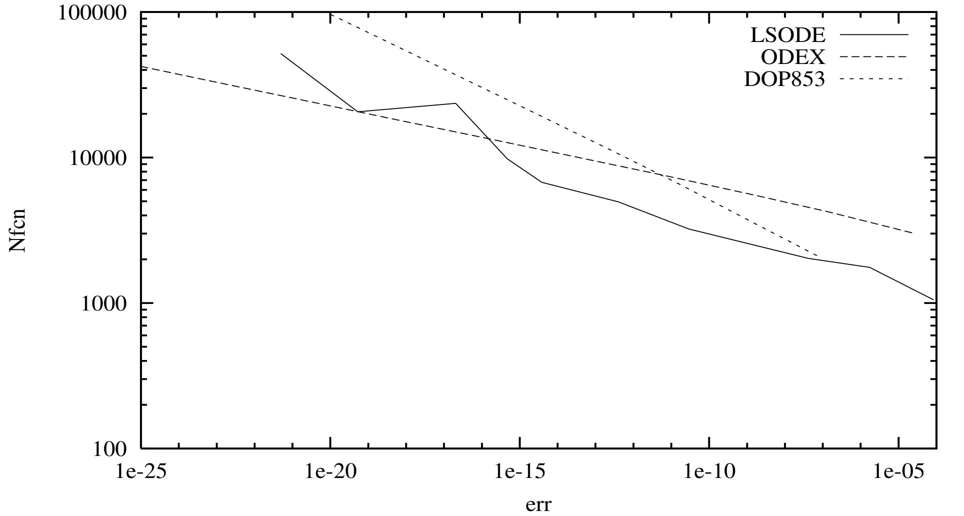


Abbildung 31: Anzahl F–Auswertungen über erreichter Genauigkeit (QP)

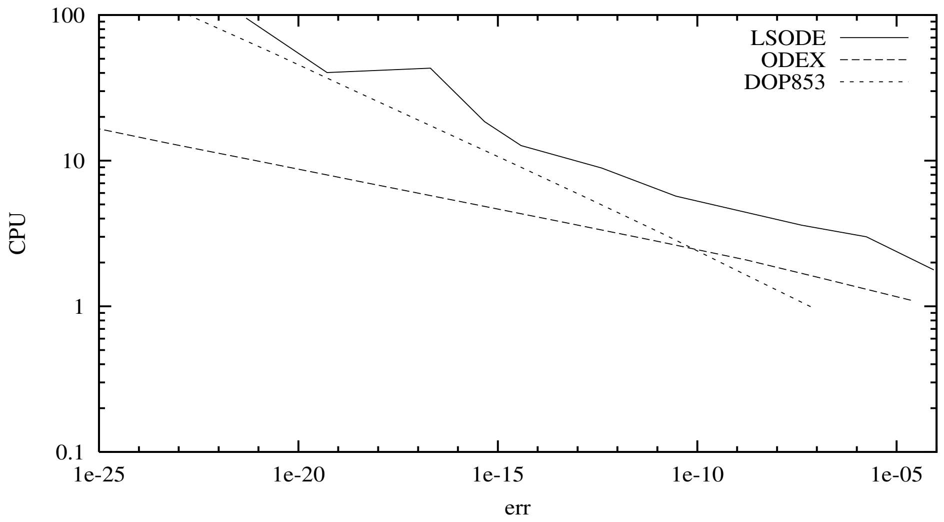


Abbildung 32: Benötigte CPU–Zeit über erreichter Genauigkeit (QP)

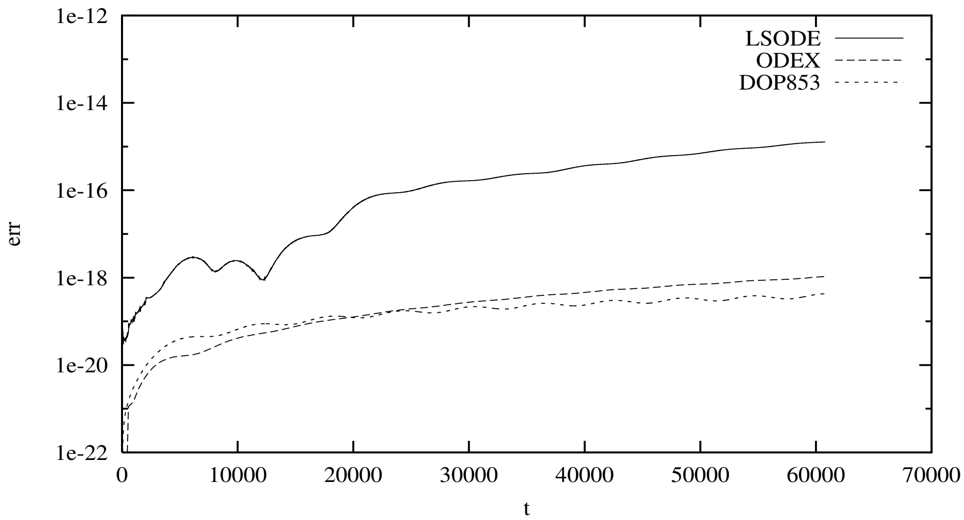


Abbildung 33: Integrationsfehler KEP1B (QP, Rtol=10^{-20})

4.5 Langzeitintegration

Im folgenden wird die Integration über 100 Perioden hinweg untersucht. Als Fehlerindikator wird nun nur der Fehler am Endpunkt der Integration betrachtet, d.h., siehe (11),

$$\varepsilon_{100} = \varepsilon(t_{end}) \ , \quad t_{end} = 100 \cdot t_{per} \ .$$

Für das vorliegende Testproblem ist dies ein adäquates Fehlermaß, da der maximale Integrationsfehler gerade an t_{end} auftritt. Aufwand und CPU–Zeiten sind in Tab. 14 und 15 dargestellt.

