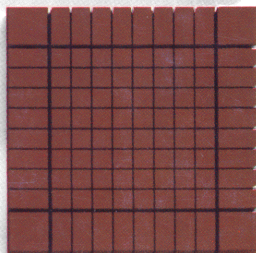
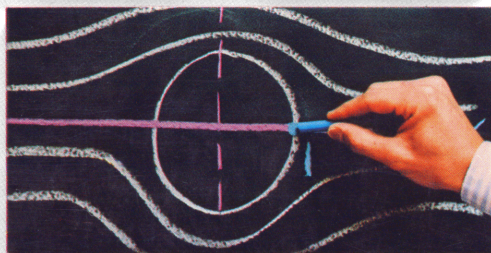
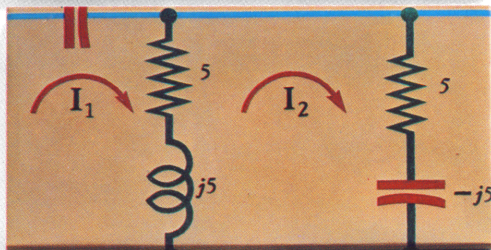
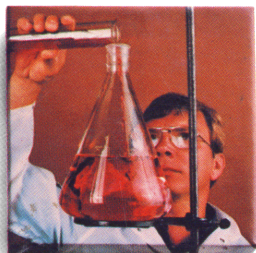
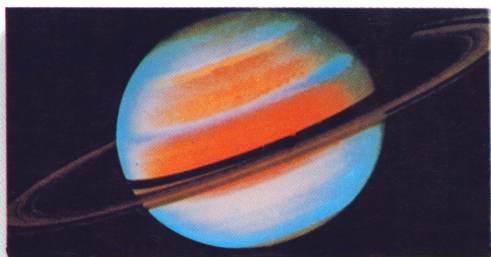


HEWLETT-PACKARD

HP-15C

Fortgeschrittene Funktionen



Hinweis

Hewlett-Packard übernimmt weder ausdrücklich noch stillschweigend irgendwelche Haftung für die in diesem Handbuch dargestellten Programm-Materialien und Tastenfolgen – weder für ihre Funktionsfähigkeit noch für ihre Eignung für irgendeine spezielle Anwendung. Die Tastenfolgen und Programme dienen lediglich der beispielhaften Illustration; der Benutzer trägt das volle Risiko bezüglich ihrer Güte und Durchführbarkeit. Sollten sich Programm-Materialien und/oder Tastenfolgen als fehlerhaft erweisen, trägt der Benutzer (und nicht Hewlett-Packard oder irgendeine dritte Seite) die Kosten für die Korrekturen und alle Neben- und Folgeschäden. Hewlett-Packard haftet für keinerlei Neben- oder Folgeschäden im Zusammenhang mit oder als Folge aus der Lieferung, Benutzung oder Leistung der Programm-Materialien und Tastenfolgen.



HP-15C

Fortgeschrittene Funktionen

Oktober 1982

00015-90013

Inhaltsverzeichnis

Einleitung	5
Abschnitt 1: Wirkungsvolle Verwendung von SOLVE	6
Auffinden von Nullstellen	6
Arbeitsweise von SOLVE	7
Behandlung von pathologischen Fällen	9
Einfache und schwierige Gleichungen	9
Ungenauere Gleichungen	10
Gleichungen mit mehreren Lösungen	10
Verwendung von SOLVE bei Polynome	10
Lösen von Gleichungssystemen	15
Auffinden lokaler Extrema einer Funktion	17
Verwenden der Ableitung	17
Verwendung einer angenäherten Steigung	20
Verwendung einer wiederholten Näherung	23
Anwendungen	26
Annuitäten und Zinseszinsen	26
Analyse diskontierter Zahlungsströme	39
Abschnitt 2: Arbeiten mit $\int$	45
Numerische Integration mit Hilfe von $\int$	45
Genauigkeit der zu integrierenden Funktion	47
Funktionen, die physikalische Vorgänge beschreiben	47
Rundungsfehler in internen Berechnungen	49
Verkürzung der Rechenzeit	49
Unterteilung des Integrationsintervalls	50
Transformation von Variablen	54
Berechnung von schwierigen Integralen	55
Anwendungen	60
Abschnitt 3: Rechnen im Komplex-Modus	65
Gebrauch des Komplex-Modus	65
Trigonometrische Modi	68
Definitionen mathematischer Funktionen	68
Arithmetische Operationen	69
Einwertige Funktionen	69

Mehrwertige Funktionen	69
Verwendung von SOLVE und  im Komplex-Modus	73
Genauigkeit im Komplex-Modus	73
Anwendungen	76
Speichern und Zurückrufen von komplexen Zahlen mittels einer Matrix	76
Berechnung der n -ten Wurzeln einer komplexen Zahl	78
Berechnen der komplexen Nullstellen einer Gleichung	80
Konturintegrale	85
Komplexe Potentiale	89
Abschnitt 4: Matrixoperationen	96
Zum Verständnis der LR -Zerlegung	96
Schlechtkonditionierte Matrizen und die Konditionszahl	98
Genauigkeit von numerischen Lösungen linearer Gleichungssysteme ..	103
Vereinfachung von schwierigen Gleichungssystemen	104
Skalierung	104
Vorkonditionierung	107
Berechnungen mit kleinsten Quadraten	110
Normalgleichungen	110
Orthogonal-Faktorisierung	113
Singuläre und fastsinguläre Matrizen	117
Anwendungen	119
Konstruktion einer Einheitsmatrix	119
Einmalige Nachiteration	119
Lösen eines nichtlinearen Gleichungssystems	122
Lösen eines großen komplexen Gleichungssystems	128
Methode der kleinsten Quadrate mit Normalgleichungen	131
Methode der kleinsten Quadrate mit sukzessiven Zeilen	140
Eigenwerte einer symmetrischen reellen Matrix	148
Eigenvektoren einer symmetrischen reellen Matrix	154
Optimierung	160
Anhang: Genauigkeit numerischer Rechnungen	172
Mißverständnisse im Zusammenhang mit Fehlern	172
Eine Fehlerhierarchie	178
Kategorie 0: Gar kein Fehler	178
Kategorie ∞ : Overflow/Underflow	179
Kategorie 1: Korrekt oder fast korrekt gerundet	179
Kategorie 1C: Kategorie 1 im Komplexen	183
Kategorie 2: Korrekte Rundung bei möglicherweise gestörten (ungenauen) Eingaben	184

4 Inhalt

Trigonometrische Funktionen von reellen Winkeln in Radiant	184
Rückwärts-Fehleranalyse	187
Rückwärts-Fehleranalyse und Singularitäten	192
Erste Zusammenfassung der Rückwärtsanalyse	194
Rückwärts-Fehleranalyse bei der Matrixinversion	200
Ist die Rückwärts-Fehleranalyse sinnvoll?	204
Index	212

Einleitung

Der HP-15C bietet Ihnen mehrere fortgeschrittene Funktionen, die bisher noch nie in einem Taschenrechner dieses Formats so bequem kombiniert wurden:

- Bestimmen der Nullstellen einer Gleichung,
- Berechnung von bestimmten Integralen,
- Berechnungen mit komplexen Zahlen und
- Matrizenrechnungen.

Das *HP-15C Benutzerhandbuch* gibt Ihnen alle grundlegenden Informationen über die Verwendung dieser fortgeschrittenen Operationen. Es enthält auch eine Vielzahl von Beispielen, die ihre Verwendung verdeutlicht. Das Benutzerhandbuch ist damit für Sie die primäre Referenz für Informationen über die fortgeschrittenen Funktionen.

Dieses Handbuch *HP-15C: Fortgeschrittene Funktionen* beginnt, wo das Benutzerhandbuch aufhört. Sie werden hier Informationen darüber finden, wie der HP-15C die fortgeschrittenen Berechnungen ausführt und wie die Ergebnisse dieser Berechnungen interpretiert werden können. Innerhalb dieses Handbuchs wird die amerikanische Form der Zahlendarstellung benutzt; d.h. der Punkt als Dezimaltrennzeichen und das Komma als Zifferntrennzeichen.

Ebenso enthält dieses Handbuch viele Programme und Anwendungsbeispiele. Die Programme verfolgen in erster Linie die Absicht, Ihnen Beispiele für die Verwendung der fortgeschrittenen Funktionen aufzuzeigen. Damit sollten Sie in der Lage sein, diese Fähigkeiten in Ihren eigenen Programmen effektiver auszunützen zu können. Zum anderen behandeln diese Programmbeispiele ein weites Gebiet von Anwendungen, so daß sie in der in diesem Handbuch angebotenen Form nützlich für Sie sein könnten.

Hinweis: Die Diskussion der meisten Themen in diesem Handbuch setzt voraus, daß Sie die grundlegenden Informationen über die Verwendung der fortgeschrittenen Funktionen verstanden haben und Sie im allgemeinen mit der behandelten Materie vertraut sind.

Wirkungsvolle Verwendung von **SOLVE**

Der **SOLVE** Algorithmus liefert eine effektive Methode zum Auffinden der Lösungen einer Gleichung. Dieser Abschnitt beschreibt die von **SOLVE** verwendete numerische Methode und gibt Ihnen nützliche Informationen über die Anwendung von **SOLVE** in verschiedenen Situationen.

Auffinden von Nullstellen

Keine numerische Methode kann ganz allgemein garantieren, eine Nullstelle einer Funktion, die mindestens eine Nullstelle besitzt, zu finden. Da nur eine endliche Anzahl von Stellen verwendet wird, ist es möglich, daß sich die berechnete von der theoretischen Funktion in bestimmten Intervallen von x unterscheidet. Außerdem kann die Nullstelle möglicherweise nicht exakt dargestellt oder zwischen Nullstellen und Unstetigkeiten unterschieden werden. Da zur Berechnung nur eine endliche Zahl von Werten der Funktion verwendet werden kann, ist es auch möglich, daß fälschlicherweise der Schluß gezogen wird, die Funktion habe keine Nullstelle.

Trotz dieser allen numerischen Methoden eigenen Einschränkungen sollte ein effektiver Algorithmus – wie der von **SOLVE** verwendete – die folgenden Eigenschaften haben:

- Existiert eine reelle Nullstelle und ist der Rechner in der Lage, diese exakt darzustellen, dann sollte diese gefunden werden. Beachten Sie, daß bei der Berechnung der Funktion für ein bestimmtes x ein Underflow auftreten kann, der den Funktionswert durch Null ersetzt, obwohl es sich bei dem betrachteten x -Wert nicht um eine Nullstelle handelt.
- Existiert eine reelle Nullstelle, die aber vom Rechner nicht exakt dargestellt werden kann, sollte sich das Ergebnis des numerischen Algorithmus vom exakten Wert der Nullstelle nur in der letzten signifikanten Stelle unterscheiden.
- Existiert keine reelle Nullstelle, sollte eine Fehlermeldung in der Anzeige erscheinen.

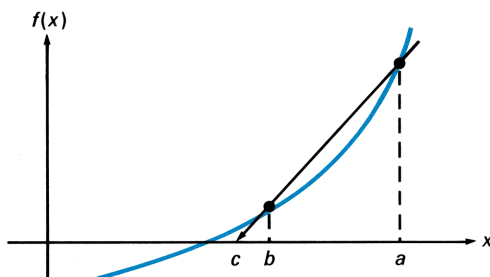
Der **SOLVE**-Algorithmus ist darauf ausgelegt, diese Kriterien zu erfüllen. Er ist leicht zu verwenden und benötigt wenig Speicherplatz. Da **SOLVE** in einem Programm auch den Fall, daß keine Nullstelle gefunden wird, meldet, kann Ihr Programm gänzlich automatisch bleiben, gleichgültig, ob **SOLVE** eine Nullstelle findet oder nicht.

Arbeitsweise von **SOLVE**

Die **SOLVE** Routine benutzt nur fünf Register des aufteilbaren Speichers des HP-15C. Die fünf Register enthalten die Stützstellen a , b und c und die beiden letzten Funktionswerte $f(a)$ und $f(b)$. Das von Ihnen zur Verfügung zu stellende Unterprogramm berechnet den Wert für $f(c)$.

Der Schlüssel für die Effizienz von **SOLVE** ist die Methode mit der der nächste Wert für c gefunden wird.

Normalerweise verwendet **SOLVE** das Sekantenverfahren zum Auffinden des nächsten Wertes. Diese Methode benutzt die Werte a , b , $f(a)$ und $f(b)$, um einen Wert c zu berechnen, dessen Funktionswert $f(c)$ näher an Null liegt.

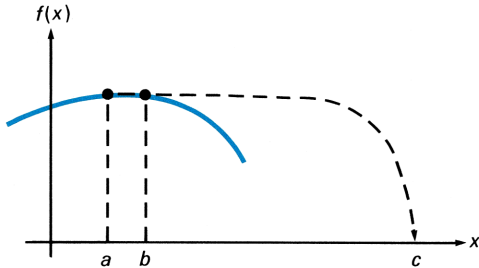


Wenn c keine Nullstelle ist, aber $f(c)$ näher bei Null liegt als $f(b)$, dann wird b in a umbenannt, c als b bezeichnet und der «Voraussageprozeß» wiederholt. Angenommen der Graph von $f(x)$ ist glatt und die anfänglichen Werte von a und b sind in der Nähe einer einfachen Nullstelle, dann konvergiert das Sekantenverfahren sehr schnell zu einer Nullstelle.

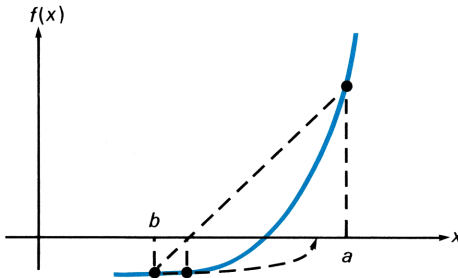
Unter bestimmten Bedingungen jedoch liefert das Sekantenverfahren keinen nächsten Wert, der die Suche nach einer Nullstelle beendet oder sie einer Nullstelle näherbringt (also z.B. einen Vorzeichenwechsel oder einen kleineren Funktionswert findet). In solchen Fällen verwendet **SOLVE** einen anderen Ansatz.

Wenn die berechnete Sekante annähernd waagrecht ist, modifiziert **SOLVE** das Sekantenverfahren, um sicherzustellen, daß $|c - b| \leq 100|a - b|$ ist. Dies ist besonders wichtig, da dadurch auch die Gefahr verringert wird, daß das Sekantenverfahren irreführende Ergebnisse erzeugt, wenn der Rundungsfehler in der Nähe der Nullstelle signifikant wird.

8 Abschnitt 1: Wirkungsvolle Verwendung von **SOLVE**



Wenn **SOLVE** schon Werte für a und b gefunden hat, so daß $f(a)$ und $f(b)$ verschiedene Vorzeichen haben, wird das Sekantenverfahren derartig modifiziert, daß c immer innerhalb des Intervalls liegt, das den Vorzeichenwechsel enthält. Dies garantiert, daß sich das Suchintervall mit jeder Iteration verkleinert und schließlich eine Nullstelle gefunden wird.



Wenn eine Iteration zu keinem Vorzeichenwechsel führt und auch die neu gefundene Stützstelle keinen betragsmäßig kleineren Funktionswert hat, legt **SOLVE** eine Parabel durch die Funktionswerte von a , b und c . **SOLVE** berechnet dann den Wert d , an dem die Parabel ihr Maximum oder Minimum hat, benennt d in a um und setzt dann die Suche nach einer Nullstelle mit dem Sekantenverfahren fort.

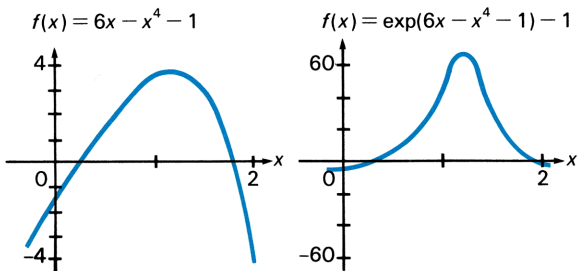
SOLVE gibt die Suche erst dann auf, wenn drei aufeinanderfolgende Parabelanpassungen zu keiner Verkleinerung des Funktionsbetrags geführt haben oder wenn $d = b$ ist. Unter diesen Bedingungen erscheint die Meldung **Error 8** in der Anzeige. Da b die Stützstelle mit dem kleinsten gefundenen Funktionsbetrag ist, werden b und $f(b)$ im X- und Z-Register abgelegt. Das Y-Register enthält den Wert von a bzw. c . Mit diesen Informationen können Sie entscheiden, wie die Berechnung fortgesetzt werden soll. Sie können die Suche dort fortsetzen, wo sie unterbrochen wurde, oder sie an einer anderen Stelle beginnen; oder Sie entscheiden, daß $f(b)$ vernachlässigbar klein ist, so daß $x = b$ die Nullstelle darstellt; oder sie transformieren die Gleichung in die Form einer leichter zu lösenden Gleichung; oder Sie kommen zu dem Schluß, daß keine Nullstelle existiert.

Behandlung pathologischer Fälle

Die folgenden Informationen sind sehr nützlich für die Bearbeitung von Problemen, die zu mißverständlichen Ergebnissen führen könnten. Ungenaue Berechnungen von Nullstellen resultieren aus Unterschieden zwischen den berechneten Funktionswerten und den gewünschten Funktionswerten. Häufig können Sie Schwierigkeiten vermeiden, wenn Sie wissen, wie Sie die Ungenauigkeiten analysieren und reduzieren können.

Einfache und schwierige Gleichungen

Die zwei Gleichungen $f(x) = 0$ und $e^{f(x)} - 1 = 0$ haben die gleichen reellen Nullstellen, wobei eine fast immer leichter zu lösen ist als die andere. Wenn z.B. $f(x) = 6x - x^4 - 1$ ist, läßt sich die erste Gleichung leichter lösen. Für $f(x) = \ln(6x - x^4)$ ist die zweite leichter. Die Schwierigkeiten liegen in dem Verhalten des Graphen der Funktion besonders in der Umgebung der Nullstelle.



Im allgemeinen gilt, daß jede Gleichung eine aus einer unendlich großen Familie äquivalenter Gleichungen ist, die alle die gleichen reellen Nullstellen haben. Einige dieser Gleichungen sind leichter zu lösen als andere. Wenn **SOLVE** für eine dieser Gleichungen keine Nullstelle finden kann, ist der Algorithmus für eine andere Gleichung derselben Familie möglicherweise erfolgreich.

Ungenauere Gleichungen

SOLVE berechnet die Lösungen einer Gleichung niemals falsch, *wenn die Funktion selbst nicht falsch berechnet wird*. Die Genauigkeit Ihres Funktionsunterprogramms bestimmt die Genauigkeit der berechneten Nullstellen.