	LSODE			ODEX			DOP853		
Rtol	Step	Fcn	ε_{100}	Step	Fcn	ε_{100}	Step	Fcn	ε_{100}
10^{-8}	10252	10388	$1.1 \cdot 10^{-2}$	302	29418	$5.8 \cdot 10^{-3}$	1366	20492	$5.6 \cdot 10^{-5}$
10^{-10}	17008	17625	$2.6 \cdot 10^{-4}$	418	40860	$3.5 \cdot 10^{-5}$	2437	36557	$2.9 \cdot 10^{-7}$
10^{-12}	19663	19834	$9.6 \cdot 10^{-6}$	416	53032	$4.3 \cdot 10^{-7}$	4340	65102	$1.4 \cdot 10^{-9}$
10^{-14}	30576	32159	$2.9 \cdot 10^{-9}$	559	71475	$2.2 \cdot 10^{-9}$	7723	115847	$5.8 \cdot 10^{-12}$
10^{-16}	45658	48444	$1.7 \cdot 10^{-10}$	757	97131	$8.8 \cdot 10^{-12}$	13738	206072	$1.3 \cdot 10^{-14}$
10^{-18}	64257	66770	$1.8 \cdot 10^{-12}$	1032	132584	$3.0 \cdot 10^{-14}$	24434	366512	$1.2 \cdot 10^{-16}$
10^{-20}	94515	100919	$1.4 \cdot 10^{-13}$	1408	181138	$1.0 \cdot 10^{-16}$	43454	651812	$2.6 \cdot 10^{-18}$
10^{-22}	205406	221491	$3.2 \cdot 10^{-15}$	1921	247289	$3.4 \cdot 10^{-19}$	77278	1159172	$3.4 \cdot 10^{-20}$
10^{-24}	181170	188728	$7.7 \cdot 10^{-18}$	2617	337143	$1.3 \cdot 10^{-21}$	137425	2061377	$3.9 \cdot 10^{-22}$
10^{-26}	461511	510028	$1.3 \cdot 10^{-19}$	3563	459203	$5.3 \cdot 10^{-24}$	244383	3665747	$4.1 \cdot 10^{-24}$

Tabelle 14: Aufwandsvergleich für 100–Perioden–Problem KEP1C

Rtol	LSODE	ODEX	DOP853
10^{-8}	11.000	10.800	8.770
10^{-10}	19.500	15.100	15.700
10^{-12}	22.900	20.400	27.800
10^{-14}	37.200	27.600	49.600
10^{-16}	58.300	37.600	88.200
10^{-18}	86.600	51.500	157.000
10^{-20}	131.000	70.600	279.000
10^{-22}	279.000	96.600	494.000
10^{-24}	261.000	132.000	878.000
10^{-26}	653.000	181.000	1560.00

Tabelle 15: CPU–Zeit–Vergleich für 100-Perioden–Problem KEP1C

Im Vergleich von erreichter zu verlangter Genauigkeit verliert LSODE zwischen 6 und 7 Dezimalstellen, ODEX zwischen 2 und 5 und DOP853 zwischen 2 und 3. Offensichtlich ist der Genauigkeitsverlust abhängig vom Typ des Verfahrens, von der spezifizierten lokalen Genauigkeit und der Integrationsdauer. Für das betrachtete Testproblem (und ähnliche Probleme) verhält sich der Gesamtfehler $\varepsilon(t)$ zum Zeitpunkt t gemäß

$$\varepsilon(t) = c(Rtol, Methode) \cdot Rtol \cdot t^2 \ . \tag{12}$$

Zur Illustration sind in Tab. 16 die Resultate einer Integration mit GBS9 über 1000 Perioden hinweg angegeben. Bei GBS9 handelt es sich um einen selbstentwickelten Code, der auf dem Gragg–Bulirsch–Stoer–Verfahren basiert. Allerdings wird die Ordnung fest gehalten (hier $k = 19$, d.h. 8–fache Extrapolation), während die Schrittweite mittels einer lokalen Fehlerkontrolle im Sinn von DEUFLHARD [48] automatisch adaptiert wird. Die Fehler ε_3, ε_{10}, ε_{100} und ε_{1000} bezeichnen den jeweiligen Fehler nach 3, 10, 100 bzw. 1000 Perioden.

Zur Überprüfung des Fehlermodells (12) werden die Quotienten

$$q_{k-1,k} = \frac{\varepsilon_k}{\varepsilon_{k-1}} \cdot (\frac{t_k}{t_{k-1}})^2, \ \{k, k-1\} \in \{3, 10, 100, 1000\}$$

Rtol	CPU	Nstep	Nfcn	ε_3	ε_{10}	ε_{100}	ε_{1000}
1.00E-12	72.3	3899	596547	1.59E-10	1.86E-09	1.90E-07	1.91E-05
1.00E-16	125.0	6733	1030149	4.88E-15	5.36E-14	5.33E-12	5.33E-10
1.00E-20	214.0	11522	1763019	2.41E-19	2.42E-18	2.31E-16	2.30E-14

Tabelle 16: Aufand und Genauigkeit von GBS9 für Problem KEP1D

gebildet. Bei exakter Gültigkeit des Fehlermodells müßte für diese Quotienten $q_{k-1,k} = 1$ gelten. Die Werte in Tab. 17 belegen, daß sich der Fehler von GBS9 tatsächlich gemäß (12) verhält. Für DOP853 ergeben sich ebenfalls Quotienten nahe bei 1. Das Verfahren ODEX hat etwas mehr streuende Werte, bei LSODE ist dies noch deutlicher. Dies mag in der variablen Ordnung dieser Verfahren begründet sein.

Rtol	$q_{3,10}$	$q_{10,100}$	$q_{100,1000}$
1.00E-12	1.05	1.02	1.02
1.00E-16	0.99	0.99	0.99
1.00E-20	0.90	0.96	0.96

Tabelle 17: Qualität des Fehlermodells für Problem KEP1D

In analoger Weise kann man nun Quotienten der verfahrens– und genauigkeitsabhängigen Konstanten c in (12) berechnen. Auch hier erhält man Resultate, die diese Abhängigkeit belegen und gleichfalls die Unabhängigkeit von t bestätigen.

Für die Praxis ist nun wichtig zu wissen, welche Genauigkeit sich mit den vorgestellten Integratoren in der „Standardgenauigkeit“ double precision bei Rechnungen über längere Zeiträume erzielen läßt. Hierfür wird nochmals das Problem KEP1C, d.h. 100 Umläufe oder ca. 8 Tage Simulationszeit herangezogen. Die Ergebnisse der Integration mit verschiedenen vorgegebenen Fehlerschranken sind in Tab. 18 zusammengefaßt.

	LSODE			ODEX			DOP853		
Rtol	CPU	Fcn	ε_{100}	CPU	Fcn	ε_{100}	CPU	Fcn	ε_{100}
10^{-8}	0.19	10383	$1.1 \cdot 10^{-2}$	0.23	29418	$5.8 \cdot 10^{-3}$	0.08	20492	$5.6 \cdot 10^{-5}$
10^{-10}	0.33	17375	$3.2 \cdot 10^{-4}$	0.32	40860	$3.5 \cdot 10^{-5}$	0.14	36557	$2.9 \cdot 10^{-7}$
10^{-12}	0.39	19834	$9.6 \cdot 10^{-6}$	0.43	53032	$4.3 \cdot 10^{-7}$	0.24	65102	$1.4 \cdot 10^{-9}$
10^{-14}	0.65	32273	$2.7 \cdot 10^{-9}$	0.58	71498	$2.1 \cdot 10^{-9}$	0.43	115847	$2.5 \cdot 10^{-11}$
10^{-16}	1.00	48118	$1.9 \cdot 10^{-10}$	0.83	102583	$2.1 \cdot 10^{-10}$	0.77	206087	$1.0 \cdot 10^{-11}$

Tabelle 18: Aufwandsvergleich für KEP1C in DP

Sie zeigen, daß man zwar unter optimalen Umständen, eine Genauigkeit von 11 Dezimalstellen erreichen kann, i.a. aber von bestenfalls 10 Stellen ausgehen sollte. Wie gut die Verfahren implementiert sind zeigt die Tatsache, daß die Resultate der DP–Versionen mit denen der QP–Versionen bei verlangten Genauigkeit von 10^{-8}–10^{-14} im wesentlichen übereinstimmen.

Somit sind in doppelt genauer Rechnung über einen Simulationszeitraum von ca. 1 Woche etwa 10 Stellen als maximale Genauigkeit zu erreichen. Bei Positionskoordinaten von ca. 10^4 km entspricht dies einer absoluten Genauigkeit von $10^{-3}m = 1$ mm. Verlängert bzw. verringert man die Integrationsdauer um einen Faktor 10, so verliert bzw. gewinnt man 2 Stellen. Dies gilt natürlich nur für das hier betrachtete einfache Testproblem und die untersuchten Integratoren.
Bereits ein einfaches Modellproblem für die Bahnberechnung von Satelliten zeigt also, daß auch bei Verfahren mit lokaler Fehlerkontrolle der globale Fehler stark anwachsen kann, wobei dessen Größe stark verfahrensabhängig ist. Wünschenswert sind daher Methoden, mit denen der globale Fehler zumindest abgeschätzt werden kann.