Unter bestimmten Bedingungen unterscheiden sich der von Ihnen berechnete und der theoretische Funktionswert, den Sie berechnen wollten. **SOLVE** kann nicht auf nachfolgende Werte Ihrer Funktion schließen. Häufig können Sie jedoch den Fehler bei der Berechnung minimieren, indem Sie Ihr Funktionsunterprogramm sorgfältig entwerfen.

Gleichungen mit mehreren Lösungen

Die Aufgabe, alle Nullstellen einer Funktion zu berechnen, wird mit zunehmender Anzahl von Wurzeln einer Funktion schwieriger. Außerdem wird im Fall, daß die Nullstellen sehr eng beieinander liegen, eine exakte Auflösung nicht möglich sein. Sie können die Methode der *Deflation* verwenden, um Nullstellen zu eliminieren. Dieses Verfahren wird im *HP-15C Benutzerhandbuch* beschrieben.

Eine Gleichung mit einer mehrfachen Nullstelle ist dadurch gekennzeichnet, daß die Funktion selbst und die ersten Ableitungen bei der mehrfachen Nullstelle den Wert 0 annehmen. Wenn **SOLVE** eine doppelte Nullstelle findet, ist die letzte Hälfte der berechneten Stellen in der Regel ungenau. Bei einer dreifachen Nullstelle dürften zweidrittel, bei einer vierfachen Nullstelle etwa dreiviertel der Stellen unsicher sein.

Verwendung von **SOLVE** bei Polynomen

Polynome gehören zu den am einfachsten zu berechnenden Funktionen. Aus diesem Grund werden sie üblicherweise dazu verwendet, Funktionen anzunähern, die physikalische Prozesse oder kompliziertere mathematische Funktionen repräsentieren.

Ein Polynom n -ten Grades kann folgendermaßen dargestellt werden:

$$a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0.$$

Diese Funktion ist gleich Null für nicht mehr als n reelle Werte von x , die als Nullstellen oder Wurzeln des Polynoms bezeichnet werden. Ein Grenzwert für

die Anzahl von *positiven* Nullstellen dieser Funktion kann bestimmt werden, indem Sie abzählen, wie oft sich das Vorzeichen der Koeffizienten ändert, wenn Sie das Polynom von links nach rechts untersuchen. Auf ähnliche Weise kann man einen oberen Grenzwert für die Anzahl von *negativen* Nullstellen erhalten, indem man in dem Polynom $-x$ für x substituiert und dann wie oben vorgeht. Ist die tatsächliche Zahl von positiven oder negativen Nullstellen kleiner als der entsprechende Grenzwert, dann unterscheiden sich die Grenzwerte um eine gerade Zahl. (Diese Relation bezeichnet man als Vorzeichenregel von Descartes.)

Lassen Sie uns als ein Beispiel das folgende Polynom betrachten:

$$f(x) = x^3 - 3x^2 - 6x + 8.$$

Es kann nicht mehr als drei Nullstellen haben. Es kann maximal zwei positive Nullstellen haben (beachten Sie den Vorzeichenwechsel vom ersten zum zweiten und vom dritten zum vierten Term) und maximal eine negative Nullstelle (dies folgt aus $f(-x) = -x^3 - 3x^2 + 6x + 8$).

Polynome werden in der Regel auf sehr kompakte Weise ausgewertet, indem man ineinandergeschachtelte Multiplikationen verwendet. (Diese Methode wird manchmal als Horner-Schema bezeichnet.) Die obige Funktion kann in folgende Form umgeschrieben werden:

$$f(x) = [(x-3)x-6]x + 8.$$

Diese Darstellung ist leichter zu programmieren und wirkungsvoller auszuführen als die Originalform, besonders da **SOLVE** den Stack mit dem Wert von x auffüllt.

Beispiel: Im Winter 1978 wartete der Arktikforscher Jean-Claude Coulerre in seinem eingeschnittenen Camp hoch im Norden auf die Rückkehr der Sonne. Coulerre wußte, daß die Sonne erst in den ersten Tagen des Monats März sichtbar sein würde. Zu diesem Zeitpunkt erreicht sie eine Deklination von $5^\circ 18'S$. An welchem Tag im März wird sich die Sonne dem frierenden Forscher wieder zeigen?

Der Zeitpunkt im März, an dem die Sonne eine Deklination von $5^\circ 18'S$ erreicht, kann ermittelt werden, indem Sie t aus der folgenden Gleichung berechnen:

$$D = a_4 t^4 + a_3 t^3 + a_2 t^2 + a_1 t + a_0$$

wobei D die Deklination in Grad und t die Zeit in Tagen, vom Beginn des Monats an gerechnet, bedeuten.

$$\begin{aligned}a_4 &= 4.2725 \times 10^{-8} \\a_3 &= -1.9931 \times 10^{-5} \\a_2 &= 1.0229 \times 10^{-3} \\a_1 &= 3.7680 \times 10^{-1} \\a_0 &= -8.1806.\end{aligned}$$

Diese Gleichung gilt für $1 \leq t < 32$, was den Monat März 1978 darstellt.

Konvertieren Sie zuerst die Gradangabe $5^\circ 18' S$ in Dezimalform (drücken Sie dazu **5.18** **[CHS]** **[g]** **[↔H]**); dies führt zu -5.5300 (unter Verwendung des Anzeigemodus **[FIX]** 4). (Südliche Breiten werden als negative Zahlen dargestellt.)

Die Lösung von Coulerre's Problem ist der Wert von t , der die folgende Gleichung erfüllt:

$$-5.300 = a_4 t^4 + a_3 t^3 + a_2 t^2 + a_1 t + a_0.$$

In die Form umgeschrieben, wie sie für **[SOLVE]** benötigt wird, hat die Gleichung folgende Gestalt:

$$0 = a_4 t^4 + a_3 t^3 + a_2 t^2 + a_1 t - 2.8806$$

wobei der letzte konstante Term den Wert der Deklination beinhaltet.

Verwendet man das Horner-Schema, nimmt die Funktion, die Null gesetzt werden soll, folgende Form an:

$$f(t) = (((a_4 t + a_3)t + a_2)t + a_1)t - 2.8806.$$

Um das Unterprogramm abzukürzen, speichern und rufen Sie die Konstanten zurück, indem Sie die Register verwenden, die dem Exponenten von t entsprechen.

Tastenfolge

[ON]/[-]

[↔]

[g] [P/R]

Anzeige

Pr Error

0.0000

000-

Löscht den Speicher des Rechners*.

Programm-Modus.

* Dieser Schritt wurde hier nur deshalb eingefügt, um sicherzustellen, daß genügend Speicherplatz für die in diesem Handbuch folgenden Beispiele zur Verfügung steht.

Tastenfolge**Anzeige**

f LBL A	001–42,21,11
RCL 4	002– 45 4
x	003– 20
RCL 3	004– 45 3
+	005– 40
x	006– 20
RCL 2	007– 45 2
+	008– 40
x	009– 20
RCL 1	010– 45 1
+	011– 40
x	012– 20
RCL 0	013– 45 0
+	014– 40
g RTN	015– 43 32

Geben Sie im Run-Modus die fünf Koeffizienten ein:

Tastenfolge**Anzeige**

g P/R		Run-Modus.
4.2725 EEX 8 CHS	4.2725 –08	
STO 4	4.2725 –08	Koeffizient von t^4 .
1.9931 CHS EEX		
5 CHS STO 3	–1.9931 –05	Koeffizient von t^3 .
1.0229 EEX 3 CHS	1.0229 –03	
STO 2	0.0010	Koeffizient von t^2 .
3.7680 EEX 1 CHS	3.7680 –01	
STO 1	0.3768	Koeffizient von t .
2.8806 CHS STO 0	–2.8806	Konstanter Term.

Da die gewünschte Lösung zwischen 1 und 32 liegen soll, geben Sie diese beiden Werte als Anfangsnäherungen ein. Benutzen Sie dann **SOLVE**, um die Nullstelle zu finden.

Tastenfolge**Anzeige**

1 ENTER	1.0000	
32	32	Anfangsnäherung.
f SOLVE A	7.5137	Berechnete Nullstelle.
R*	7.5137	Identische vorletzte Näherung.

14 Abschnitt 1: Wirkungsvolle Verwendung von **SOLVE**

Tastenfolge	Anzeige	
R↵	0.0000	Funktionswert.
g R↵ g R↵	7.5137	Restauriert den Stack.

Der Tag ist der siebte März. Konvertieren Sie den gebrochenen Anteil der Zahl in dezimale Stunden und dann in Stunden, Minuten und Sekunden.

Tastenfolge	Anzeige	
f FRAC	0.5137	Gebrochener Anteil der Tagesangabe.
24 x	12.3293	Dezimalangabe der Stunden.
f →H.MS	12.1945	Stunden, Minuten, Sekunden.

Der Forscher Coulerre wird die Sonne am siebten März um 12^h 19^m 45^s (Coordinated Universal Time) wiedersehen.

Wenn Sie Coulerre's Funktion $f(t)$ genau betrachten, erkennen Sie, daß sie vier reelle Nullstellen haben kann – drei positive und eine negative. Versuchen Sie weitere positive Nullstellen zu finden, indem Sie **SOLVE** mit größeren positiven Anfangsnäherungen verwenden.

Tastenfolge	Anzeige	
1000 ENTER 1100	1,100	Zwei größere positive Anfangsnäherungen.
f SOLVE A	Error 8	Keine Nullstelle wurde gefunden.
↵	278.4497	Letzte SOLVE Näherung.
R↵	276.7942	Vorletzte SOLVE Näherung.
R↵	7.8948	Funktionswert ungleich Null.
g R↵ g R↵	278.4497	Restauriert den Stack.
f SOLVE A	Error 8	Wieder keine Nullstelle gefunden.
↵	278.4398	Letzte SOLVE Näherung fast unverändert.
R↵	278.4497	Vorletzte SOLVE Näherung.
R↵	7.8948	Der gleiche Funktionswert.

Sie haben ein positives lokales Minimum gefunden, aber keine Nullstelle. Versuchen Sie jetzt, die negative Wurzel zu finden.

Tastenfolge1000 **CHS** **ENTER**1100 **CHS****f** **SOLVE** **A****R↵****R↵****Anzeige****-1,000.0000****-1,100****-108.9441****-108.9441****1.6000 -08**

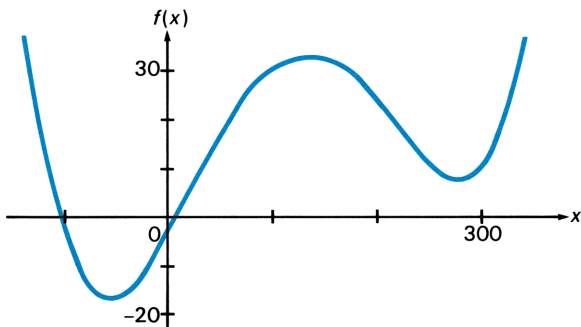
Zwei größere negative Anfangsnäherungen.

Negative Wurzel.

Vorletzte Näherung.

Funktionswert.

Es gibt keinen Grund weiter zu suchen, Sie haben alle möglichen Lösungen gefunden. Die negative Lösung hat keine Bedeutung, da sie außerhalb des Intervalls liegt, in dem die Deklinationsnäherung gültig ist. Der Graph der Funktion bestätigt das Ergebnis, das Sie gefunden haben.



Lösen von Gleichungssystemen

SOLVE wurde so konzipiert, daß es den Wert einer einzigen Variablen sucht, der eine einzelne Gleichung erfüllt. Wenn eine Problemstellung ein System von Gleichungen mit mehreren Variablen enthält, können Sie unter Umständen trotzdem **SOLVE** verwenden, um die Lösung zu finden.

Bei einigen Gleichungssystemen der Form

$$\begin{aligned} f_1(x_1, \dots, x_n) &= 0 \\ &\vdots \\ f_n(x_1, \dots, x_n) &= 0 \end{aligned}$$

ist es durch algebraische Manipulationen möglich, alle bis auf eine Variable zu eliminieren. Dabei werden die Gleichungen so modifiziert, daß sich alle Variablen bis auf eine in Abhängigkeit von der verbleibenden Variablen darstellen lassen. Mit diesen Ausdrücken können Sie das Problem darauf reduzieren, mit **SOLVE** die Nullstelle einer einzelnen Gleichung zu finden. Die Werte der anderen Variablen der Nullstelle können dann mit den abgeleiteten Ausdrücken berechnet werden.

16 Abschnitt 1: Wirkungsvolle Verwendung von SOLVE

Dieses Verfahren ist oftmals nützlich, um die komplexe Nullstelle einer komplexen Gleichung zu finden. In solchen Problemen kann die komplexe Gleichung durch zwei reelle Gleichungen ausgedrückt werden; eine für die reelle Komponente und eine für die imaginäre Komponente – mit zwei reellen Variablen, die den Real- und Imaginärteil der komplexen Nullstelle darstellen.

Die komplexe Gleichung $z + 9 + 8e^{-z} = 0$ hat beispielsweise keine reellen Lösungen z , aber hat unendlich viele komplexe Lösungen $z = x + iy$. Diese Gleichung kann durch zwei reelle Gleichungen ausgedrückt werden:

$$\begin{aligned}x + 9 + 8e^{-x} \cos y &= 0 \\y - 8e^{-x} \sin y &= 0\end{aligned}$$

y kann nun wie folgt aus den Gleichungen eliminiert werden. Da das Vorzeichen von y in den Gleichungen keine Rolle spielt, nehmen wir an, daß $y > 0$ ist, so daß jede Lösung (x, y) automatisch die Lösung $(x, -y)$ liefert. Schreiben Sie die zweite Gleichung folgendermaßen um:

$$x = \ln(8(\sin y)/y),$$

wobei $\sin y > 0$ sein muß, so daß $2n\pi < y < (2n+1)\pi$ für $n = 0, 1, \dots$

Die erste Gleichung wird zu

$$\begin{aligned}y &= \cos^{-1}(-e^x(x+9)/8) + 2n\pi \\&= (2n+1)\pi - \cos^{-1}(e^x(x+9)/8)\end{aligned}$$

für $n = 0, 1, \dots$ Substituiert man diese Gleichung in die zweite Gleichung, so erhält man den folgenden Ausdruck:

$$x + \ln\left(\frac{(2n+1)\pi - \cos^{-1}(e^x(x+9)/8)}{\sqrt{64 - (e^x(x+9))^2}}\right) = 0.$$

Sie können jetzt **SOLVE** verwenden, um die Lösung x dieser Gleichung zu finden (für jeden gegebenen Wert von n , der Anzahl von Nullstellen). Wenn Sie x kennen, können Sie den entsprechenden Wert von y berechnen.

Als letztes wollen wir an diesem Beispiel die Auswahl der Anfangsnäherungen betrachten. Da das Argument des Arcuscosinus Werte zwischen -1 und 1 annimmt, muß x kleiner als -0.1059 sein (diese Zahl findet man durch Probieren oder mit Hilfe von **SOLVE**). Die Anfangsnäherungen sollten in der Nähe dieses Wertes liegen, also z.B. -0.11 und -0.2 .

(Die komplexe Gleichung in diesem Beispiel wird auf Seite 81 mit einem iterativen Verfahren gelöst. Eine andere Methode zur Lösung eines Systems nichtlinearer Gleichungen wird auf Seite 122 beschrieben.)

Auffinden lokaler Extrema einer Funktion

Verwenden der Ableitung

Das übliche Verfahren zur Suche relativer Maxima und Minima des Graphs einer Funktion verwendet die *Ableitung* der Funktion. Die Ableitung ist eine Funktion, die die Steigung des Graphs beschreibt. x -Werte, bei denen die Ableitung Null ist, repräsentieren mögliche lokale Extrema der Funktion. (Obwohl bei gutartigen Funktionen ungewöhnlich, können auch diejenigen Werte von x , an denen die Ableitung unendlich oder undefiniert ist, mögliche Extrema darstellen.) Wenn es möglich ist, die Ableitung einer Funktion in geschlossener Form darzustellen, können Sie **SOLVE** verwenden, um die Nullstellen der Ableitung zu finden, was wiederum die Stellen relativer Minima und Maxima anzeigt.

Beispiel: Für den Entwurf eines senkrechten Sendeturms möchte der Radioingenieur Karlchen Strahl den Winkel finden, unter dem die relative Feldintensität vom Turm aus gesehen am kleinsten ist. Die von dem Turm erzeugte relative Intensität ist gegeben durch die Formel.

$$E = \frac{\cos(2\pi h \cos \theta) - \cos(2\pi h)}{[1 - \cos(2\pi h)] \sin \theta}$$

wobei E die relative Feldintensität bedeutet, h die Antennenhöhe (ausgedrückt in Wellenlängen) darstellt und θ der Winkel in Radiant, von der Vertikalen aus gemessen, ist. Im Entwurf ist die Höhe des Turms 0.6 Wellenlängen.

Der gesuchte Winkel ist der, für den die Ableitung der relativen Intensität nach θ gleich Null ist.

18 Abschnitt 1: Wirkungsvolle Verwendung von **SOLVE**

Um Programmspeicherplatz und Ausführungszeit zu sparen, speichern Sie die folgenden Konstanten in Registern ab und rufen sie bei Bedarf wieder ab:

$$\begin{array}{ll} r_0 = 2\pi h & \text{wird in Register } R_0 \text{ gespeichert,} \\ r_1 = \cos(2\pi h) & \text{wird in Register } R_1 \text{ gespeichert,} \\ r_2 = 1/[1 - \cos(2\pi h)] & \text{wird in Register } R_2 \text{ gespeichert.} \end{array}$$

Die Ableitung der Intensität E nach dem Winkel θ wird durch die folgende Gleichung beschrieben:

$$\frac{dE}{d\theta} = r_2 \left[r_0 \sin(r_0 \cos \theta) - \frac{\cos(r_0 \cos \theta) - r_1}{\sin \theta \tan \theta} \right].$$

Geben Sie ein Unterprogramm zur Berechnung der Ableitung ein:

Tastenfolge	Anzeige	
g P/R	00 –	Programm-Modus.
f CLEAR PRGM	00 –	
f LBL 0	001 – 42,21, 0	
COS	002 – 24	
RCL 0	003 – 45 0	
x	004 – 20	
COS	005 – 24	
RCL 1	006 – 45 1	
–	007 – 30	
x$\geq$y	008 – 34	
SIN	009 – 23	
÷	010 – 10	
x$\geq$y	011 – 34	
TAN	012 – 25	
÷	013 – 10	
CHS	014 – 16	
x$\geq$y	015 – 34	
COS	016 – 24	
RCL 0	017 – 45 0	

Tastenfolge	Anzeige
$\times$	018— 20
SIN	019— 23
RCL 0	020— 45 0
$\times$	021— 20
$+$	022— 40
RCL 2	023— 45 2
$\times$	024— 20
g RTN	025— 43 32

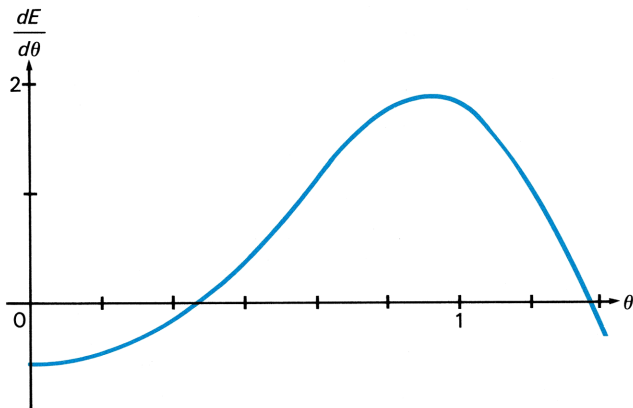
Berechnen und speichern Sie im Radiant-Modus die drei Konstanten.