4.6 Globale Fehlerschätzung

Obwohl Untersuchungen zur globalen Fehlerfortpflanzung eine lange Tradition haben, s. z.B. Zadunaisky [49], und verschiedene Methoden zur seiner Schätzung vorgeschlagen wurden, etwa Skeel [50], haben derartige Techniken keinen Einzug in gängige public domain Codes zur Lösung von gewöhnlichen Differentialgleichungen (etwa LSODE, DOP853, ODEX, ...) gehalten. Die wohl einfachste „Methode“ zur Abschätzung des globalen Fehlers besteht darin, das Problem mit dem gleichen oder einem anderen Verfahren genauer zu lösen und die Lösungsdifferenz als Approximation des globalen Fehlers anzusehen. Neben einigen technischen Problemen (Abspeichern der Lösung, i.a. unterschiedliche Integrationspunkte, ...) ist es dabei unbefriedigend, daß ein Abbruch der Integration aufgrund eines zu großen globalen Fehlers während der Rechnung selber nicht möglich ist. Im folgenden soll eine einfache Möglichkeit der globalen Fehlerschätzung vorgeschlagen und ihre Robustheit und Effizienz geprüft werden.
Dazu betrachte man wieder das GBS–Verfahren mit lokaler Fehlerkontrolle, adaptiver Schrittweite und fester Ordnung. In diesem Verfahren werden in jedem Schritt k verschiedene Approximationen $T_{i,1}$, $i = 1, \ldots, k$ an die wahre Lösung $y(t_j)$ berechnet. Durch Extrapolation werden weitere, genauere Approximationen bestimmt. Durch Vergleich der genauesten ($T_{k,k}$) mit der zweitbesten ($T_{k,k-1}$) Approximation schätzt man den lokalen Fehler und leitet einen Schrittweitenvorschlag für den nächsten Schritt ab. Als Startwert im nächsten Schritt wird $T_{k,k}$ verwendet.
Zur globalen Fehlerschätzung wird das Differentialgleichungssystem simultan zu der eben dargestellten Integration nochmals mit der gleichen Technik gelöst. Man extrapoliert aber eine Stufe weiter, d.h. man berechnet, ausgehend von den $k+1$ Approximationen $\bar{T}_{i,1}$, $i = 1, \ldots, k+1$ eine noch genauere Approximation $\bar{T}_{k+1,k+1}$. Der Vergleich von $T_{k,k}$ mit $\bar{T}_{k+1,k+1}$ liefert einen globalen Fehlerschätzer für $T_{k,k}$. Für die Weiterintegration der zusätzlichen Lösung wird dann $\bar{T}_{k+1,k+1}$ verwendet.

	GBS5G				
Rtol	CPU	Nstep	Fcn	ε_{100}^{true}	ε_{100}^{est}
10^{-6}	0.09	403	49264	$1.92 \cdot 10^{-1}$	$1.90 \cdot 10^{-1}$
10^{-8}	0.14	637	77812	$5.23 \cdot 10^{-4}$	$5.19 \cdot 10^{-4}$
10^{-10}	0.22	990	120878	$1.19 \cdot 10^{-6}$	$1.18 \cdot 10^{-6}$
10^{-12}	0.33	1516	185050	$3.18 \cdot 10^{-9}$	$3.15 \cdot 10^{-9}$
10^{-13}	0.41	1873	228604	$2.06 \cdot 10^{-10}$	$1.99 \cdot 10^{-10}$
10^{-14}	0.50	2312	228604	$9.27 \cdot 10^{-12}$	$2.02 \cdot 10^{-11}$
	GBS5G				
10^{-6}	0.10	219	56712	$6.90 \cdot 10^{-1}$	$6.78 \cdot 10^{-1}$
10^{-8}	0.13	305	78900	$3.30 \cdot 10^{-3}$	$3.27 \cdot 10^{-3}$
10^{-10}	0.18	410	105990	$2.56 \cdot 10^{-5}$	$2.55 \cdot 10^{-5}$
10^{-12}	0.24	562	142206	$1.19 \cdot 10^{-7}$	$1.19 \cdot 10^{-7}$
10^{-13}	0.28	562	169974	$7.17 \cdot 10^{-9}$	$7.15 \cdot 10^{-9}$
10^{-14}	0.33	770	198870	$4.27 \cdot 10^{-10}$	$4.44 \cdot 10^{-10}$

Tabelle 19: Globale Fehlerschätzung für KEP1C in DP

In einem ersten Test wird der globale Fehlerschätzer auf das Problem KEP1C angewendet. Die Resultate für 2 typische Vertreter (GBS5G und GBS7G) sind in Tab. 19 dargestellt. Bei beiden Varianten stimmen Fehlerschätzung und wahrer Fehler hervorragend überein. Noch stringentere Toleranzen als 10^{-14} können nicht realisiert werden, da dann die zusätzliche Extrapolationsstufe für den globalen Fehlertest keinen Genauigkeitsgewinn mehr liefert. Im Vergleich zu entsprechenden Verfahren ohne globalen Fehlerschätzer wächst der Aufwand um einen Faktor von knapp 2.5.

4.7 Das restringierte Dreikörperproblem

Als Modell für eine kritische Satellitenbahnberechnung soll das sog. restringierte Dreikörperproblem dienen. Es beschreibt die Bewegung von drei Körpern unter dem Einfluß ihrer gegenseitigen Massenanziehung, wobei folgende Vereinfachungen gemacht werden. Zwei Massen m_1, m_2 bewegen sich auf Kreisbahnen um ihren gemeinsamen Schwerpunkt, die dritte Masse m_3 sei verschwindend klein und alle Massen bewegen sich in einer Ebene.

In einem mitbewegten kartesischen (ξ, η)–Koordinatensystem (die Massen m_1, m_2 befinden sich auf der ξ–Achse in Position $-\mu$ bzw. $1-\mu$) ergeben sich die Bewegungsgleichungen für die Koordinaten der dritten Masse, siehe etwa DEUFLHARD UND BORNEMANN [51], zu

$$\begin{aligned}
\xi'' &= 2\eta' + \xi - \bar{\mu}(\xi - \mu)\rho - \mu(\xi - \bar{\mu})\bar{\rho} \\
\eta'' &= -2\xi' + \eta - \bar{\mu}\eta\rho - \mu\eta\bar{\rho}
\end{aligned}$$

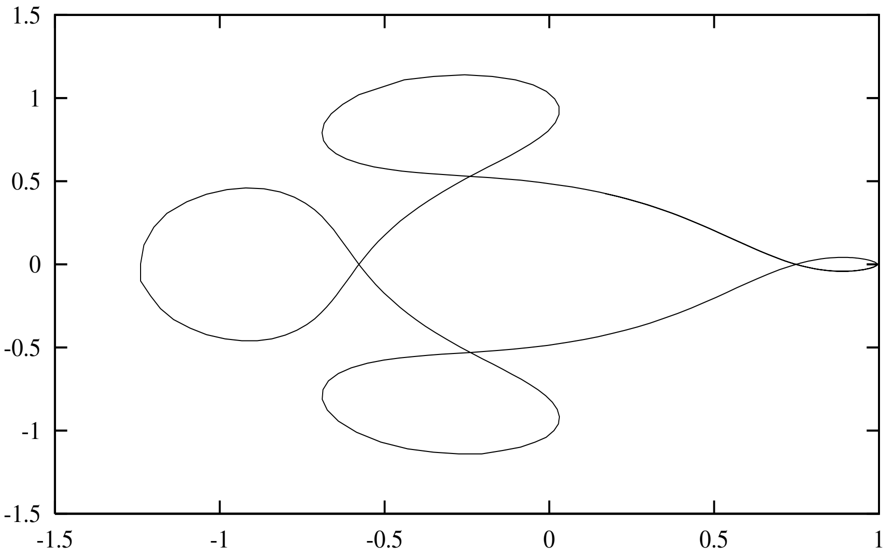


Abbildung 34: Eine periodische Bahn des Dreikörperproblems

mit

$$\mu = \frac{m_2}{m_1+m_2} \ , \quad \bar{\mu} = 1 - \mu \ ,$$

$$\rho = ((\xi + \mu)^2 + \eta^2)^{-3/2} \ , \quad \bar{\rho} = ((\xi - \bar{\mu})^2 + \eta^2)^{-3/2} \ .$$

Bei geeignet gewählten Anfangswerten erhält man periodische Bahnen unterschiedlichen Typs. Für die Werte $\mu = 0.01227747$ (Mond–Erde–Problem) und

$$\xi_0 = 0.999\ldots \ , \quad \eta_0 = 1.296\ldots \cdot 10^{-7} \ ,$$

$$\xi_0' = 2.044\ldots \cdot 10^{-5} \ , \quad \eta_0' = -2.001\ldots \ .$$

ergibt sich eine vierschlaufige periodische Bahn mit eine Umlaufzeit von $t_{Per} \approx 17$, vgl. Abb. 34.
Im folgenden soll das Problem über etwas mehr als 4 Perioden hinweg betrachtet werden, d.h. $t_{end} = 69$ (Problem RTB). Zunächst vergleichen wir wieder die 3 Standardverfahren (DP–Versionen) bei verschiedenen vorgegebenen Genauigkeitsforderungen. Die Resultate sind in Tab. 20 angegeben. Dabei bezeichnet ε_{max} den maximalen Fehler bzgl. einer genaueren Referenzlösung. Dieser Wert tritt nicht an t_{end} auf, sondern kurz vorher. Die besondere numerische Schwierigkeit dieses Problems zeigt sich darin, daß selbst bei einer Genauigkeitsforderung von $Rtol = 10^{-10}$ der wahre (maximale) relative Fehler im Bereich von 1 liegt. In den Abb. 35–37 sind die numerischen Lösungen von DOP853 für die Toleranzen 10^{-8}, 10^{-9} und 10^{-10} dargestellt. Erst bei

	LSODE			ODEX			DOP853		
Rtol	CPU	Fcn	ε_{max}	CPU	Fcn	ε_{max}	CPU	Fcn	ε_{max}
10^{-8}	0.37	2718	$2.5 \cdot 10^{0}$	0.20	10672	$2.0 \cdot 10^{0}$	0.17	8219	$1.0 \cdot 10^{0}$
10^{-10}	0.64	4572	$1.0 \cdot 10^{0}$	0.35	19560	$7.4 \cdot 10^{-1}$	0.25	13080	$7.4 \cdot 10^{-1}$
10^{-12}	1.11	7846	$3.6 \cdot 10^{-1}$	0.71	39724	$2.8 \cdot 10^{-2}$	0.43	21437	$2.7 \cdot 10^{-2}$
10^{-14}	1.65	11592	$1.4 \cdot 10^{-1}$	4.93	244205	$1.2 \cdot 10^{-2}$	0.70	33424	$1.1 \cdot 10^{-3}$