Tastenfolge	Anzeige	
g P/R		Run-Modus.
g RAD		Schaltet in den Radiant-Modus.
2 g π $\times$	6.2832	
.6 $\times$ STO 0	3.7699	Konstante r_0 .
COS STO 1	-0.8090	Konstante r_1 .
CHS 1 $+$	1.8090	
1/x STO 2	0.5528	Konstante r_2 .

Die relative Feldintensität ist maximal bei einem Winkel von 90° (senkrecht zum Turm). Um das Minimum zu finden, verwenden Sie Anfangsnäherungen näher bei Null, beispielsweise die Winkel 10° und 60° im Bogenmaß.

Tastenfolge	Anzeige	
10 f $\rightarrow$ RAD	0.1745	
60 f $\rightarrow$ RAD	1.0472	Anfangsnäherungen.
f SOLVE 0	0.4899	Lösungswinkel, der eine Steigung von Null ergibt.
R$\downarrow$ R$\uparrow$	-5.5279 -10	Steigung an dem gefundenen Winkel.
g R$\uparrow$ g R$\uparrow$	0.4899	Restauriert den Stack.
g $\rightarrow$ DEG	28.0680	Winkel in Altgrad.

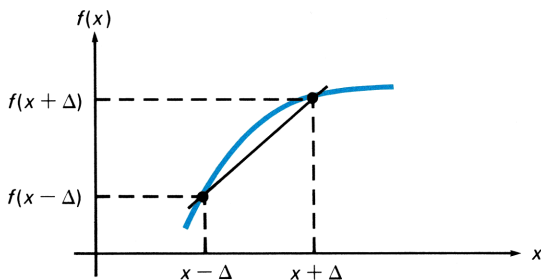
Die relative Feldintensität ist am kleinsten unter einem Winkel von 28.0680° gegenüber der Horizontalen.



Verwendung einer angenäherten Steigung

Die Ableitung einer Funktion kann auch numerisch angenähert werden. Wenn Sie eine Funktion in zwei Punkten relativ nahe bei x auswerten (etwa bei $x + \Delta$ und $x - \Delta$), können Sie die Steigung der Sekante als Approximation für die Steigung der Funktion in x verwenden:

$$s = \frac{f(x + \Delta) - f(x - \Delta)}{2\Delta}$$



Die Genauigkeit dieser Annäherung hängt von der Größe von Δ und der Natur der Funktion ab. Kleinere Werte von Δ ergeben bessere Approximationen der Ableitung, aber übertrieben kleine Werte können die Ursache für einen Rundungsfehler sein. Bei Werten von x , an denen die Steigung Null ist, befindet sich möglicherweise ein relatives Minimum oder Maximum der Funktion.

Beispiel: Lösen Sie das vorherige Beispiel nochmals, jedoch diesmal ohne die Formel für die Ableitung, $dE/d\theta$, zu verwenden.

Finden Sie den Winkel, an dem die Abteilung (numerisch berechnet) der Intensität E gleich Null ist.

Geben Sie im Programm-Modus die folgenden zwei Unterprogramme ein: eines, um die Ableitung der Intensität abzuschätzen, und eines, um die Intensitätsfunktion E zu berechnen. In dem folgenden Unterprogramm wird die Steigung zwischen $\theta - 0.001$ und $\theta + 0.001$ Radiant berechnet. (Ein Intervall, das ungefähr 0.1° entspricht.)

Tastenfolge

Anzeige

g **P/R**

000–

Programm-Modus.

f **LBL** **A**

001–42,21,11

EEX

002– 26

CHS

003– 16

3

004– 3

Berechnet E für $\theta + 0.001$.

+

005– 40

ENTER

006– 36

GSB **B**

007– 32 12

x $\geq$ y

008– 34

EEX

009– 26

CHS

010– 16

3

011– 3

Berechnet E für $\theta - 0.001$.

–

012– 30

ENTER

013– 36

GSB **B**

014– 32 12

–

015– 30

2

016– 2

EEX

017– 26

CHS

018– 16

3

019– 3

Tastenfolge**Anzeige**

$\div$	020–	10
g RTN	021–	43 32
f LBL B	022–	42,21,12
COS	023–	24
RCL 0	024–	45 0
$\times$	025–	20
COS	026–	24
RCL 1	027–	45 1
$-$	028–	30
$x \geq y$	029–	34
SIN	030–	23
$\div$	031–	10
RCL 2	032–	45 2
$\times$	033–	20
g RTN	034–	43 32

Unterprogramm für $E(\theta)$.

Im vorherigen Beispiel war der Rechner im Radiant-Modus, und die drei Konstanten wurden in den Registern R_0 , R_1 und R_2 gespeichert. Geben Sie die gleichen Anfangsnäherungen ein wie zuvor und führen Sie **SOLVE** aus.

Tastenfolge**Anzeige**

g P/R	
10 f $\rightarrow$ RAD	0.1745
60 f $\rightarrow$ RAD	1.0472
f SOLVE A	0.4899
R$\uparrow$ R$\uparrow$	0.0000
g R$\uparrow$ g R$\uparrow$	0.4899
ENTER ENTER f B	-0.2043
$x \geq y$	0.4899
g $\rightarrow$ DEG	28.0679

Run-Modus.

Anfangsnäherungen.

Lösungswinkel, der eine Steigung von Null ergibt.
Steigung des gefundenen Winkels.

Restauriert den Stack.

Verwendet das Funktions-
unterprogramm, um die mini-
male Intensität zu berechnen.

Ruft den θ -Wert zurück.

Winkel in Altgrad.

Diese numerische Näherung der Ableitung zeigt die minimale Feldintensität von -0.2043 bei einem Winkel von 28.0679° an. (Dieser Winkel unterscheidet sich von der vorigen Lösung um 0.0001° .)

Verwendung einer wiederholten Näherung

Die im folgenden dargestellte Technik ist nützlich, wenn die Berechnung der Ableitung Schwierigkeiten bereitet. Es ist jedoch eine langsame Methode, da sie wiederholte Verwendung der Taste **SOLVE** erfordert. Andererseits müssen Sie keinen brauchbaren Wert für Δ finden wie bei der vorigen Methode. Um das relative Minimum einer Funktion zu finden, definieren Sie eine neue Funktion

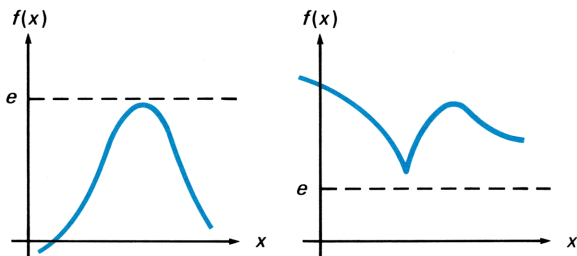
$$g(x) = f(x) - e$$

wobei e eine Zahl in der unmittelbaren Nähe des geschätzten Extremwertes der Funktion $f(x)$ ist. Wenn e geeignet gewählt wurde, wird $g(x)$ in der Umgebung des Extremwertes von $f(x)$ gegen Null streben, aber nicht gleich Null werden. Verwenden Sie dann **SOLVE**, um $g(x)$ in der Umgebung des Extremwertes zu untersuchen. Das gewünschte Ergebnis ist **Error 8**.

- Wenn **Error 8** angezeigt wird, ist der Wert im X-Register ein x -Wert in der Nähe des Extremwertes. Die Zahl im Z-Register gibt ungefähr an, wie weit e vom Extremwert von $f(x)$ entfernt ist.

Modifizieren Sie e , um es näher an den (aber nicht gleich dem) Extremwert zu bringen. Benutzen Sie dann nochmals **SOLVE**, um das neue $g(x)$ für den zuvor gefundenen Wert von x zu testen.

- Wenn eine Nullstelle von $g(x)$ gefunden wurde, liegt entweder e nicht jenseits dem Extremwert von $f(x)$, oder **SOLVE** hat ein anderes Gebiet gefunden, wo $f(x)$ gleich e ist. Wählen Sie einen neuen Wert für e , so daß er nahe, aber jenseits dem Extremwert von $f(x)$ liegt, und benutzen Sie **SOLVE** nochmals. Sie können jedoch auch $g(x)$ modifizieren, um die weiter entfernte Nullstelle zu eliminieren.



24 Abschnitt 1: Wirkungsvolle Verwendung von **SOLVE**

Beispiel: Lösen Sie das vorige Beispiel, ohne die Ableitung der relativen Feldintensität E zu berechnen.

Das Unterprogramm zur Berechnung von E und die benötigten Konstanten wurden im vorherigen Beispiel bereits eingegeben.

Geben Sie im Programm-Modus ein Unterprogramm ein, das einen geschätzten Extremwert von der Feldintensität subtrahiert. Der Extremwert sollte in einem Register gespeichert werden, so daß er gegebenenfalls manuell verändert werden kann.

Tastenfolge	Anzeige	
[g] [P/R]	000–	Programm-Modus.
[f] [LBL] 1	001–42,21, 1	Label des Unterprogramms.
[GSB] [B]	002– 32 12	Berechnet E .
[RCL] 9	003– 45 9	
[–]	004– 30	Subtrahiert den geschätzten Extremwert.
[g] [RTN]	005– 43 32	

Schätzen Sie die Minimumsintensität im Run-Modus durch manuelle Annäherung der Funktion ab.

Tastenfolge	Anzeige	
[g] [P/R]		Run-Modus.
10 [f] [→RAD]	0.1745	} Wertet die Funktion bei 10°, 30°, 50°, ... aus.
[ENTER] [f] [B]	–0.1029	
30 [f] [→RAD]	0.5236	
[ENTER] [f] [B]	–0.2028	
50 [f] [→RAD]	0.8727	
[ENTER] [f] [B]	0.0405	

Basierend auf diesen Näherungen, verwenden Sie -0.25 als geschätzten Extremwert und für **SOLVE** die Anfangsnäherungen von ungefähr 10° und 30° (in Radiant).

Tastenfolge	Anzeige	
.25 CHS STO 9	-0.2500	Speichert die Schätzung des Extremwertes.
.2 ENTER	0.2000	
.6	0.6	Anfangsnäherungen.
f SOLVE 1	Error 8	Keine Nullstelle wurde gefunden.
← STO 4	0.4849	Speichert die letzte θ -Näherung.
R↓ STO 5	0.4698	Speichert die vorletzte θ -Näherung.
R↓	0.0457	Abstand vom Extremwert.
.9 x	0.0411	Verändert den geschätzten Extremwert um 90 Prozent des Abstandes.
STO + 9	0.0411	
RCL 4	0.4849	Ruft die letzte θ -Näherung zurück.
ENTER ENTER f B	-0.2043	Berechnet die Intensität E .
←	0.0000	Ruft die vorletzte θ -Näherung zurück, erhält aber die letzte im Y-Register.
RCL 5	0.4698	
f SOLVE 1	Error 8	Keine Nullstelle wurde gefunden.
←	0.4898	Letzte θ -Näherung.
x$\geq$y	0.4893	Vorletzte θ -Näherung.
x$\geq$y	0.4898	Ruft die letzte θ -Näherung zurück.
ENTER ENTER f B	-0.2043	Berechnet die Intensität E .
x$\geq$y	0.4898	Ruft den θ -Wert zurück.
g →DEG	28.0660	Winkel in Altgrad.
g DEG	28.0660	Schaltet den Rechner wieder in den Altgrad-Modus.

Die zweite Wiederholung liefert zwei Näherungen für θ , die sich in der vierten Dezimalstelle unterscheiden. Die Feldintensität E der zwei Iterationen sind auf

vier Stellen hinter dem Komma identisch. Stoppt man das Programm an dieser Stelle, erhält man eine minimale Feldintensität von -0.2043 unter einem Winkel von 28.0660° . (Dieser Winkel weicht von früheren Lösungen um ungefähr 0.002° ab.)

Anwendungen

Die folgenden Anwendungen verdeutlichen, wie Sie **SOLVE** dazu verwenden können, normalerweise komplizierte Berechnungen – Finden von Zinsen, die nicht direkt berechnet werden können – zu vereinfachen. Andere Anwendungen von **SOLVE** sind in Abschnitt 3 und 4 aufgeführt.

Annuitäten und Zinseszinsen

Dieses Programm löst eine Vielzahl von finanzmathematischen Problemen, die auf den Größen Geld, Zeit und Zins basieren. In diesen Problemen kennen Sie meist den Wert von drei der folgenden vier Variablen und wollen den Wert der verbleibenden Variablen berechnen:

- | | |
|------------|--|
| <i>n</i> | Die <i>Anzahl</i> der Abrechnungsperioden. (Beispielsweise gilt für ein Darlehen über dreißig Jahre bei monatlicher Abzahlung $n = 12 \times 30 = 360$.) |
| <i>i</i> | Der <i>Zinssatz</i> pro Abrechnungszeitraum ausgedrückt in Prozent. (Um <i>i</i> zu berechnen, dividieren Sie den Jahreszinssatz durch die Anzahl von Abrechnungsperioden pro Jahr.) |
| <i>PV</i> | Der <i>Barwert</i> ist die ursprüngliche Zahlung oder der Gegenwartswert einer Reihe zukünftiger Zahlungen. |
| <i>PMT</i> | Wiederkehrende <i>Zahlung</i> pro Abrechnungsperiode. |
| <i>FV</i> | Letzte Zahlung oder <i>Endwert</i> einer Reihe früherer Zahlungen. |

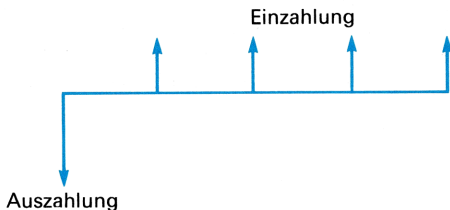
Mögliche Aufgabenstellungen mit Annuitäten und Zinseszinsen

Zulässige Variablenkombinationen	Typische Anwendungen		Vorbereitende Maßnahmen
	Nachschüssige Zahlungsweise	Vorschüssige Zahlungsweise	
n, i, PV, PMT (Geben Sie drei Größen ein und berechnen Sie die vierte.)	Darlehen mit direkter Rückzahlung. Diskontierte Schuldverschreibungen. Hypotheken.	Leasing-Verträge. Ratenzahlungen.	Verwenden Sie f CLEAR REG oder setzen Sie FV Null.
n, i, PV, PMT, FV (Geben Sie vier Größen ein und berechnen Sie die fünfte.)	Darlehen und Schuldverschreibungen mit kumulierter Rückzahlung.	Leasing-Verträge mit Restwerten. Ratenzahlungen.	Keine.
n, i, PMT, FV (Geben Sie drei Größen ein und berechnen Sie die vierte.)	Tilgungsfonds.	Ratensparverträge. Versicherungen.	Verwenden Sie f CLEAR REG oder setzen Sie PV Null.
n, i, PV, FV (Geben Sie drei Größen ein und berechnen Sie die vierte.)	Wertsteigerungen. Sparguthaben.		Verwenden Sie f CLEAR REG oder setzen Sie PMT Null.

Das Programm läßt sowohl vorschüssige als auch nachschüssige Zahlungsweise zu. Nachschüssige Zahlungen (Zahlungen am Ende der Abrechnungsperiode) findet man häufig bei Anschaffungsdarlehen und Hypotheken. Vorschüssige Zahlungen (Zahlungen am Anfang der Abrechnungsperiode) sind bei Leasingverträgen üblich. Bei Zahlungen am Ende einer Periode ist Flag 0 zu löschen; bei Zahlungen zu Beginn einer Periode ist Flag 0 zu setzen. Wenn die Aufgabenstellung keine Zahlungen beinhaltet, ist der Zustand von Flag 0 ohne Bedeutung.

Dieses Programm verwendet die Konvention, daß Auszahlungen als negative Werte eingegeben und angezeigt werden und Einzahlungen als positive Werte eingegeben und angezeigt werden.

Ein finanzmathematisches Problem kann gewöhnlich durch einen Zahlungsstrahl dargestellt werden. Dies ist eine grafische Darstellung des Zeitpunkts und der Richtung von finanziellen Transaktionen von Zahlungen. Das Diagramm hat eine waagrechte Zeitlinie, die in Abrechnungsperioden unterteilt ist – z.B. Monate oder Jahre. Ein- oder Auszahlungen von Geldbeträgen werden durch senkrechte Pfeile markiert, die der Konvention folgen, daß ein nach oben weisender Pfeil (positiv) eine Einzahlung und ein nach unten weisender Pfeil (negativ) eine Auszahlung repräsentiert. (Die folgenden Beispiele werden jeweils durch einen Zahlungsstrahl veranschaulicht.)



Durch Drücken von **f** **CLEAR** **REG** können Sie den Rechner auf eine neue Aufgabenstellung vorbereiten. Es ist jedoch nicht notwendig, die Register bei jeder neuen Aufgabenstellung zu löschen; Sie müssen nur die Werte derjenigen Variablen eingeben, die sich von Problem zu Problem verändern. Wenn eine Variable in einem Problem nicht vorkommt, geben Sie einfach den Wert Null für diese Variable ein. Wird *PMT* zum Beispiel in einem Problem verwendet, aber nicht im nächsten, geben Sie Null als Wert für *PMT* in dem zweiten Problem ein.