Tabelle 20: Aufwands- und Genauigkeitsvergleich für Problem RTB in DP

$Rtol = 10^{-12}$ liegen für alle Verfahren die 4 Umläufe optisch übereinander. Bei einer verlangten Genauigkeit von 10^{-14} liefert LSODE eine kaum genauere Lösung, bei ODEX bricht die Schrittweitensteuerung zusammen während DOP853 immerhin eine auf 3 Dezimalstellen korrekte Lösung abliefert. Eine weitere Periode kann aber auch mit diesem Verfahren nicht mehr integriert werden.
In einem letzten Experiment wird nun geprüft, wie sich das mit einem globalen Fehlerschätzer ausgestattete Verfahren GBS5G verhält. Dabei wird die Integration automatisch abgebrochen, wenn der geschätzte globale Fehler größer als 0.1 wird. In Tab. 21 sind die Resultate zusammengefaßt. Offensichtlich arbeitet der Fehlerschätzer auch hier sehr gut, selbst in dem kritischen Bereicht $Rtol = 10^{-14}$.

	GBS5G			
Rtol	Fcn	t_{stop}	ε_{max}^{true}	ε_{max}^{est}
10^{-8}	22952	51.19	0.32	0.32
10^{-10}	37527	62.00	0.11	0.11
10^{-12}	62360		$1.8 \cdot 10^{-2}$	$1.7 \cdot 10^{-2}$
10^{-14}	84523		$1.5 \cdot 10^{-2}$	$2.0 \cdot 10^{-2}$

Tabelle 21: Globale Fehlerschätzung für RTB mit GBS5G

Zum Abschluß sei noch bemerkt, daß das entsprechende Verfahren ohne globalen Fehlerschätzer (GBS5) bei einer verlangten Genauigkeit von $Rtol = 10^{-12}$ das Problem in 462 Schritten bei 31 262 Funktionsauswertungen mit einem Maximalfehler von $1.8 \cdot 10^{-2}$ löst. Verwendet man dieses Verfahren ohne Schrittweitenadaptierung, d.h. mit konstanter, vorgegebener Schrittweite (GBS5F), so benötigt man 35 000 Schritte mit 1 715 000 Funktionsauswertungen um einen Maximalfehler von $2.5 \cdot 10^{-2}$ zu erreichen.