Die den finanzmathematischen Berechnungen zugrundeliegende Gleichung lautet:

$$PV + \frac{PMT \cdot A}{i/100} [1 - (1 + i/100)^{-n}] + FV(1 + i/100)^{-n} = 0$$

wobei $i \neq 0$ und

$$A = \begin{cases} 1 & \text{bei nachschüssiger Zahlungsweise.} \\ 1 + i/100 & \text{bei vorschüssiger Zahlungsweise.} \end{cases}$$

Das Programm hat folgende Charakteristika:

- **SOLVE** wird zur Berechnung von i benutzt. Da dies eine iterative Funktion ist, nimmt die Berechnung von i mehr Zeit in Anspruch als die Berechnung der übrigen Variablen. Es gibt jedoch Probleme, die nicht mit dieser Technik gelöst werden können. Wenn **SOLVE** keine Nullstelle finden kann, wird **Error 4** angezeigt.
- Bei der Berechnung einer der links aufgelisteten Variablen führen folgende Bedingungen zu einer **Error 4** Anzeige:

n	$PMT = -PV \cdot i / (100 \cdot A)$ $(PMT \cdot A - FV \cdot i / 100) / (PMT \cdot A + PV \cdot i / 100) \leq 0$ $i \leq -100$
i	SOLVE kann keine Wurzel finden.
PV	$i \leq -100$
PMT	$n = 0$ $i = 0$ $i \leq -100$
FV	$i \leq -100$

- Wenn in einem Problem ein Zinssatz von 0% auftaucht, erfolgt die Anzeige **Error 0** (oder **Error 4**, wenn Sie versuchen nach PMT aufzulösen).
- Probleme mit extrem großen ($> 10^6$) oder extrem kleinen ($< 10^{-6}$) Werten für n und i liefern sinnlose Ergebnisse.
- Zinsberechnungen mit einer Schlußzahlung, die ein entgegengesetztes Vorzeichen als die wiederkehrenden Zahlungen haben, können mehr als nur eine mathematisch korrekte Lösung (oder überhaupt keine Lösung) haben. Das Programm kann eine der Lösungen finden, zeigt jedoch keine anderen Möglichkeiten auf.

Tastenfolge

g **P/R**

f **CLEAR** **PRGM**

Anzeige

000-

Programm-Modus.

Tastenfolge

f **LBL** **A**
STO 1
R/S
GSB 1
g **LSTx**
RCL **x** 0
RCL 5
x $\geq$ y
-
g **LSTx**
RCL **+** 3
g **x=0**
GTO 0
 $\div$
CHS
g **TEST** 4
GTO 0
g **LN**
RCL 6
g **LN**
 $\div$
STO 1
g **RTN**
f **LBL** **B**
STO 2
R/S
 $\cdot$
2
ENTER
EEX

Anzeige

001–42,21,11 *n*-Routine.
002– 44 1 Speichert *n*.
003– 31
004– 32 1 Berechnet *n*.
005– 43 36
006–45,20, 0
007– 45 5
008– 34
009– 30 Berechnet $FV - 100 \text{ PMT } A/i$.
010– 43 36
011–45,40, 3 Berechnet $PV + 100 \text{ PMT } A/i$.
012– 43 20 Prüft, ob $\text{PMT} = -PV \cdot i/(100 \cdot A)$.
013– 22 0
014– 10
015– 16
016–43,30, 4 Prüft, ob $x \leq 0$.
017– 22 0
018– 43 12
019– 45 6
020– 43 12
021– 10
022– 44 1
023– 43 32
024–42,21,12 *i* Routine.
025– 44 2 Speichert *i*.
026– 31
027– 48
028– 2
029– 36
030– 26

Tastenfolge
Anzeige

CHS	031–	16	
3	032–	3	
g CF 1	033–43, 5, 1		Löscht Flag 1 für das SOLVE Unterprogramm.
f SOLVE 3	034–42,10, 3		
GTO 4	035–	22 4	
GTO 0	036–	22 0	
f LBL 4	037–42,21, 4		
EEX	038–	26	
2	039–	2	
x	040	20	Berechnet <i>i</i> .
STO 2	041–	44 2	
g RTN	042–	43 32	
f LBL C	043–42,21,13		<i>PV</i> Routine.
STO 3	044–	44 3	Speichert <i>PV</i> .
R/S	045–	31	
GSB 1	046–	32 1	Berechnet <i>PV</i> .
GSB 2	047–	32 2	
CHS	048–	16	
STO 3	049–	44 3	
g RTN	050–	43 32	
f LBL D	051–42,21,14		<i>PMT</i> Routine.
STO 4	052–	44 4	Speichert <i>PMT</i> .
R/S	053–	31	
1	054–	1	Berechnet <i>PMT</i> .
STO 4	055–	44 4	
GSB 1	056–	32 1	
RCL 3	057–	45 3	
GSB 2	058–	32 2	
$x \geq y$	059–	34	
÷	060–	10	
CHS	061–	16	
STO 4	062–	44 4	
g RTN	063–	43 32	

Tastenfolge

[f] [LBL] [E]

[STO] 5

[R/S]

[GSB] 1

[RCL] [+] 3

[RCL] [÷] 7

[CHS]

[STO] 5

[g] [RTN]

[f] [LBL] 1

[g] [SF] 1

1

[RCL] 2

[g] [%]

[f] [LBL] 3

[STO] 8

1

[STO] 0

[+]

[g] [TEST] 4

[GTO] 0

[STO] 6

[g] [F?] 0

[STO] 0

[RCL] 1

[CHS]

[y^x]

[STO] 7

1

[x^zy]

[-]

Anzeige

064-42,21,15

FV Routine.

065- 44 5

Speichert *FV*.

066- 31

067- 32 1

Berechnet *FV*.

068-45,40, 3

069-45,10, 7

070- 16

071- 44 5

072- 43 32

073-42,21, 1

074-43, 4, 1

Setzt Flag 1 für das Unterprogramm 3.

075- 1

076- 45 2

077- 43 14

Berechnet $i/100$.

078-42,21, 3

SOLVE Unterprogramm.

079- 44 8

080- 1

081- 44 0

082- 40

083-43,30, 4

Prüft ob $i \leq 100$.

084- 22 0

085- 44 6

086-43, 6, 0

Prüft auf nachschüssige Zahlungsweise.

087- 44 0

088- 45 1

089- 16

090- 14

Berechnet $(1 + i/100)^{-n}$.

091- 44 7

092- 1

093- 34

094 - 30

Berechnet $1 - (1 + i/100)^{-n}$.

Tastenfolge**Anzeige**

[g] [x=0]	095– 43 20	Prüft, ob $i = 0$ oder $n = 0$.
[GTO] 0	096– 22 0	
[RCL] [x] 0	097–45,20, 0	
[RCL] 4	098– 45 4	
[RCL] [÷] 8	099–45,10, 8	
[x]	100– 20	
[g] [F?] 1	101–43, 6, 1	Prüft, ob Flag 1 gesetzt ist.
[g] [RTN]	102– 43 32	
[RCL] [+] 3	103–45,40, 3	[SOLVE] Unterprogramm geht weiter.
[f] [LBL] 2	104–42,21, 2	
[RCL] 5	105– 45 5	
[RCL] [x] 7	106–45,20, 7	Berechnet $FV(1 + i/100)^{-n}$.
[+]	107– 40	
[g] [RTN]	108– 43 32	Ende des [SOLVE] Unterprogramms.

Verwendete Labels: A, B, C, D, E, 0, 1, 2 und 3.

Verwendete Register: $R_0(A)$, $R_1(n)$, $R_2(i)$, $R_3(PV)$, $R_4(PMT)$, $R_5(FV)$, R_6 , R_7 und R_8 .

Um das Programm zu benutzen,

1. Drücken Sie 8 **[f]** **[DIM]** **[(i)]**, um R_0 bis R_8 zu reservieren.
2. Drücken Sie **[f]** **[USER]**, um den User-Modus zu aktivieren.
3. Drücken Sie, falls nötig **[f]** **[CLEAR]** **[REG]**, um alle finanzmathematischen Variablen zu löschen. Wenn Sie in alle Finanzregister neue Werte speichern werden, brauchen Sie diese zuvor nicht zu löschen.
4. Setzen Sie Flag 0 je nach Art der Zahlung:
 - Drücken Sie **[g]** **[CF]** 0 für Zahlungen am Ende einer Periode.
 - Drücken Sie **[g]** **[SF]** 0 für Zahlungen am Beginn einer Periode.
5. Geben Sie die bekannten Werte der Finanzvariablen ein:
 - Um n einzugeben, tasten Sie den Wert ein und drücken **[A]**.
 - Um i einzugeben, tasten Sie den Wert ein und drücken **[B]**.

34 Abschnitt 1: Wirkungsvolle Verwendung von SOLVE

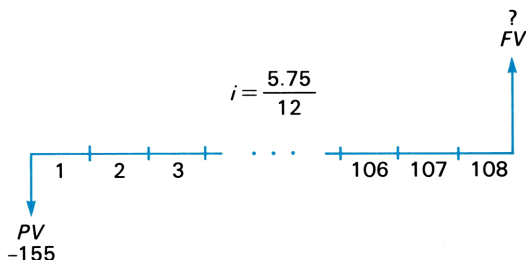
- Um PV einzugeben, tasten Sie den Wert ein und drücken C.
- Um PMT einzugeben, tasten Sie den Wert ein und drücken D.
- Um FV einzugeben, tasten Sie den Wert ein und drücken E.

6. Berechnen Sie die unbekannten Werte:

- Um n zu berechnen, drücken Sie A R/S.
- Um i zu berechnen, drücken Sie B R/S.
- Um PV zu berechnen, drücken Sie C R/S.
- Um PMT zu berechnen, drücken Sie D R/S.
- Um FV zu berechnen, drücken Sie E R/S.

7. Um ein anderes Problem zu lösen, wiederholen Sie Schritt 3 bis 6 so oft wie nötig. Vergewissern Sie sich zuvor, daß jede Variable, die nicht in dem Problem benötigt wird, den Wert Null hat.

Beispiel: Sie zahlen 155 DM auf ein Sparkonto ein, das einen Jahreszins von $5\frac{3}{4}\%$ hat und monatlich abgerechnet wird. Welchen Betrag können Sie nach neun Jahren abheben?



Tastenfolge

g P/R
f CLEAR REG
f FIX 2
f USER
g CF 0
9 ENTER 12 x A

Anzeige

108.00

Run-Modus.

Löscht die finanzmathematischen Variablen.

Aktiviert den User-Modus.

Nachschüssige Zahlungsweise.

Gibt $n = 9 \times 12$ ein.

Tastenfolge

 5.75 **ENTER** 12 **÷** **B**

 155 **CHS** **C**
E **R/S**
Anzeige
0.48
-155.00
259.74

 Gibt $i = 5.75/12$ ein.

 Gibt $PV = -155$ (es handelt sich um eine Auszahlung) ein.

 Berechnet FV .

Wenn Sie eine Summe von 275 DM am Ende der neun Jahre abheben wollen, wie hoch muß der Zinssatz dann sein?

Tastenfolge

 275 **E**
B **R/S**

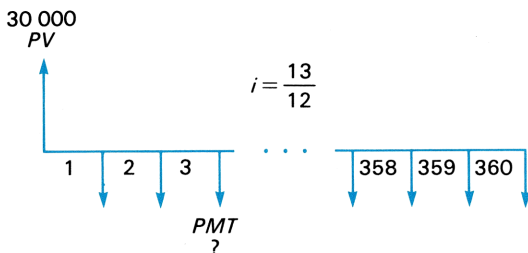
 12 **×**
Anzeige
275.00
0.53
6.39

 Gibt $FV = 275$ ein.

 Berechnet i .

Berechnet den jährlichen Zins.

Beispiel: Sie nehmen eine Hypothek in Höhe von 30 000 DM mit einer Laufzeit von 30 Jahren und einen Zinssatz von 13% auf. Wie groß muß Ihre monatliche Zahlung an die Bank sein, um die Hypothek vollständig abzutragen?


Tastenfolge
f **CLEAR** **REG**

 30 **ENTER** 12 **×** **A**

 13 **ENTER** 12 **÷** **B**

 30000 **C**
D **R/S**
Anzeige
360.00
1.08
30,000.00
-331.86

Löscht die finanzmathematischen Variablen.

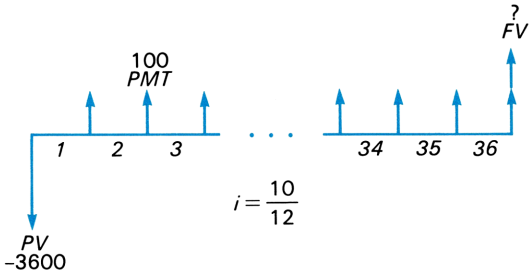
 Gibt $n = 30 \times 12$ ein.

 Gibt $i = 13/12$ ein.

 Gibt $PV = 30000$ ein.

 Berechnet PMT (das Minuszeichen deutet eine Auszahlung an).

Beispiel: Sie bieten ein Darlehen von 3600 DM an, das innerhalb von 36 Monaten mit monatlichen Zahlungen von 100 DM und einem Jahreszinssatz von 10% getilgt werden muß. Welche Schlußzahlung muß zusammen mit der 36. Zahlung geleistet werden, um das Darlehen vollständig zurückzuzahlen?



Tastenfolge

Anzeige

f CLEAR REG

Löscht die finanzmathematischen Variablen.

36 A

36.00

Gibt $n = 36$ ein.

10 ENTER 12 ÷ B

0.83

Gibt $i = 10/12$ ein.

3600 CHS C

-3600.00

Gibt $PV = -3600$ (Auszahlung) ein.

100 D

100.00

Gibt $PMT = 100$ (Einzahlung) ein.

E R/S

675.27

Berechnet FV .

Die letzte Zahlung ist $675.27 \text{ DM} + 100.00 \text{ DM} = 775.27 \text{ DM}$, da die letzte Rate und die Schlußzahlung am Ende der letzten Periode fällig sind.

Beispiel: Sie erhalten ein Darlehen von 50 000 DM mit einem Jahreszinssatz von 14% über einen Zeitraum von 360 Monaten. Finden Sie den Restbetrag nach der 24. Zahlung und den zwischen der 12. und 24. Zahlung aufgelaufenen Zins.

Sie können das Programm verwenden, um den aufgelaufenen Zins und den Restbetrag des Darlehens zu berechnen. Der aufgelaufene Zins ist gleich der gesamten, während der Periode geleisteten Zahlung minus der Schuldsommenverringerung. Die Schuldsommenverringerung ist die Differenz zwischen der Restschuld am Anfang und am Ende der Periode.

Berechnen Sie zuerst die Abzahlung auf das Darlehen.

Tastenfolge	Anzeige	
f CLEAR REG		Löscht die finanzmathematischen Variablen.
360 A	360.00	Gibt $n = 360$ ein.
14 ENTER 12 ÷ B	1.17	Gibt $i = 14/12$ ein.
50000 CHS C	-50.000.00	Gibt $PV = -50\,000$ ein.
D R/S	592.44	Berechnet PMT .

Berechnen Sie nun die verbliebene Restschuld für den 24. Monat:

Tastenfolge	Anzeige	
24 A	24.00	Gibt $n = 24$ ein.
E R/S	49,749.56	Berechnet FV für den 24. Monat.

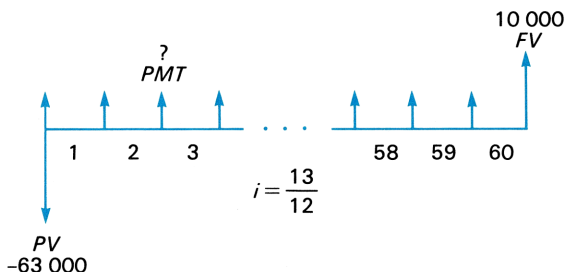
Speichern Sie sich diese Restschuld und berechnen Sie dann die Restschuld nach 12 Monaten und die Schuldsommenverringering zwischen dem 12. und dem 24. Monat.

Tastenfolge	Anzeige	
STO I	49,749.56	
12 A	12.00	Gibt $n = 12$ ein.
E R/S	49,883.48	Berechnet FV für den 12. Monat.
RCL I	49,749.56	Ruft FV für den 24. Monat zurück.
-	133.92	Berechnet die Schuldsommenverringering.

Der aufgelaufene Zins ist der Wert der 12 Zahlungen minus der Schuldsommenverringering.

Tastenfolge	Anzeige	
RCL 4	592.44	Ruft PMT zurück.
12 ×	7,109.23	Berechnet den Wert der Zahlung.
x$\geq$y -	6,975.31	Berechnet den aufgelaufenen Zins.

Beispiel: Eine Leasingfirma erwägt den Kauf eines Minicomputers zum Preis von 63 000 DM und möchte aus dem Leasing des Computers über einen Zeitraum von fünf Jahren eine Rendite von 13% erzielen. Die Firma erwartet, den Computer am Ende der Leasingperiode für mindestens 10 000 DM wieder verkaufen zu können. Wie hoch muß die monatliche Leasinggebühr sein, damit die Rendite von 13% erzielt wird? (Da die Leasinggebühr am Anfang des Monats fällig ist, müssen Sie Flag 0 setzen, um den Rechner vorschüssige Zahlungsweise zu signalisieren.)

**Tastenfolge**

[f] CLEAR [REG]

[g] [SF] 0

5 [ENTER] 12 [x] [A]

13 [ENTER] 12 [÷] [B]

63000 [CHS] [C]

10000 [E]

[D] [R/S]

Anzeige**60.00****1.08****-63,000.00****10,000.00****1,300.16**

Löscht die finanzmathematischen Variablen.

Spezifiziert vorschüssige Zahlungsweise.

Gibt $n = 5 \times 12$ ein.Gibt $i = 13/12$ ein.Gibt $PV = -63\,000$ ein.Gibt $FV = 10\,000$ ein.Berechnet PMT .

Wenn sich der Preis des Computers auf 70 000 DM erhöht, wie hoch muß die monatliche Leasinggebühr in diesem Fall sein?

Tastenfolge

70000 [CHS] [C]

[D] [R/S]

Anzeige**-70,000.00****1,457.73**Gibt $PV = -70\,000$ ein.Berechnet PMT .

Wenn man die Zahlungen auf 1500 DM erhöht, wie groß ist die Rendite in diesem Fall?

Tastenfolge	Anzeige	
1500 D	1,500.00	Gibt $PMT = 1500$ ein.
B R/S	1.18	Berechnet i (monatlich).
12 x	14.12	Berechnet die jährliche Rendite.
f USER	14.12	Desaktiviert den User-Modus.

Analyse diskontierter Zahlungsströme

Dieses Programm berechnet zwei Kenngrößen zur Analyse diskontierter Zahlungsströme: NPV («net present value» oder Nettobarwert) und IRR («internal rate of return» oder interne Verzinsung). Die Werte von NPV und IRR können für bis zu 24 Gruppen von Zahlungen berechnet werden.