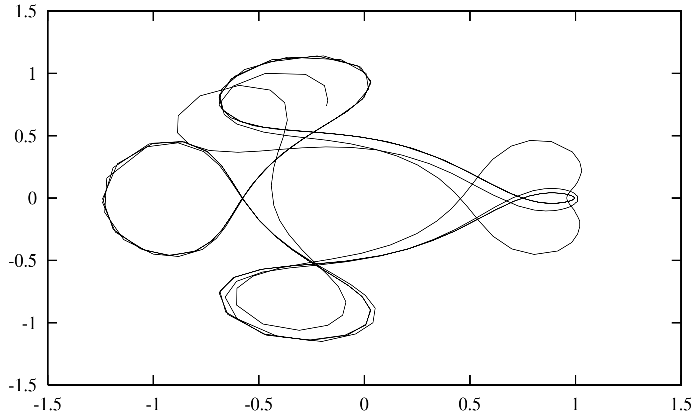


Abbildung 35: Problem RTB, DOP853, Rtol=10^{-8}

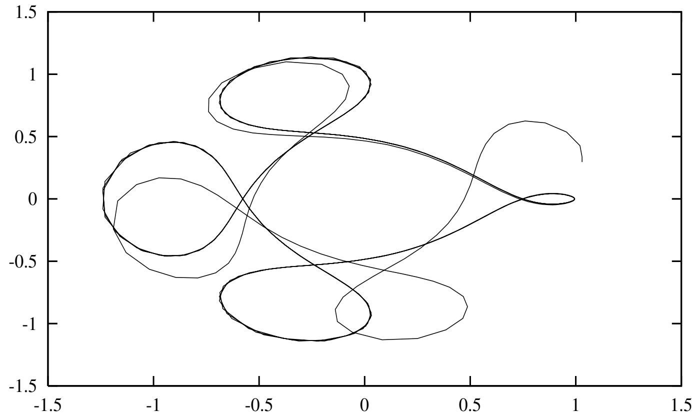


Abbildung 36: Problem RTB, DOP853, Rtol=10^{-9}

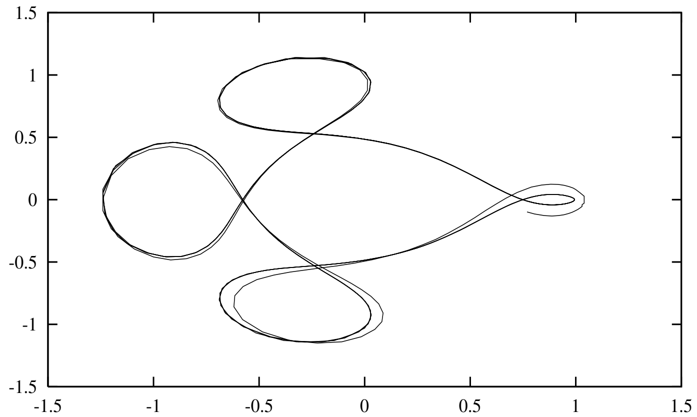


Abbildung 37: Problem RTB, DOP853, Rtol=10^{-10}

4.8 Zusammenfassung

Es hat sich gezeigt, daß unter den gegebenen Umständen, wie etwa einer Rechnung in kartesischen Koordinaten, selbst einfachste Bahnberechnungen im Langzeitverhalten eine deutliche Fehlerakkumulation aufweisen. Dies wird bei komplexeren Modellen mit realitätsnahen Schwerefeldmodellen, Berücksichtigung von Reibung, Sonnenwind usw. ebenso der Fall sein. Bei Simulationen für einen Zeitraum von etwa einer Woche ist demnach die Position eines Satelliten bestenfalls auf 1 mm genau zu bestimmen. Im Vergleich verschiedener Integrationsverfahren zeigen sich zwar einerseits deutliche Unterschiede, so etwa in der Größe des Genauigkeitsverlusts bei gleicher vorgegebener Fehlerschranke. Vergleicht man dagegen den Aufwand in Relation zur erzielten Genauigkeit, so erhält man für die Verfahren eine über weite Genauigkeitsbereiche ähnliche Effizienz.
Um zu gesicherten Aussagen über die wahre erzielte Genauigkeit zu gelangen, ist eine sichere Schätzung des globalen Fehlers notwendig. Hierzu wurde ein einfach realisierbares, robust arbeitendes und nicht zu aufwendiges Verfahren vorgestellt.

Literatur

[1] J.J. Dongarra, J. DuCroz, S. Hammarling, R. Hanson: *An Extended Set of Fortran Basic Linear Algebra Subprograms.* ACM Trans. on Math. Soft. **14**, pp. 1–32 (1988).
http://cm.bell-labs.com/netlib/blas/index.html

[2] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. DuCroz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostruchov, D. Sorensen: *LAPACK Users' Guide.*, Third edition. SIAM, Philadelphia 1999.
http://www.netlib.org/lapack/lug

[3] *The NAG Fortran Library Manual, Mark 19.* The Numerical Algorithms Group Limited, Oxford (1999).
http://www.nag.com/numeric/FLOLCH.html

[4] *IMSL Fortran Numerical Libraries.* Visual Numerics Inc., Houston 1999.
http://www.vni.com/products/imsl/imslfort.html

[5] *Engineering and Scientific Subroutine Library Products, General Information.* IBM (1998).
http://www.rs6000.ibm.com/resource/aix_resource/sp_books/essl/gim/gim1198/esslgim.html

[6] M.J. Daydé, I.S. Duff: *A Blocked Implementation of Level 3 BLAS for RISC Processors.* ENSEEIHT–IRIT Technical Report, RT/APO/97/2 (1997).

[7] *SUN Performance Workshop Fortran,* SUN (1999).
http://www.sun.com/workshop/fortran/spec.html

[8] R.C. Whaley, J.J. Dongarra: *Automatically Tuned Linear Algebra Software*, 1999.
http://www.netlib.org/atlas/

[9] Å. Björck: *Numerical Methods for Least Squares Problems.* SIAM, Philadelphia 1996.

[10] L.S. Blackford, J. Choi, A. Cleary, E. D'Azevedo, J. Demmel, I. Dhillon, J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, R.C. Whaley: *ScaLAPACK Users' Guide.* SIAM, Philadelphia 1997.

[11] R.A. van de Geijn: *Using PLAPACK: parallel linear algebra package.* The MIT Press, Cambridge, Massachusetts 1997.

[12] G.A. Baker: *Geopotential Recovery Using Satellite Gradiometry and GPS Observables on Parallel Architectures.* Center for Space Research, University of Texas at Austin, JPL Presentation 1997.

[13] W. Gropp, E. Lusk, A. Skellum: *Using MPI: Parallel Portable Programming with the Message Passing Interface.* The MIT Press, Cambridge, Massachusetts 1994.

[14] M. Hanke, P.C. Hansen: *Regularization methods for large–scale problems.* Surveys Math. Indust. **3**, pp. 253–315 (1994).

[15] P.C. Hansen: *Rank–Deficient and Discrete Ill–Posed Problems.* Doctoral Dissertation, UNI•C, Technical University of Denmark, Lyngby, Denmark 1995.

[16] P.C. Hansen: *Regularization Tools, A Matlab Package for Analysis and Solution of Discrete Ill–Posed Problems, Version 3.0 for Matlab 5.2.* Department of Mathematical Modelling, Technical University of Denmark (1998).
http://www.imm.dtu.dk/~pch/Regutools/regutools.html

[17] H.W. Engl, M. Hanke, A. Neubauer: *Regularization of Inverse Problems.* Kluwer Academic Publishers, Dordrecht, Boston, London 1996.

[18] C. Reigber: *Gravity field recovery from satellite tracking data.* in: Theory of satellite goedesy and gravity field determination, eds. F. Sansò, P. Rummel, pp. 197–234, Springer, Berlin 1989.

[19] J. Bouman: *Quality of regularization methods.* DEOS Report no 98.2, Delft University Press, Delft 1998.

[20] D. Calvetti, B. Lewis, L. Reichel: *Smooth or abrupt: a comparison of regularization methods.* to appear in: Proceedings of SPIE 1998 Annual Meeting.

[21] G.H. Golub, P.C. Hansen, D.P.O'Leary: *Tikhonov regularization and total least squares.* IMM Technical Report 15/97, Department of Mathematical Modelling, Technical University of Denmark (1997).

[22] T. Huckle: *Efficient Computation of Sparse Approximate Inverses.* Num. Lin. Alg. **5**, pp. 57–71 (1998).

[23] P.C. Hansen: *Analysis of discrete ill–posed problems by means of the L–curve.* SIAM Review **34**, pp. 561–580 (1992).

[24] P.C. Hansen, D.P. O'Leary: *The use of the L–curve in the regularization of discrete ill–posed problems.* SIAM J. Sci. Comput. **14**, pp. 1487–1503 (1993).

[25] T. Regińska: *A regularization parameter in discrete ill–posed problems.* SIAM J. Sci. Comput. **17**, pp. 740–749 (1996).

[26] R. Floberghagen, J. Bouman: *On the Information Content and Regularization of Lunar Gravity Field Solutions.* Deos Progress Letters **98**, pp. 1–19 (1998).

[27] *The L–Curve and its Use in the Numerical Treatment of Inverse Problems.* in: Advances in Computational Biomedicine, Vol. 3: Inverse Problems in Electrocardiology, ed. P. Johnston, WIT Press, to appear.

[28] M. Hanke: *Limitations of the L–curve method in ill–posed problems.* BIT **36**, pp. 287–301 (1996).

[29] C.R. Vogel: *Non–conergence of the L–Curve Regularization Parameter Selection Method.* Inverse Problems **12**, pp. 535–547 (1996).

[30] G.H. Golub, M.T. Heath, G. Wahba: *Generalized cross–validation as a method for choosing a good ridge parameter.* Technometrics **21**, pp. 215–223 (1979).

[31] G. Wahba: *Practical approximate solutions to linear operator equations when the data are noisy.* SIAM J. Numer. Anal. **14**, 651–667 (1977).

[32] T. Raus: *The principle of the residual in the solution of ill–posed problems with nonselfadjoint operator.* Uchen. Zap. Tartu Gos. Univ. **715**, pp. 12–20 (1985).

[33] H.W. Engl, H. Gfrerer: *A posteriori parameter choice for generalized regularization methods for solving linear ill–posed problems.* Appl. Numer. Math. **4**, pp. 395–417 (1988).

[34] H. Gfrerer: *An a posteriori parameter choice for ordinary and iterated Tikhonov regularization of ill–posed problems leading to optimal convergence rates.* Math. Comp. **49**, pp. 507–522 (1987).

[35] M. Hanke, T. Raus: *A general heuristic for choosing the regularization parameter in ill–posed problems.* SIAM J. Sci. Comput. **17**, pp. 956–972 (1996).

[36] A. Neumaier: *Solving ill–conditioned and singular linear systems: a tutorial on regularization.* SIAM Review **40**, pp. 636–666 (1998).

[37] J. Bouman: *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.

[38] J. Bouman: *Quality Assessment of Geopotential Models by Means of Redundancy Decomposition?* DEOS Progress Letters **97**, pp. 49–54 (1997).

[39] J. Bouman, R. Koop: *Quality Differences between Tikhonov Regularization and Generalized Biased Estimation in Gradiometric Analysis.* DEOS Progress Letters **97**, pp. 42–48 (1997).

[40] J. Bouman, R. Koop: *Stabilization of global gravity field solutions by combining satellite gradiometry and airborne gravimetry.* DEOS Progress Letters **98**, pp. 43–45 (1998).

[41] P. Xu: *The value of minimum norm estimation of geopotential fields.* Geophys. J. Int. **111**, pp. 170–178 (1992).

[42] G. Wahba: *Spline Models for Observational Data.* SIAM, Philadelphia 1990.

[43] A.C. Hindmarsh: *LSODE and LSODI, Two New Initial Value Ordinary Differential Equation Solvers.* ACM SIGNUM Newsletter **15**, pp. 10–11, (1980).

[44] A.C. Hindmarsh: *ODEPACK, a Systematized Collection of ODE Solvers.* In Scientific Computing, R.S. Stepelman et al. (eds.) North–Holland, Amsterdam, pp. 55–64, (1983).

[45] R. Bulirsch, J. Stoer: *Numerical treatment of ordinary differential equations by extrapolation methods.* Numer. Math. **8**, pp. 1–13 (1966).

[46] E. Hairer, S. P. Nørset, G. Wanner: *Solving Ordinary Differential Equations I: Nonstiff Problems.* Second Ed., Springer, Berlin 1993.

[47] J. R. Dormand, P. J. Prince: *A family of embedded Runge–Kutta formulae.* J. Comp. Appl. Math. **6**, pp. 19–26, (1980).

[48] P. Deuflhard: *Recent Progress in Extrapolation Methods for ODE's.* SIAM Review **27**, pp. 55–535 (1985).

[49] P.E. Zadunaisky: *On the estimation of error propagated in the numerical solution of a system of ordinary differential equations.* Numer. Math. **27**, pp. 21–39 (1976).

[50] R.D. Skeel: *Thirteen ways to estimate global error.* Numer. Math. **48**, pp. 1–20 (1986).

[51] P. Deuflhard, F. Bornemann: *Numerische Mathematik II. Integration gewöhnlicher Differentialgleichungen.* Walter de Gruyter, Berlin, New York 1994.