Die Zahlungen werden in einer zweiseptigen Matrix **C** abgelegt, wobei jede Zeile jeweils eine Zahlungsgruppe aufnimmt. Die Zahlungen innerhalb einer Gruppe müssen aufeinander folgen und jeweils den gleichen Betrag haben. Das erste Element einer Zeile nimmt dann den Zahlungsbetrag und das zweite Element die Anzahl der Zahlungen auf. Das allererste Element von **C** muß den Betrag der ursprünglichen Zahlung enthalten. Die Zahlungen müssen in gleichen Zeitabständen erfolgen; wenn für mehrere Perioden keine Zahlung auftritt, ist in die entsprechende Zeile von **C** der Wert 0 als Zahlungsbetrag und die Anzahl der Nullzahlungen einzutragen.

Nachdem alle Zahlungen in der Matrix **C** gespeichert sind, können Sie einen angenommenen Zinssatz eingeben und den Nettobarwert (NPV) der Investition berechnen. Zusätzlich können Sie den Wert der internen Verzinsung (IRR) berechnen. Die IRR ist der Zinssatz, bei der der Nettobarwert gleich Null wird.

Die grundlegende Gleichung zur Berechnung von NPV lautet

$$NPV = \begin{cases} \sum_{j=1}^k CF_j \left(\frac{1 - (1 + i/100)^{-n_j}}{i/100} \right) (1 + i/100)^{-\sum_{l < j} n_l} & \text{für } i > -100 \\ & i \neq 0 \\ \sum_{j=1}^k CF_j n_j & \text{für } i = 0 \end{cases}$$

wobei $\sum_{l < 1} n_l$ als -1 definiert ist.

40 Abschnitt 1: Wirkungsvolle Verwendung von **SOLVE**

Wie im letzten Programm gilt auch hier die Übereinkunft, daß Einzahlungen als positive Zahlen und Auszahlungen als negative Zahlen eingegeben und angezeigt werden.

Das Programm hat folgende Charakteristika:

- Die Zahlungsfolge (einschließlich der Anfangsinvestition) muß sowohl den positiven als auch den negativen Zahlungsfluß enthalten; d.h. es muß mindestens ein Vorzeichenwechsel erfolgen.
- Zahlungsfolgen mit mehreren Vorzeichenwechsel haben mehr als eine Lösung. Dieses Programm kann eine Lösung finden, aber es kann keine weiteren Möglichkeiten aufzeigen.
- Die Berechnung von *IRR* kann mehrere Minuten in Anspruch nehmen (fünf und mehr), je nach Anzahl der Zahlungen.
- Das Programm zeigt **Error 4** an, wenn es keine Lösung für *IRR* finden kann oder wenn bei der Berechnung von *NPV* $i \leq -100\%$ gilt.

Tastensequenz

g **P/R**

f **CLEAR** **PRGM**

f **LBL** **A**

EEX

2

÷

GSB 2

R/S

f **LBL** **B**

1

ENTER

EEX

CHS

3

f **SOLVE** 2

GTO 1

GTO 0

f **LBL** 1

EEX

Anzeige

000-

001-42,21,11

002- 26

003- 2

004- 10

005- 32 2

006- 31

007-42,21,12

008- 1

009- 36

010- 26

011- 16

012- 3

013-42,10, 2

014- 22 1

015- 22 0

016-42,21, 1

017- 26

Programm-Modus.

NPV Routine.

Berechnet *IRR*/100.

IRR Routine.

Programmverzweigung falls keine Lösung für *IRR* gefunden wurde.

Tastenfolge
Anzeige

2	018–	2
x	019–	20
R/S	020–	31
f LBL 2	021–42,21, 2	
g CF 0	022–43, 5, 0	
STO 2	023–	44 2
1	024–	1
STO 4	025–	44 4
+	026–	40
g TEST 4	027–43,30, 4	
GTO 0	028–	22 0
STO 3	029–	44 3
0	030–	0
STO 5	031–	44 5
f MATRIX 1	032–42,16, 1	
f LBL 3	033–42,21, 3	
g F? 0	034–43, 6, 0	
GTO 7	035–	22 7
GSB 6	036–	32 6
RCL 2	037–	45 2
g x=0	038–	43 20
GTO 4	039–	22 4
1	040–	1
+	041–	40
GSB 6	042–	32 6
CHS	043–	16
y^x	044–	14
STO 4	045–	44 4
1	046–	1
x≥y	047–	34
–	048–	30

 Berechnet NPV .

 Berechnet $1 + IRR/100$.

 Prüft, ob $IRR \leq -100$.

 Programmverzweigung für $IRR \leq -100$.

Prüft, ob alle Zahlungsströme gebraucht werden.

Programmverzweigung, falls alle Zahlungsströme gebraucht werden.

 Prüft, ob $IRR = 0$.

 Programmverzweigung, falls $IRR = 0$.

Tastenfolge
RCL $\div$ 2

RCL $\times$ 3

GTO 5

f **LBL** 4

x $\geq$ y
GSB 6

f **LBL** 5

x
STO $+$ 5

RCL 4

STO $\times$ 3

GTO 3

f **LBL** 6

f **USER** **RCL** **C**
f **USER**
g **RTN**
g **SF** 0

g **RTN**
f **LBL** 7

RCL 5

g **RTN**
Anzeige

049–45,10, 2

050–45,20, 3

051– 22 5

052–42,21, 4

053– 34

054– 32 6

055–42,21, 5

056– 20

057–44,40, 5

058– 45 4

059–44,20, 3

060– 22 3

061–42,21, 6

062u 45 13

063– 43 32

064–43, 4, 0

065– 43 32

066–42,21, 7

067– 45 5

068– 43 32

Ruft das Zahlungselement zurück.

Setzt Flag 0, wenn das letzte Element auftritt.

Ruft *NPV* zurück.

Verwendete Labels: A, B und 0 bis 7.

Verwendete Register: R₀ bis R₅.

Verwendete Matrix: C.

Um das Programm für die Analyse von diskontierten Zahlungsströmen zu verwenden:

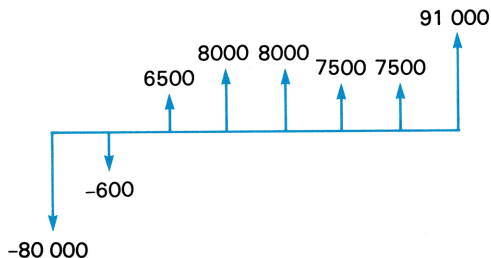
1. Drücken Sie 5 **f** **DIM** **(0)**, um die Register R₀ bis R₅ neu aufzuteilen.
2. Drücken Sie **f** **USER**, um den User-Modus zu aktivieren (wenn nicht bereits geschehen).
3. Geben Sie die Anzahl der Zahlungsgruppen ein und drücken Sie dann **ENTER** 2 **f** **DIM** **C**, um die Matrix C zu dimensionieren.
4. Drücken Sie **f** **MATRIX** 1, um den Zeilen- und Spaltenindex auf 1 zu setzen.
5. Für jede Zahlungsgruppe:
 - a. Tasten Sie den Betrag ein und drücken Sie **STO** **C**,
 - b. Tasten Sie dann ein, aus wieviel Zahlungen diese Gruppe besteht und drücken Sie **STO** **C**.

6. Berechnen Sie den gewünschten Parameter:

- Um *IRR* zu berechnen, drücken Sie **B**.
- Um *NPV* zu berechnen, geben Sie den Zinssatz *i* ein und drücken **A**.
Wiederholen Sie diesen Schritt für so viele Zinssätze wie nötig.

7. Wiederholen Sie die Schritte 3 bis 6 für andere Zahlungsströme.

Beispiel: Ein Investor kauft ein Zweifamilienhaus für 80 000 DM, das er nach sieben Jahren wieder verkaufen will. Er muß im ersten Jahr etwas Geld für einige Reparaturen am Haus aufbringen. Am Ende des siebten Jahres verkauft er das Haus für 91 000 DM. Er erwartet eine Rendite von 9% nach Abzug der Steuern. Ist dies mit den folgenden Zahlungen (nach Steuern) zu erreichen?



Tastenfolge

g **P/R**

f **FIX** 2

5 **f** **DIM** **(i)**

6 **ENTER** 2

f **DIM** **C**

f **MATRIX** 1

f **USER**

80000 **CHS** **STO** **C**

1 **STO** **C**

600 **CHS** **STO** **C**

1 **STO** **C**

6500 **STO** **C**

1 **STO** **C**

8000 **STO** **C**

2 **STO** **C**

7500 **STO** **C**

2 **STO** **C**

91000 **STO** **C**

1 **STO** **C**

Anzeige

5.00

2

2.00

2.00

2.00

-80,000.00

1.00

-600.00

1.00

6,500.00

1.00

8,000.00

2.00

7,500.00

2.00

91,000.00

1.00

Run-Modus.

Reserviert die Register R₀ bis R₅.

Anfangsinvestition.

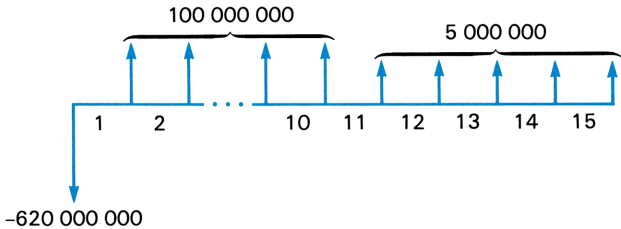
Tastenfolge	Anzeige	
9	9	Gibt die gewünschte Rendite ein.
A	-4,108.06	NPV.

Da der Wert von *NPV* negativ ist, wird durch diese Investition die gewünschte 9prozentige Rendite nicht erreicht. Berechnen Sie nun *IRR*.

Tastenfolge	Anzeige	
B	8.04	<i>IRR</i>
		(nach ungefähr 8 Minuten).

Die tatsächliche Verzinsung ist kleiner als die gewünschten 9%.

Beispiel: Von einer Investition in Höhe von 620 000 000 DM wird erwartet, daß sie für die nächsten 15 Jahre die in dem folgenden Diagramm gezeigten jährlichen Erträge liefert:



Wie hoch ist die erwartete Verzinsung?

Tastenfolge	Anzeige	
3 ENTER 2	2	
f DIM C	2.00	
f MATRIX 1	2.00	
620000000		
CHS STO C	-620,000,000.0	
1 STO C	1.00	
100000000 STO C	100,000,000.0	
10 STO C	10.00	
5000000 STO C	5,000,000.00	
5 STO C	5.00	
B	10.06	<i>IRR</i> .
f FIX 4	10.0649	
f USER	10.0649	Desaktiviert den User-Modus.

Arbeiten mit $\int \frac{x}{y}$

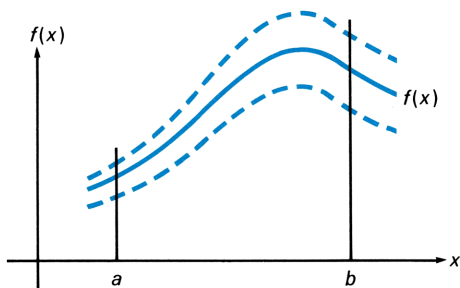
Mit Hilfe der Funktion $\int \frac{x}{y}$ des HP-15C können Sie beliebig vorgegebene Funktionen numerisch integrieren. Dieser Abschnitt zeigt Ihnen, wie Sie $\int \frac{x}{y}$ effektiv verwenden können und beschreibt Techniken, mit denen Sie schwierige Integrale meistern können.

Numerische Integration mit Hilfe von $\int \frac{x}{y}$

Ein Rechner kann mittels einer numerischen Integration fast nie ein Integral präzise berechnen. Die Funktion $\int \frac{x}{y}$ fragt Sie jedoch auf sehr komfortable Art und Weise, ein wie großer Fehler noch toleriert werden kann. Sie müssen das Anzeigeformat gemäß der Anzahl der genauen Stellen des Integranden $f(x)$ wählen. Tatsächlich bestimmt die Auswahl des Anzeigeformats die Breite eines Streifens um den Graphen von $f(x)$. Das durch $\int \frac{x}{y}$ abgeschätzte Integral ist proportional zu der Fläche unter einem nicht spezifizierten Graph, der vollständig innerhalb des Streifens liegt. Diese Schätzung kann um die Größe des Streifens selbst variieren, so daß $\int \frac{x}{y}$ also auch diese Fläche abschätzt. Ist I das gesuchte Integral, so gilt

$$I = \left(\begin{array}{l} \text{Fläche unter dem in dem} \\ \text{Streifen liegenden Graph} \end{array} \right) \pm \left(\begin{array}{l} \frac{1}{2} \text{ der Fläche} \\ \text{des Streifens} \end{array} \right)$$

Der HP-15C legt die erste Flächennäherung im X-Register und die zweite – die Fehlerabschätzung des Integrals – im Y-Register ab.



Beispielsweise kann $f(x)$ einen physikalischen Effekt beschreiben, dessen Wertigkeit nur mit einer Genauigkeit von ± 0.005 bestimmt werden kann. In diesem Fall hat auch der berechnete Wert von $f(x)$ eine Unsicherheit von 0.005. Benutzt man das Anzeigeformat $\left[\text{FIX}\right]$ 2, wird dem Rechner dadurch mitgeteilt, daß nur die ersten beiden Dezimalstellen von Bedeutung sind. Der Rechner braucht dann keine Zeit damit zu verschwenden, ohnehin ungenaue Stellen zu berechnen. Stattdessen kann Ihnen der Rechner in kurzer Zeit einen recht genauen Überblick über den Wertebereich geben, in dem das Integral liegen muß.

Der HP-15C schützt Sie nicht davor, daß Sie $f(x)$ als wesentlich genauer deklarieren als es wirklich ist. Sie können der Wahl des Anzeigeformats eine ausführliche Fehleranalyse vorausschicken, oder einfach raten. Sie können das Anzeigeformat $\left[\text{SCI}\right]$ 4 oder $\left[\text{FIX}\right]$ 4 auch beibehalten, ohne sich weitere Gedanken darüber zu machen. Der Rechner liefert Ihnen dann eine Näherung für das Integral und seine zugehörige Fehlerabschätzung. Diese Werte erlauben Ihnen dann, das Ergebnis genauer als ohne jegliche Zusatzinformation zu interpretieren.

Der $\left[\int\right]$ Algorithmus verwendet das Romberg-Verfahren zur Berechnung des Integrals. Verschiedene zusätzliche Verfeinerungen machen das implementierte Verfahren noch effektiver.

$\left[\int\right]$ benutzt Stützstellen mit ungleichen Abständen, da äquidistante Stützstellen bei periodischen Funktionen irreführende Ergebnisse liefern können. Die Abstandsbildung läßt sich beispielsweise durch Substitution von

$$x = \frac{3}{2}u - \frac{1}{2}u^3$$

in

$$I = \int_{-1}^1 f(x) dx = \int_{-1}^1 f\left(\frac{3}{2}u - \frac{1}{2}u^3\right) \frac{3}{2} (1 - u^2) du$$

bei Auswertung von u auf einem äquidistanten Gitter verdeutlichen. Außer dem Unterdrücken einer Resonanz hat diese Substitution noch zwei weitere Vorteile. Erstens muß der Integrand nicht an den Intervallrändern ausgewertet werden (außer wenn das Intervall so schmal ist, daß keine anderen Möglichkeiten zur Verfügung stehen). Die Folge davon ist, daß die Berechnung eines

$$\int_0^3 \frac{\sin x}{x} dx$$

Integrals der Form nicht wegen der Division durch Null an dem Endpunkt unterbrochen werden würde. Zweitens kann  Funktionen integrieren, die sich wie $\sqrt{|x-a|}$ verhalten, d.h. deren Steigung am Intervallrand unendlich ist. Solche Funktionen treten auf, wenn Sie Flächen berechnen, die von glatten, geschlossenen Kurven begrenzt werden.

Eine weitere Verfeinerung ist, daß  bei der Akkumulation der internen Summen eine höhere Genauigkeit verwendet, und zwar 13 signifikante Stellen. Dies ermöglicht, daß Tausende von Näherungen akkumuliert werden können (falls nötig), ohne daß durch Rundung mehr Information verloren geht, als dies innerhalb des Unterprogramms geschieht.

Genauigkeit der zu integrierenden Funktionen

Die Genauigkeit eines durch  berechneten Integrals hängt von der Genauigkeit der im Unterprogramm berechneten Funktion ab. Diese Genauigkeit, die Sie durch die Wahl des Anzeigeformats bestimmen, wird hauptsächlich von den folgenden drei Größen bestimmt:

- Die Genauigkeit der empirischen Konstanten der Funktion.
- Den Grad der Genauigkeit, mit dem die Funktion einen physikalischen Vorgang beschreibt.
- Die Größe des Rundungsfehlers in der internen Berechnung des Rechners.

Funktionen, die physikalische Vorgänge beschreiben

Funktionen wie $\cos(4\theta - \sin \theta)$ sind *rein mathematische Abbildungen*. In diesem Zusammenhang bedeutet das, daß die Funktionen keine empirischen Konstanten enthalten und weder die Variablen noch die Integrationsgrenzen repräsentieren physikalische Größen. Für diese Art von Funktionen können Sie so viele Stellen wie sie wollen (bis zu neun) über das Anzeigeformat spezifizieren, um den gewünschten Grad der Genauigkeit des Integrals* zu erhalten. Alles, was Sie tun müssen, ist zwischen der Genauigkeit und der hierfür nötigen Rechenzeit abzuwägen.

* Solange $f(x)$ trotz des Rundungsfehlers auf die Anzahl der angezeigten Ziffern genau berechnet wird.

Zusätzliche Überlegungen sind jedoch nötig, wenn Sie eine Funktion integrieren müssen, die einen tatsächlichen physikalischen Vorgang beschreibt. Grundsätzlich sollten Sie sich bei solchen Funktionen die Frage stellen, *ob die Genauigkeit, die Sie bei der Berechnung des Integrals erreichen wollen, durch die Genauigkeit der Funktion selbst gerechtfertigt ist*. Es ist z.B. sinnlos, mehr als drei Ziffern durch das Anzeigeformat zu spezifizieren, wenn die Funktion empirische Konstanten enthält, die nur auf drei Dezimalstellen genau angegeben sind.

Eine andere wichtige – und meist versteckte und demzufolge leicht zu übersehende Tatsache ist, daß beinahe alle Funktionen, die ein physikalisches System darstellen, *von Natur aus zu einem bestimmten Grad ungenau sind*, da sie nur ein mathematisches «Modell» eines tatsächlichen physikalischen Prozesses oder Ereignisses sind. Ein mathematisches Modell ist selbst eine *Näherung*, die den Effekt von bekannten oder unbekannten Faktoren, die nichts über das untersuchte physikalische Phänomen aussagen, derartig vernachlässigt, daß das Ergebnis nicht an Aussagekraft verliert.

Ein Beispiel für ein mathematisches Modell ist die *Normalverteilungsfunktion*

$$\int_{-\infty}^t \frac{e^{-(x-\mu)^2/2\sigma^2}}{\sigma\sqrt{2\pi}} dx$$

die sich als sehr nützlich für die Gewinnung von Informationen aus physikalischen Messungen an lebenden Organismen oder Messungen von Produktdimensionen oder Durchschnittstemperaturen usw. erwiesen hat. Solche mathematische Beschreibungen sind normalerweise entweder von theoretischen Überlegungen oder von experimentellen Daten abgeleitet. Um von praktischem Nutzen zu sein, sind sie mit bestimmten Annahmen verbunden, wie z.B. dem Vernachlässigen von relativ bedeutungslosen Faktoren. Beispielsweise hängt die Genauigkeit von Ergebnissen, die unter Verwendung der Normalverteilungsfunktion als Modell für die Verteilung von bestimmten Größen erhalten wurden, von der Größe der betrachteten Grundgesamtheit ab. Ein weiteres Beispiel ist die Gleichung $s = s_0 - \frac{1}{2}gt^2$, die die Höhe eines frei fallenden Körpers liefert. Bei dieser Gleichung wird ignoriert, daß die Gravitationsbeschleunigung g von der Höhe abhängt.

Das heißt also, daß mathematische Beschreibungen der physikalischen Welt uns nur Ergebnisse mit einer begrenzten Genauigkeit liefern können. Wenn Sie ein Integral berechnen, das eine scheinbare Genauigkeit hat, die größer ist

als die Genauigkeit, mit der das Modell das tatsächliche Verhalten beschreibt, ist es nicht sinnvoll, den berechneten Wert mit der gesamten berechneten, scheinbaren Genauigkeit zu verwenden.

Rundungsfehler in internen Berechnungen

Jeder Rechner – einschließlich des HP-15C – rundet berechnete Ergebnisse auf eine endliche Anzahl von Stellen (10 Stellen im Falle des HP-15C). Dieser *Rundungsfehler* kann sich fortpflanzen, und so ist es möglich, daß berechnete Ergebnisse – besonders Ergebnisse von Berechnungen einer Funktion, die verschiedene mathematische Operationen enthält – nicht auf alle zehn angezeigten Stellen genau sind. Beachten Sie, daß der Rundungsfehler die Berechnung *jedes* mathematischen Ausdrucks betrifft, nicht nur die Funktion, die mit  integriert werden soll. (Im Anhang finden Sie zusätzliche Informationen zu diesem Thema.)

Wenn $f(x)$ eine Funktion ist, die einen physikalischen Vorgang beschreibt, ist die Ungenauigkeit durch den Rundungsfehler unwesentlich im Vergleich zu den Unsicherheiten, die von empirischen Daten herrühren. Wenn $f(x)$ eine Funktion ist, die wir rein mathematisch genannt haben, ist ihre Genauigkeit nur durch den Rundungsfehler begrenzt. Um präzise zu bestimmen, wieviele Stellen einer berechneten Funktion von dem Rundungsfehler betroffen werden, bedarf es einer aufwendigen Analyse. In der Praxis werden solche Effekte üblicherweise (und hinreichend) mehr durch Erfahrung als durch eine Analyse bestimmt.

In bestimmten Situationen bewirken Rundungsfehler seltsame Ergebnisse, besonders, wenn Sie Ergebnisse der Berechnung von Integralen vergleichen, die mathematisch gleich sind, sich aber durch eine Variablentransformation unterscheiden. Es ist jedoch unwahrscheinlich, daß solche Situationen in typischen Anwendungen auftreten.

Verkürzung der Rechenzeit

Die Zeit, die  zur Berechnung eines Integrals benötigt, hängt davon ab, wie schnell eine gewisse Stützstellendichte in dem Gebiet, in dem die Funktion interessant ist, erreicht wird. Die Berechnung des Integrals einer Funktion verlängert sich, wenn das Integrationsintervall hauptsächlich Gebiete enthält, wo die Funktion nicht interessant ist. Glücklicherweise können Sie, wenn Sie ein solches Integral zu berechnen haben, das Problem so modifizieren, daß die Rechenzeit entscheidend verkürzt wird. Zwei solche Techniken sind die Unterteilung des Integrationsintervalls und die Transformation von Variablen.

Unterteilung des Integrationsintervalls

In Gebieten, wo sich die Steigung von $f(x)$ nennenswert verändert, ist eine hohe Dichte von Stützstellen nötig, um eine Näherung zu erhalten, die sich merklich von einer Iteration zur anderen verändert. In Bereichen jedoch, wo die Steigung der Funktion beinahe konstant bleibt, ist dies nicht nötig. Dies liegt daran, daß die Auswertung von zusätzlichen Stützstellen nicht viel neue Informationen über die Funktion liefert, so daß sich die Differenzen aufeinanderfolgender Näherungen nicht entscheidend verändern würden. Folglich kann in solchen Gebieten eine Näherung von vergleichbarer Genauigkeit mit wesentlich weniger Stützstellen erreicht werden: viel der Zeit, die für die Berechnung der Funktion in diesen Gebieten verwendet wird, ist verschwendet. Wenn Sie solche Funktionen integrieren, können Sie Rechenzeit einsparen, wenn Sie die folgende Vorgehensweise befolgen:

1. Zerlegen Sie das Integrationsintervall in Teilintervalle, in denen die Funktion interessant ist, und solche, in denen die Funktion uninteressant ist.
2. Berechnen Sie das Integral über dem Intervall, in dem die Funktion interessant ist, indem Sie das Anzeigeformat so wählen, daß das Ergebnis die gewünschte Genauigkeit hat.
3. Berechnen Sie das Integral über dem Intervall, in dem die Funktion uninteressant ist, mit einer geringeren Genauigkeit, indem Sie im Anzeigeformat weniger Stellen spezifizieren.
4. Um das Integral des gesamten Integrationsintervalls zu berechnen, addieren Sie die Näherungen und ihre Fehlerabschätzungen von jedem Integrationsintervall. Dies kann sehr einfach mittels der Taste  geschehen.

Bevor Sie das Integrationsintervall unterteilen, überprüfen Sie, ob der Rechner einen Underflow erzeugt, wenn er die Funktion in dem Gebiet der unteren (oder oberen) Integrationsgrenze* auswertet. Da es keinen Grund dafür gibt, die Funktion für einen Wert von x zu berechnen, für den der Rechner einen Underflow erzeugt, kann in manchen Fällen die obere Integrationsgrenze gesenkt werden, um kostbare Rechenzeit zu sparen.

* Wenn das Ergebnis der Berechnung irgendeiner Größe eine Zahl kleiner als 10^{-99} ist, wird das Ergebnis durch Null ersetzt. Diese Bedingung wird als «Underflow» bezeichnet.

Erinnern Sie sich daran, daß Sie, sobald Sie ein Unterprogramm zur Berechnung von $f(x)$ eingegeben haben, $f(x)$ für jeden Wert von x auswerten können, indem Sie diesen Wert in das X-Register eingeben und **ENTER** **ENTER** **ENTER** **GSB**, gefolgt von dem Label des Unterprogramms, drücken.

Wenn der Rechner an der oberen Integrationsgrenze einen Underflow erzeugt, versuchen Sie es mit kleineren Werten für die obere Integrationsgrenze, bis Sie den Punkt erreicht haben, an dem der Rechner keinen Underflow mehr erzeugt.

Dies soll anhand des folgenden Integrals demonstriert werden:

$$\int_0^{\infty} x e^{-x} dx .$$

Geben Sie das Unterprogramm ein, das die Funktion $f(x) = x e^{-x}$ berechnet:

Tastenfolge	Anzeige	
g P/R		Programm-Modus.
f CLEAR PRGM	000–	Löscht den Programmspeicher.
f LBL 1	001–42,21, 1	
CHS	002– 16	
e^x	003– 12	
x	004– 20	
g RTN	005– 43 32	

Setzen Sie den Rechner in den Run-Modus und das Anzeigeformat auf **SCI** 3. Testen Sie dann verschiedene Werte von x , um herauszufinden, wo der Rechner einen Underflow erzeugt.

Tastenfolge	Anzeige	
g P/R		Run-Modus.
f SCI 3		Setzt das Format auf SCI 3.
EEX 3	1 03	Gibt 1000 in das X-Register.
ENTER ENTER ENTER	1.000 03	Füllt den Stack mit x .
GSB 1	0.000 00	Underflow bei $x = 1000$.
300 ENTER	3.000 02	Neuer Versuch mit einem kleineren Wert für x .
ENTER ENTER	3.000 00	
GSB 1	0.000 02	Es liegt immer noch ein Underflow vor.
200 ENTER	2.000 02	Neuer Versuch mit einem kleineren Wert für x .

Tastenfolge	Anzeige		
ENTER ENTER	2.000	02	
GSB 1	2.768	-85	Kein Underflow bei $x = 200$; verwenden Sie einen Wert zwi- schen 200 und 250.
225 ENTER	2.250	02	
ENTER ENTER	2.250	02	
GSB 1	4.324	-96	Ein «Fast»-Underflow.

An dieser Stelle können Sie **SOLVE** verwenden, um den kleinsten Wert von x zu finden, für den der Rechner einen Underflow erzeugt.

Tastenfolge	Anzeige		
R↓	2.250	02	Verschieben Sie den Stack nach unten, bis sich der zuletzt getestete Wert im X- und Y-Register befindet.
f SOLVE 1	2.280	02	Der kleinste Wert von x , mit dem der Rechner einen Under- flow erzeugt, ist ungefähr 228.

Sie haben jetzt also herausgefunden, daß Sie nur von 0 bis 228 integrieren müssen. Da der Integrand nur für Werte von x kleiner als zehn interessant ist, unterteilen Sie das Integrationsintervall am günstigsten an dieser Stelle. Das Problem hat sich nun auf folgende Form reduziert:

$$\int_0^{\infty} x e^{-x} dx \approx \int_0^{228} x e^{-x} dx = \int_0^{10} x e^{-x} dx + \int_{10}^{228} x e^{-x} dx.$$

Tastenfolge	Anzeige		
7 f DIM (i)	7.000	00	Teilt die Statistikregister ab.
f CLEAR Σ	0.000	00	Löscht die Statistikregister.
0 ENTER	0.000	00	Gibt die untere Integra- tionsgrenze des ersten Integra- tionsintervalls ein.
10	10		Gibt die obere Integra- tionsgrenze des ersten Integra- tionsintervalls ein.

Tastenfolge	Anzeige		
  1	9.995	-01	Integral über dem Intervall (0,10) mit dem Anzeigeformat  3 berechnet.
	1.000	00	Summiert die Näherung und ihre Fehlerabschätzung in Registern R ₃ und R ₅ auf.
	1.841	-04	Fehlerabschätzung der Näherung.
 	1.000	01	Schiebt den Stack nach unten, bis die obere Integrationsgrenze des ersten Integrationsintervalls im X-Register steht.
228	228		Gibt die obere Integrationsgrenze des zweiten Integrals in das X-Register ein. Die obere Integrationsgrenze des ersten Integrals wird in das Y-Register geschoben und wird zur unteren Integrationsgrenze des zweiten Integrals.
  0	2.	02	Spezifiziert das Anzeigeformat  0 für eine schnelle Berechnung des Integrals im Intervall (10,228). Wenn die Fehlerabschätzung der Näherung nicht klein genug erscheint, können Sie die Näherung mit einem Anzeigeformat wiederholen, das mehr signifikante Stellen liefert.
  1	5.	-04	 0 Integral über dem Intervall (10,228).
  3	5.328	-04	Ändert das Anzeigeformat wieder auf  3.
	7.568	-05	Prüft die Fehlerabschätzung der Näherung. Da sie kleiner als die des ersten Intervalls ist, führt  0 zu einer Näherung mit ausreichender Genauigkeit.

Tastenfolge	Anzeige		
$\boxed{x \approx y}$	5.328	-04	Gibt die Näherung und ihre Fehlerabschätzung in das X- bzw. Y-Register zurück, bevor diese Werte in den entsprechenden Statistikregistern aufsummiert werden.
$\boxed{\Sigma+}$	2.000	00	Summiert die Näherung und ihre Fehlerabschätzung.
$\boxed{\text{RCL}} \quad \boxed{\Sigma+}$	1.000	00	Integral über das ganze Intervall (0, 228) (zurückgerufen aus R ₃).
$\boxed{x \approx y}$	2.598	-04	Fehlerabschätzung des Integrals (aus R ₅).

Transformation von Variablen

In vielen Problemen, in denen die Funktion ihren Wert über einen großen Teil des Integrationsintervalls nur langsam ändert, kann eine geeignete Variablentransformation die Rechenzeit zur Berechnung des Integrals merklich verkürzen.

Lassen Sie uns als Beispiel nochmals das folgende Integral betrachten:

$$\int_0^{\infty} x e^{-x} dx.$$

Es sei $e^{-x} = u^3.$

Dann gilt $x = -3 \ln u$

und $dx = -3 \frac{du}{u}.$

Die Substitution ergibt

$$\begin{aligned} \int_0^{\infty} x e^{-x} dx &= \int_{e^{-0}}^{e^{-\infty}} (-3 \ln u)(u^3) \left(-3 \frac{du}{u} \right) \\ &= \int_1^0 9u^2 \ln u \, du. \end{aligned}$$

Geben Sie ein Unterprogramm ein, das die Funktion $f(u) = 9u^2 \ln u$ berechnet.

Tastenfolge   3  

9

 **Anzeige**

000–

001–42,21, 3

002– 43 12

003– 34

004– 43 11

005– 20

006– 9

007– 20

008– 43 32

Programm-Modus.

Geben Sie die Integrationsgrenzen ein und drücken Sie dann   3, um das Integral zu berechnen.

Tastenfolge 1 

0

  3**Anzeige**

1.000 00

0

1.000 00

3.020 –04

Run-Modus.

Eingabe der unteren Integrationsgrenze.

Eingabe der oberen Integrationsgrenze.

Näherung des Integralwertes.

Fehlerabschätzung der Näherung.

Die Näherung stimmt mit dem in dem vorigen Beispiel berechneten Wert überein.

Berechnung von schwierigen Integralen

Gewisse Bedingungen können die Rechenzeit verlängern oder ungenaue Ergebnisse erzeugen. Wie schon im *HP-15C Benutzerhandbuch* diskutiert wurde, hängen diese Bedingungen vom Verhalten des Integranden innerhalb des Integrationsintervalls ab.

Eine Klasse von sehr schwer zu berechnenden Integralen sind die uneigentlichen Integrale. Ein uneigentliches Integral ist ein Integral, das den Wert ∞ auf mindestens eine der folgenden Weisen enthält:

- Eine oder beide Integrationsgrenzen sind $\pm \infty$, wie z.B.

$$\int_{-\infty}^{\infty} e^{-u^2} du = \sqrt{\pi}.$$

- Der Integrand strebt gegen $\pm \infty$ irgendwo im Integrationsintervall, wie z.B.

$$\int_0^1 \ln(u) du = 1.$$

- Der Integrand oszilliert unendlich oft irgendwo innerhalb des Integrationsintervalls, wie z.B.

$$\int_0^1 \cos(\ln u) du = 1/2.$$

Etwa gleich unangenehm sind «beinahe» uneigentliche Integrale, die folgende Charakteristik haben:

- Der Integrand oder dessen erste Ableitung verändert sich stark innerhalb eines relativ schmalen Teils des Integrationsintervalls oder oszilliert häufig innerhalb dieses Streifens.

Der HP-15C versucht mit gewissen Integralen des zweiten Typs von uneigentlichen Integralen so zu verfahren, daß er den Integrand an den Integrationsgrenzen nicht auswertet.

Da uneigentliche und beinahe uneigentliche Integrale in der Praxis nicht selten vorkommen, sollten Sie wissen, wie Sie diese genau berechnen können. Die folgenden Beispiele verdeutlichen nützliche Methoden zur Berechnung solcher Integrale.

Lassen Sie uns den folgenden Integranden betrachten:

$$f(x) = \frac{\sqrt{-2 \ln \cos(x^2)}}{x^2}.$$

Diese Funktion verliert an Genauigkeit, wenn x kleine Werte annimmt. Dies kommt daher, daß $\cos(x^2)$ auf 1 gerundet wird, wodurch die Information über die Größe von x verlorengeht. Wenn Sie jedoch $u = \cos(x^2)$ substituieren, können Sie den Integranden berechnen:

$$f(x) = \begin{cases} 1 & \text{für } u = 1 \\ \frac{\sqrt{-2 \ln u}}{\cos^{-1} u} & \text{für } u \neq 1 \end{cases}$$

Obwohl die Programmverzweigung für $u = 1$ vier Schritte zu Ihrem Unterprogramm addiert, wird die Berechnung des Integrals dadurch sehr viel genauer.

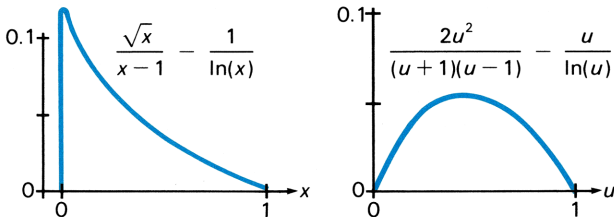
Lassen Sie uns als zweites Beispiel den folgenden Integranden betrachten:

$$\int_0^1 \left(\frac{\sqrt{x}}{x-1} - \frac{1}{\ln x} \right) dx.$$

Die Ableitung des Integranden nähert sich unendlich, wenn x gegen 0 strebt, wie es in der folgenden Illustration gezeigt wird. Durch die Substitution $x = u^2$ verhält sich die Funktion innerhalb des Integrationsintervalls gutartiger. Dieses Integral ist dann leicht zu berechnen:

$$\int_0^1 \left(\frac{2u^2}{(u+1)(u-1)} - \frac{u}{\ln u} \right) du.$$

Substituieren Sie nicht (u^2-1) für $(u+1)(u-1)$, da der zweite Teil des Ausdrucks durch den Rundungsfehler die Hälfte seiner signifikanten Ziffern verliert, wenn sich u 1 nähert und im Graph der Funktion einen zusätzlichen Ausschlag bei $u = 1$ erzeugt.



Als ein weiteres Beispiel wollen wir eine Funktion mit einer langen Flanke betrachten, die um ein Vielfaches größer ist als der restliche Graph (diesen Teil wollen wir «Körper» des Graphen nennen, hier liegen die interessanten Gebiete der Funktion). Eine Funktion wie

$$f(x) = e^{-x^2} \quad \text{oder} \quad g(x) = \frac{1}{x^2 + 10^{-10}}.$$

hat diese Eigenschaften.

Schmale Flanken, wie bei der Funktion $f(x)$, können abgeschnitten werden, ohne den Grad der Ungenauigkeit entscheidend zu vergrößern oder die Geschwindigkeit der Integration zu verlangsamen. $g(x)$ hat jedoch eine breite Flanke, die bei der Berechnung des Integrals

$$\int_{-t}^t g(x) dx$$

für große t nicht vernachlässigt werden kann.

Für solche Funktionen ist eine Substitution wie $x = a + b \tan u$ sehr nützlich, wobei a innerhalb des «Körpers» des Graphen liegt und b ungefähr die Breite der Flanke ist. Wenn Sie diese Substitution in $f(x)$ durchführen, ergibt sich für $a = 0$ und $b = 1$ folgendes Integral

$$\int_0^t f(x) dx = \int_0^{\tan^{-1}t} e^{-\tan^2 u} (1 + \tan^2 u) du,$$

das sehr rasch berechnet wird, auch wenn t in der Größenordnung von 10^{10} liegt. Die gleiche Substitution liefert auch bei $g(x)$ mit Werten von $a = 0$ und $b = 10^{-5}$ gute Ergebnisse.

Dieses Beispiel verwendet die Methode der Unterteilung des Integrationsintervalls. Obwohl die Funktion innerhalb des gesamten Integrationsintervalls Eigenschaften haben kann, die extrem erscheinen, kann die Funktion in Teilen des Integrationsintervalls einen gutartigen Verlauf haben. Die Methode der Unterteilung des Integrationsintervalls ist am erfolgreichsten, wenn man sie mit einer geeigneten Substitution verbindet. Betrachten Sie das folgende Integral:

$$\begin{aligned} \int_0^\infty dx/(1+x^{64}) &= \int_0^1 dx/(1+x^{64}) + \int_1^\infty dx/(1+x^{64}) \\ &= \int_0^1 dx/(1+x^{64}) + \int_0^1 u^{62} du/(u^{64}+1) \\ &= \int_0^1 (1+x^{62}) dx/(1+x^{64}) \\ &= 1 + \int_0^1 (x^{62} - x^{64}) dx/(1+x^{64}) \\ &= 1 + \frac{1}{8} \int_0^1 (1-v^{1/4}) v^{55/8} dv/(1+v^8). \end{aligned}$$

Diese Schritte verwenden die Substitutionen $x = 1/u$ und $x = v^{1/8}$ und einige algebraische Manipulationen. Obwohl das ursprüngliche Integral uneigentlich

ist, kann das letzte Integral leicht mit $\int$ berechnet werden. Durch das Ausklammern des konstanten Terms aus dem Integral erhält man (mit dem Anzeigeformat SCI 8) ein Ergebnis mit 13 signifikanten Stellen:

$$1.000401708155 \pm 1.2 \times 10^{12}.$$

Das letzte Beispiel ist aus dem wirklichen Leben genommen und hängt mit dem elektrostatischen Feld einer ellipsoiden Probeladung mit den Halbachsen a , b und c zusammen:

$$V = \int_0^\infty \frac{dx}{(a^2 + x)\sqrt{(a^2 + x)(b^2 + x)(c^2 + x)}}$$

für $a = 100$, $b = 2$ und $c = 1$.*

Transformieren Sie dieses uneigentliche Integral in ein eigentliches mit der Substitution $x = (a^2 - c^2)/(1 - u^2) - a^2$:

$$V = p \int_r^1 \sqrt{(1 - u^2)/(u^2 + q)} du$$

wobei

$$p = 2/((a^2 - c^2)\sqrt{a^2 - b^2}) = 2.00060018 \times 10^{-6}$$

$$q = (b^2 - c^2)/(a^2 - b^2) = 3.001200480 \times 10^{-3}$$

$$r = c/a = 0.01.$$

Dieses Integral ist jedoch beinahe uneigentlich, da q und r beide fast Null sind. Indem Sie aber ein Integral in geschlossener Form, das ausreichend dem problematischen Teil von V ähnelt, verwenden, kann diese Schwierigkeit vermieden werden. Versuchen Sie

$$W = p \int_r^1 du / \sqrt{u^2 + q} = p \ln(u + \sqrt{u^2 + q}) \Big|_r^1$$

$$= p \ln((1 + \sqrt{1 + q})/(r + \sqrt{r^2 + q}))$$

$$= 8.40181880708 \times 10^{-6}.$$

Und dann

$$\begin{aligned} V &= W + p \int_r^1 (\sqrt{(1 - u^2)/(u^2 + q)} - 1/\sqrt{u^2 + q}) du \\ &= p \int_r^1 \left(\frac{W/p}{1 - r} - \frac{u^2}{(1 + \sqrt{1 - u^2})\sqrt{u^2 + q}} \right) du. \end{aligned}$$

* Siehe Stratton, J.A., *Electromagnetic Theory*, McGraw-Hill, New York, 1941, Seite 201–217.

Der HP-15C berechnet dieses Integral ohne Schwierigkeit. Sie brauchen sich keine Gedanken über den Term $\sqrt{1-u^2}$ machen, wenn u gegen 1 strebt, da die Ziffern, die durch Rundung verloren gehen, nicht gebraucht werden.

Anwendungen

Das folgende Programm berechnet den Wert von vier speziellen Funktionen für das Argument x :

$$P(x) = \frac{1}{2\pi} \int_{-\infty}^x e^{-t^2/2} dt \quad (\text{Normalverteilungsfunktion})$$

$$Q(x) = 1 - P(x) = \frac{1}{2\pi} \int_x^{\infty} e^{-t^2/2} dt \quad (\text{Komplementäre Normalverteilungsfunktion})$$

$$\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt \quad (\text{Fehlerfunktion})$$

$$\text{erfc}(x) = 1 - \text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_x^{\infty} e^{-t^2} dt \quad (\text{Komplementäre Fehlerfunktion})$$

Das Programm, das diese Funktionen berechnet, verwendet die Transformation $u = e^{-t^2}$, wobei $|x| > 1.6$ ist.

Der Funktionswert wird im X-Register abgelegt und die Fehlerabschätzung des *Integrals* im Y-Register. (Die Fehlerabschätzung des *Funktionswertes* liegt ungefähr in der Größenordnung der Zahl im Y-Register.) Das ursprüngliche Argument ist im Register R_0 abgespeichert.

Das Programm hat die folgenden Charakteristika:

- Das Anzeigeformat spezifiziert die Genauigkeit des Integranden in der gleichen Weise wie für  selbst. Wenn Sie jedoch eine unnötig große Zahl von Stellen spezifizieren, erhöht sich die Rechenzeit.
- Kleine Funktionswerte, wie $Q(20)$, $P(-20)$ und $\text{erfc}(10)$, werden exakt und schnell berechnet.

Tastenfolge

[g] [P/R]
[f] [CLEAR] [PRGM]
[f] [LBL] [A]
[STO] 2
[CHS]
[GTO] 2

[f] [LBL] [B]
[STO] 2
[f] [LBL] 2
2
[√x]
[÷]
[GSB] [C]
2
[÷]
[RCL] 2
[STO] 0
[R↓]
[g] [RTN]

[f] [LBL] [C]
1
[GSB] 4
[g] [F?] 1

[GTO] 5

1
[−]

[f] [LBL] 5
[CHS]
[g] [RTN]

[f] [LBL] [E]
0
[f] [LBL] 4

[g] [CF] 1

Anzeige

000–
001–42,21,11
002– 44 2
003– 16
004– 22 2

005–42,21,12
006– 44 2
007–42,21, 2
008– 2
009– 11
010– 10
011– 32 13
012– 2
013– 10
014– 45 2
015– 44 0
016– 33
017– 43 32

018–42,21,13
019– 1
020– 32 4
021–43, 6, 1

022– 22 5

023– 1
024– 30

025–42,21, 5
026– 16
027– 43 32

028–42,21,15
029– 0
030–42,21, 4

031–43, 5, 1

Programm-Modus.

Programm für $P(x)$.

Speichert x in R_2 .

Berechnet $-x$.

Programmverzweigung, um $P(x) = Q(-x)$ zu berechnen.

Programm für $Q(x)$.

Speichert x in R_2 .

Berechnet $\operatorname{erfc}(x/\sqrt{2})$.

Berechnet $Q(x) = \frac{1}{2} \operatorname{erfc}(x/\sqrt{2})$.

Speichert x in R_0 .

Gibt den Funktionswert zurück.

Programm für $\operatorname{erfc}(x)$.

Überprüft den Zustand von Flag 1.

Programmverzweigung, falls Flag 1 gesetzt ist.

Berechnet $\operatorname{erf}(x) - 1$, wenn Flag 1 gelöscht ist.

Berechnet $\operatorname{erfc}(x)$.

Gibt den Funktionswert zurück.

Programm für $\operatorname{erf}(x)$.

Unterprogramm für $\operatorname{erf}(x)$ oder $\operatorname{erfc}(x)$.

Löscht Flag 1.

Tastenfolge

Anzeige

STO 1

032– 44 1

Speichert 0 für $\text{erf}(x)$, 1 für $\text{erfc}(x)$.

033– 34

STO 0

034– 44 0

 **ABS**

035– 43 16

Berechnet $|x|$.

1

036– 1

.

037– 48

6

038– 6

 **TEST** 8

039–43,30, 8

Prüft, ob $|x| > 1.6$ ist.**GTO** 6

040– 22 6

Programmverzweigung für $|x| > 1.6$.

0

041– 0

RCL 0

042– 45 0

Ruft x zurück.  0

043–42,20, 0

Integriert e^{-t^2} von 0 bis x .

2

044– 2



045– 20

 **LBL** 3

046–42,21, 3

Unterprogramm zur Division durch $\sqrt{\pi}$. 

047– 43 26



048– 11



049– 10

 **RTN**

050– 43 32

 **LBL** 6

051–42,21, 6

Unterprogramm zur Integration, wenn $|x| > 1.6$ ist.
Setzt Flag 1. **SF** 1

052–43, 4, 1

0

053– 0

RCL 0

054– 45 0

055– 43 11

CHS

056– 16



057– 12

Berechnet e^{-x^2} .  1

058–42,20, 1

Integriert $(-\ln u)^{-1/2}$ von 0 bis e^{-x^2} .**GSB** 3

059– 32 3

Dividiert das Integral durch $\sqrt{\pi}$.**RCL** 0

060– 45 0

ENTER

061– 36

 **ABS**

062– 43 16



063– 10

Berechnet das Vorzeichen von x .

064– 20

Tastenfolge**Anzeige****RCL** 1

065– 45 1

Ruft 1 für $\operatorname{erfc}(x)$ und 0 für $\operatorname{erf}(x)$ zurück.**g** **LSTx**

066– 43 36

–

067– 30

+

068– 40

Paßt das Integral an das Vorzeichen von x und der Funktion an.**CHS**

069– 16

g **RTN**

070– 43 32

f **LBL** 0

071–42,21, 0

Unterprogramm zur Berechnung von e^{-t^2} .**g** **x²**

072– 43 11

CHS

073– 16

e^x

074– 12

g **RTN**

075– 43 32

f **LBL** 1

076–42,21, 1

Unterprogramm zur Berechnung von $(-\ln u)^{-1/2}$.**g** **x=0**

077– 43 20

g **RTN**

078– 43 32

g **LN**

079– 43 12

CHS

080– 16

√x

081– 11

1/x

082– 15

g **RTN**

083– 43 32

Verwendete Labels: A, B, C, E, 0, 1, 2, 3, 4, 5 und 6.

Verwendete Register: $R_0(x)$, R_1 , R_2 .

Verwendeter Flag: 1.

Das Programm ist wie folgt zu benutzen:

1. Geben Sie das Argument x in die Anzeige ein.
2. Berechnen Sie die gewünschte Funktion:
 - Drücken Sie **f** **A**, um $P(x)$ zu ermitteln.
 - Drücken Sie **f** **B**, um $Q(x)$ zu ermitteln.
 - Drücken Sie **f** **E**, um $\operatorname{erf}(x)$ zu ermitteln.
 - Drücken Sie **f** **C**, um $\operatorname{erfc}(x)$ zu ermitteln.

Beispiel: Berechnen Sie $Q(20)$, $P(1.234)$ und $\text{erf}(0.5)$ mit Anzeigeformat $\left[\text{SCI}\right]$ 3.

Tastenfolge	Anzeige	
$\left[\text{g}\right] \left[\text{P/R}\right]$		Run-Modus.
$\left[\text{f}\right] \left[\text{SCI}\right] 3$		Spezifiziert das Format.
$20 \left[\text{f}\right] \left[\text{B}\right]$	2.754 -89	$Q(20)$.
$1.234 \left[\text{f}\right] \left[\text{A}\right]$	8.914 -01	$P(1.234)$.
$.5 \left[\text{f}\right] \left[\text{E}\right]$	5.205 -01	$\text{erf}(0.5)$.

Beispiel: Berechnen Sie für eine normalverteilte Zufallsvariable X mit einem Mittelwert von 2.151 und einer Standardabweichung von 1.085 die Wahrscheinlichkeit $\text{Pr}[2 < X \leq 3]$.

$$\begin{aligned} \text{Pr}[2 < X \leq 3] &= \text{Pr}\left[\frac{2 - 2.151}{1.085} < \frac{X - \mu}{\sigma} \leq \frac{3 - 2.151}{1.085}\right] \\ &= \text{P}\left(\frac{3 - 2.151}{1.085}\right) - \text{P}\left(\frac{2 - 2.151}{1.085}\right) \end{aligned}$$

Tastenfolge	Anzeige	
$2 \left[\text{ENTER}\right]$	2.000 00	
$2.151 \left[-\right]$	-1.510 -01	
$1.085 \left[\div\right]$	-1.392 -01	
$\left[\text{f}\right] \left[\text{A}\right]$	4.447 -01	Berechnet $\text{Pr}[X \leq 2]$.
$\left[\text{STO}\right] 3$	4.447 -01	Speichert den Wert.
$3 \left[\text{ENTER}\right]$	3.000 00	
$2.151 \left[-\right]$	8.490 -01	
$1.085 \left[\div\right]$	7.825 -01	
$\left[\text{f}\right] \left[\text{A}\right]$	7.830 -01	Berechnet $\text{Pr}[X \leq 3]$.
$\left[\text{RCL}\right] 3$	4.447 -01	Ruft $\text{Pr}[X \leq 2]$ zurück.
$\left[-\right]$	3.384 -01	Berechnet $\text{Pr}[2 < X \leq 3]$.
$\left[\text{f}\right] \left[\text{FIX}\right] 4$	0.3384	

Rechnen im Komplex-Modus

Wichtige physikalische Aufgaben mit reellen Daten können oft mit relativ einfachen Rechnungen gelöst werden, wenn man komplexe Zahlen verwendet. Dieser Abschnitt vermittelt einen wichtigen Einblick in das Rechnen mit komplexen Zahlen und zeigt an mehreren Beispielen die Lösung von Aufgabenstellungen mit komplexen Zahlen.

Gebrauch des Komplex-Modus

Im Komplex-Modus können Sie mit dem HP-15C komplexwertige Ausdrücke einfach berechnen. Im allgemeinen wird ein mathematischer Ausdruck im Komplex-Modus genauso wie im normalen, «reellen» Modus eingegeben. Betrachten wir beispielsweise ein Programm, das das Polynom $P(x) = a_n x^n + \dots + a_1 x + a_0$ für den Wert x im X-Register berechnet. Im Komplex-Modus kann dasselbe Programm $P(z)$, wo z eine komplexe Zahl ist, berechnen. Auf ähnliche Weise können im Komplex-Modus andere Ausdrücke, wie die Gamma-Funktion $\Gamma(x)$ im nächsten Beispiel, für komplexe Argumente berechnet werden.

Beispiel: Schreiben Sie ein Programm, das die folgende nicht abbrechende Näherung

$$\ln(\Gamma(x)) = (x - 1/2)\ln x - x + a_0 + \frac{a_1}{x + \frac{a_2}{x + \frac{a_3}{x + \dots}}}$$

für die ersten sechs Werte von a berechnet:

$$\begin{aligned} a_0 &= 1/2 \ln(2\pi) \\ a_1 &= 1/12 \\ a_2 &= 1/30 \\ a_3 &= 53/210 \\ a_4 &= 195/371 \\ a_5 &= 1.011523068 \\ a_6 &= 1.517473649. \end{aligned}$$

Da diese Näherung sowohl für reelle als auch für komplexe Argumente mit $\operatorname{Re}(z) > 0$ gültig ist, kann dieses Programm im Komplex-Modus $\ln(\Gamma(z))$ (für hinreichend große $|z|$) berechnen. Wenn $|z| > 4$ (und $\operatorname{Re}(z) > 0$) ist, ist die Näherung auf 9 oder 10 Stellen genau.

Geben Sie das folgende Programm ein:

Tastenfolge	Anzeige	Programm-Modus.
g P/R		Programm-Modus.
f CLEAR PRGM	000—	
f LBL A	001—42,21,11	
6	002— 6	
STO I	003— 44 25	Speichert den Zähler im Index-register.
x$\geq$y	004— 34	
ENTER	005— 36	
ENTER	006— 36	
ENTER	007— 36	Restauriert z.
RCL 6	008— 45 6	Ruft a_6 zurück.
f LBL 1	009—42,21, 1	Schleife, um den Bruch fortzusetzen.
+	010— 40	
RCL (i)	011— 45 24	Ruft a_i zurück.
x$\geq$y	012— 34	Restauriert z.
÷	013— 10	
f DSE I	014—42, 5,25	Erniedrigt den Zähler.
GTO 1	015— 22 1	
RCL 0	016— 45 0	Ruft a_0 zurück.
+	017— 40	
x$\geq$y	018— 34	Restauriert z.
-	019— 30	
g LSTx	020— 43 36	Ruft z wieder zurück.
g LN	021— 43 12	Berechnet $\ln(z)$.
g LSTx	022— 43 36	Ruft z wieder zurück.
-	023— 48	
5	024— 5	
-	025— 30	Berechnet $z - \frac{1}{2}$.

Tastenfolge	Anzeige	
[x]	026– 20	
[+]	027– 40	Berechnet $\ln(\Gamma(z))$.
[g] [RTN]	028– 43 32	

Speichern Sie die Konstanten in der Reihenfolge ihrer Indices in den Speichern R_0 bis R_6 ab.

Tastenfolge	Anzeige	
[g] [P/R]		Run-Modus.
2 [g] π [x]	6.2832	
[g] [LN] 2 $\div$	0.9189	
[STO] 0	0.9189	Speichert a_0 .
12 [1/x] [STO] 1	0.0833	Speichert a_1 .
30 [1/x] [STO] 2	0.0333	Speichert a_2 .
53 [ENTER] 210 $\div$	0.2524	
[STO] 3	0.2524	Speichert a_3 .
195 [ENTER] 371 $\div$	0.5256	
[STO] 4	0.5256	Speichert a_4 .
1.011523068 [STO] 5	1.0115	Speichert a_5 .
1.517473649 [STO] 6	1.5175	Speichert a_6 .

Berechnen Sie mit diesem Programm $\ln(\Gamma(4.2))$, und vergleichen Sie das Ergebnis mit $\ln(3.2!)$, das mit der **[x!]** Funktion berechnet wurde. Berechnen Sie auch $\ln(\Gamma(1 + 5i))$.

Tastenfolge	Anzeige	
4.2 [f] [A]	2.0486	Berechnet $\ln(\Gamma(4.2))$.
[f] [FIX] 9	2.048555637	Zeigt 10 Stellen an.
3.2 [f] [x!]	7.756689536	Berechnet $(3.2)! = \Gamma(3.2 + 1)$.
[g] [LN]	2.048555637	Berechnet $\ln(3.2!)$.
1 [ENTER]	1.000000000	Gibt den Realteil von $1 + 5i$ ein.
5 [f] [I]	1.000000000	Bildet die komplexe Zahl $1 + 5i$.

Tastenfolgef **A**f **Re $\Im$ Im**f **FIX** 4**Anzeige**

-6.130324145

3.815898575

3.8159

Realteil von $\ln(\Gamma(1 + 5i))$.Imaginärteil von $\ln(\Gamma(1 + 5i))$.

Der einzige Mehraufwand bei der Berechnung des komplexen Ergebnisses besteht darin, den Imaginärteil des Arguments z einzugeben. Das Ergebnis $\ln(\Gamma(1 + 5i))$ hat jeweils 10 korrekte Stellen im reellen und imaginären Anteil.

Trigonometrische Modi

Obwohl die Statusanzeige für den aktuellen trigonometrischen Modus im Komplex-Modus weiter aufleuchtet, werden komplexe Funktionen *immer* im *Bogenmaß* berechnet. Die Statusanzeige deutet den Modus (Grad, Bogenmaß oder Neugrad) nur für die zwei komplexen Umrechnungen **↔P** und **↔R** an.

Wenn Sie $re^{i\theta}$ berechnen wollen, wo θ in Grad gegeben ist, kann **e^x** nicht direkt benutzt werden, da θ in Bogenmaß vorliegen muß. Wenn Sie von Grad auf Bogenmaß umrechnen, verlieren Sie etwas an Genauigkeit, besonders bei Werten wie 180° , für die der Wert π im Bogenmaß mit 10 Stellen nicht genau dargestellt werden kann.

Im Komplex-Modus jedoch berechnet die Funktion **↔R** $re^{i\theta}$ exakt, wobei θ in *jedem* Maß vorliegen darf (das von der Statusanzeige angedeutet wird). Geben Sie einfach r und θ in die komplexen X-Register in der Form $r + i\theta$ ein und führen dann **↔R** aus, um den komplexen Wert

$$re^{i\theta} = r \cos \theta + ir \sin \theta$$

zu berechnen. (Das Programm, das am Ende dieses Abschnitts bei Berechnung der n -ten Wurzel einer komplexen Zahl angegeben ist, benutzt diese Funktion.)

Definitionen mathematischer Funktionen

Es folgt eine Liste der Definitionen der Operationen des HP-15C im Komplex-Modus. In diesen wird eine komplexe Zahl ausgedrückt durch $z = x + iy$ (kartesische Form) oder $z = re^{i\theta}$ (Polar-Form). Es gilt $|z| = \sqrt{x^2 + y^2}$.

Arithmetische Operationen

$$(a + ib) \pm (c + id) = (a \pm c) + i(b \pm d)$$

$$(a + ib)(c + id) = (ac - bd) + i(ad + bc)$$

$$z^2 = z \times z$$

$$1/z = x/|z|^2 - iy/|z|^2$$

$$z_1 \div z_2 = z_1 \times 1/z_2$$

Einwertige Funktionen

$$e^z = e^x(\cos y + i \sin y)$$

$$10^z = e^{z \ln 10}$$

$$\sin z = \frac{1}{2i}(e^{iz} - e^{-iz})$$

$$\cos z = \frac{1}{2}(e^{iz} + e^{-iz})$$

$$\tan z = \sin z / \cos z$$

$$\sinh z = \frac{1}{2}(e^z - e^{-z})$$

$$\cosh z = \frac{1}{2}(e^z + e^{-z})$$

$$\tanh z = \sinh z / \cosh z$$

Mehrwertige Funktionen

Im allgemeinen kann die Umkehrfunktion $f^{-1}(z)$ einer Funktion $f(z)$ mehr als einen Wert für ein bestimmtes Argument z annehmen. Zum Beispiel hat $\cos^{-1}(z)$ für jedes Argument z unendlich viele Funktionswerte. Der HP-15C berechnet den eindeutigen *Hauptwert*, der in der Teilmenge des Wertebereichs liegt, die als *Hauptzweig* von f^{-1} bezeichnet wird. In der folgenden Diskussion wird die eindeutige Umkehrfunktion (die auf den Hauptzweig beschränkt ist) mit Großbuchstaben bezeichnet – wie z.B. $\text{COS}^{-1}(z)$ – um sie von der vieldeutigen Umkehrfunktion $\cos^{-1}(z)$ zu unterscheiden.

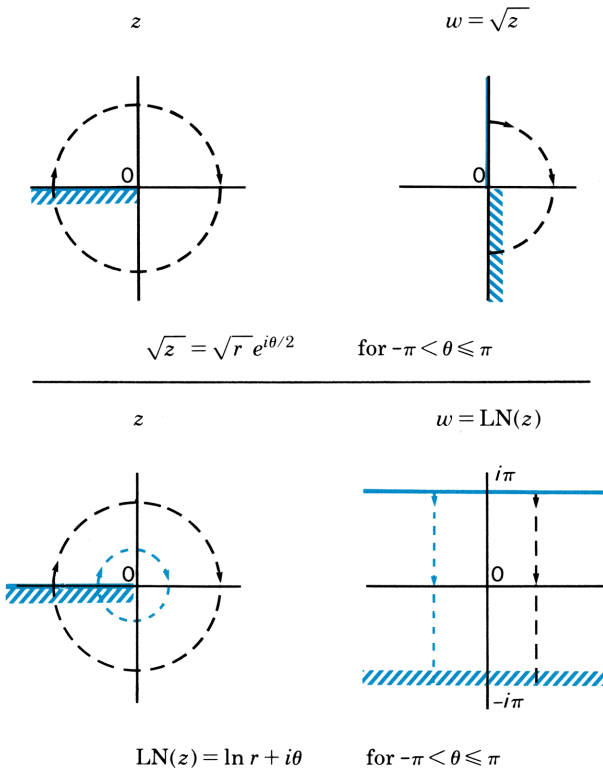
Betrachten wir beispielsweise die n -ten Wurzeln einer komplexen Zahl z . Schreiben Sie z in der Polarform als $z = r e^{i(\theta + 2k\pi)}$ für $-\pi \leq \theta$ mit $k = 0, \pm 1, \pm 2, \dots$. Wenn n eine positive ganze Zahl ist, dann gilt:

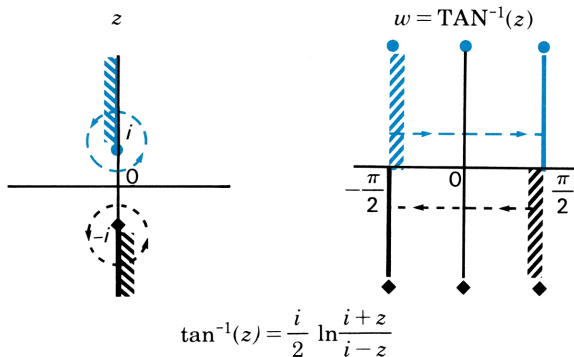
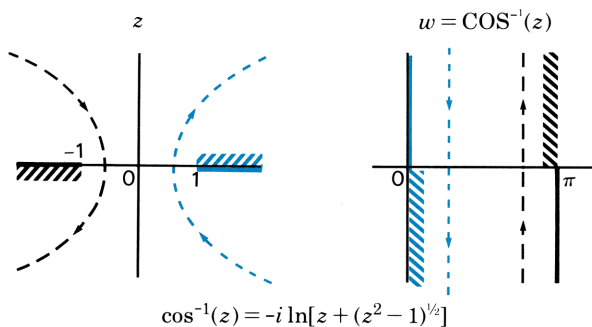
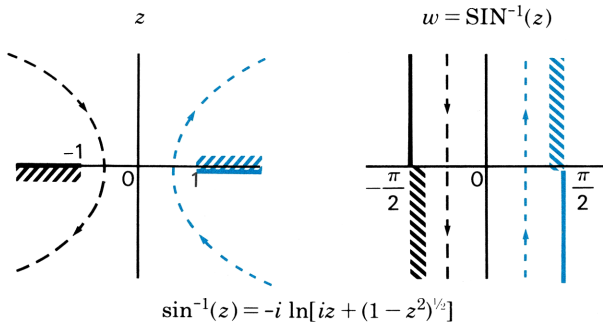
$$z^{1/n} = r^{1/n} e^{i(\theta/n + 2k\pi/n)} = r^{1/n} e^{i\theta/n} e^{i2k\pi/n}.$$

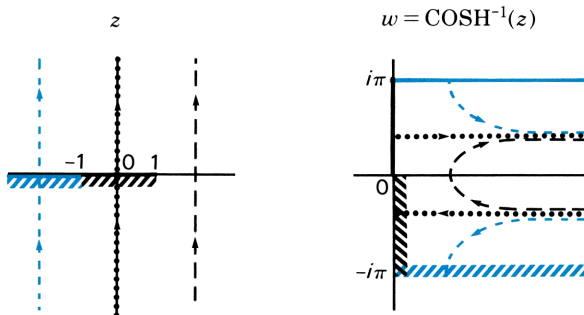
Nur $k = 0, 1, \dots, n-1$ sind nötig, da sich $e^{i2k\pi/n}$ mit einer Periode von n wiederholt. Diese Gleichung definiert die n -ten Wurzeln von z , und $r^{1/n} e^{i\theta/n}$ mit $-\pi < \theta \leq \pi$ ist der Hauptzweig von $z^{1/n}$. (Das Programm auf Seite 78 berechnet die n -ten Wurzeln von z .)

Die folgenden Abbildungen zeigen die Hauptzweige der Umkehrrelationen. Der linke Graph in der jeweiligen Abbildung zeigt den eingeschränkten Definitionsbereich der Umkehrfunktion, der rechte Graph zeigt den Wertebereich des Hauptzweiges.

Die Definitionen der Hauptzweige einiger Umkehrrelationen sind nicht universell vereinbart. Die Hauptzweige, mit denen der HP-15C rechnet, wurden sorgfältig gewählt. Erstens sind sie in den Bereichen, in denen die Argumente der reellwertigen Umkehrfunktionen definiert sind, analytisch. Das heißt, der Verzweigungsschnitt liegt dort, wo die entsprechende reellwertige Umkehrfunktion nicht definiert ist. Zweitens werden die meisten wichtigen Symmetrien erhalten. Beispielsweise gilt für alle z : $\text{SIN}^{-1}(-z) = -\text{SIN}^{-1}(z)$.







$$\cosh^{-1}(z) = \ln[z + (z^2 - 1)^{1/2}]$$

In den letzten vier Abbildungen wurden die dargestellten Hauptzweige aus den angegebenen Gleichungen ermittelt, sie verwenden jedoch nicht notwendigerweise die Hauptzweige von $\ln(z)$ und von $\sqrt{z}$.

Die weiteren Umkehrfunktionen können aus den obigen Illustrationen und den folgenden Gleichungen bestimmt werden.

$$\text{LOG}(z) = \text{LN}(z) / \text{LN}(10)$$

$$\text{SINH}^{-1}(z) = -i \text{SIN}^{-1}(iz)$$

$$\text{TANH}^{-1}(z) = -i \text{TAN}^{-1}(iz)$$

$$w^z = e^{z \text{LN}(w)}$$

Um *alle* Werte einer Umkehrrelation aus dem Hauptwert, den der HP-15C berechnet, zu bestimmen, können Sie die folgenden Ausdrücke benutzen, in denen $k = 0, \pm 1, \pm 2, \dots$ gilt:

$$z^{1/2} = \pm \sqrt{z}$$

$$\ln(z) = \text{LN}(z) + i2k\pi$$

$$\sin^{-1}(z) = (-1)^k \text{SIN}^{-1}(z) + k\pi$$

$$\cos^{-1}(z) = \pm \text{COS}^{-1}(z) + 2k\pi$$

$$\tan^{-1}(z) = \text{TAN}^{-1}(z) + k\pi$$

$$\sinh^{-1}(z) = (-1)^k \text{SINH}^{-1}(z) + ik\pi$$

$$\cosh^{-1}(z) = \pm \text{COSH}^{-1}(z) + i2k\pi$$

$$\tanh^{-1}(z) = \text{TANH}^{-1}(z) + ik\pi$$

$$w^z = w^z e^{i2\pi kz}$$

Verwendung von **SOLVE** und $\int$ im Komplex-Modus

Die Funktionen **SOLVE** und $\int$ verwenden Algorithmen, die die Werte, die die vorgegebene Funktion entlang der reellen x -Achse annimmt, untersuchen. Auch im Komplex-Modus arbeiten die Funktionen **SOLVE** und $\int$ nur mit dem reellen Stack, selbst wenn Ihr Funktionsunterprogramm komplexe Berechnungen enthält.

SOLVE wird beispielsweise nicht die Nullstellen einer komplexwertigen Funktion suchen, sondern die Funktion auf der reellen Achse auswerten und eine Nullstelle des Realteils der Funktion suchen. Ebenso berechnet $\int$ das Integral des Realteils der Funktion in einem Intervall auf der reellen Achse. Diese Operationen sind in zahlreichen Anwendungen nützlich, wie etwa bei der Berechnung von Konturintegralen und komplexen Potentialen. (Siehe die Anwendungen am Ende dieses Abschnittes.)

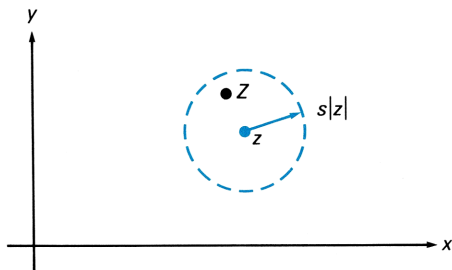
Genauigkeit im Komplex-Modus

Komplexe Zahlen haben sowohl einen Real- als auch einen Imaginärteil, und daher kommt der Genauigkeit bei komplexen Berechnungen eine höhere Bedeutung als bei reellen Berechnungen zu.

Wenn man mit *reellen* Zahlen rechnet, ist x eine gute Näherung für X , wenn die relative Differenz $E(X,x) = |(X-x)/x|$ klein ist. Diese hängt direkt mit der Anzahl von korrekten oder signifikanten Ziffern der Näherung X zusammen. Wenn $E(X,x) < 5 \times 10^{-n}$ ist, dann hat X mindestens n signifikante Ziffern. Definieren Sie für *komplexe* Zahlen $E(Z,z) = |(Z-z)/z|$. Diese Größe hängt jedoch *nicht* direkt mit der Anzahl korrekter Ziffern in jeder *Komponente* von Z zusammen.

Wenn beispielsweise $E(X,x)$ und $E(Y,y)$ beide klein sind, dann muß $E(Z,z)$ für $z = x + iy$ auch klein sein. In anderen Worten, wenn $E(X,x) < s$ und $E(Y,y) < s$ ist, dann ist $E(Z,z) < s$. Betrachten wir jedoch $z = 10^{10} + i$ und $Z = 10^{10}$. Der Imaginärteil von Z ist völlig ungenau, und doch ist $E(Z,z) < 10^{-10}$. Obwohl also die Imaginärteile von z und Z vollkommen verschieden sind, liegen in einem relativen Sinn z und Z außerordentlich dicht beieinander.

Es gibt eine einfache, geometrische Interpretation des komplexen relativen Fehlers. Bei jeder Näherung Z für z gilt $E(Z,z) < s$ (wo s eine positive reelle Zahl ist) genau dann, wenn Z innerhalb eines Kreises mit Radius $s|z|$ und Mittelpunkt z in der komplexen Ebene liegt.



Wenn man Näherungen verlangt, in denen die *Komponenten* beide genau sind, fordert man mehr, als daß die relativen Fehler klein sein sollen. Betrachten wir das folgende Beispiel, wo die Rechnungen mit vierstelliger Genauigkeit durchgeführt sind. Es illustriert die Grenzen, die eine endliche Rechengenauigkeit einer Rechnung mit komplexen Zahlen auferlegt.

$$z_1 = Z_1 = 37.1 + 37.3i$$

$$z_2 = Z_2 = 37.5 + 37.3i$$

und

$$\begin{aligned} Z_1 \times Z_2 &= (37.10 \times 37.50 - 37.30 \times 37.30) + i(37.10 \times 37.30 + 37.30 \times 37.50) \\ &= (1391. - 1391.) + i(1384. + 1399.) \\ &= 0 + i(2783.) \end{aligned}$$

Das richtige Ergebnis ist $z_1 z_2 = 0.04 + 2782.58i$. Obwohl Z_1 und Z_2 keinen Fehler haben, hat der Realteil Ihres vierstelligen Produktes keine korrekten signifikanten Stellen, und doch ist der relative Fehler des komplexen Produktes kleiner als 2×10^{-4} .

Dieses Beispiel illustriert, daß sich der Fehler bei der komplexen Multiplikation nicht komponentenweise fortpflanzt. Aber selbst wenn die komplexe Multiplikation genaue Komponenten ergeben würde, würden die Rundungsfehler bei einer Kettenrechnung schnell ungenaue Komponenten liefern. Andererseits wächst der relative Fehler, der ein Maß für die Genauigkeit der Rechnung ist, nur langsam an.

Wenn wir beispielsweise wie vorher vier Stellen Genauigkeit verwenden,

$$z_1 = (1 + 1/300) + i$$

$$Z_1 = 1.003 + i$$

$$z_2 = Z_2 = 1 + i$$

dann ist

$$\begin{aligned} Z_1 \times Z_2 &= (1.003 + i) \times (1 + i) \\ &= 0.003 + 2.003i \\ &= 3.000 \times 10^{-3} + 2.003i \end{aligned}$$

Der *korrekte* vierstellige Wert ist $3.333 \times 10^{-3} + 2.003i$. In diesem Beispiel sind der Real- und der Imaginärteil von Z_1 und Z_2 genau, und die Arithmetik ist exakt. Ihr «Produkt» ist jedoch nicht genau, d.h. der Realteil hat nur eine signifikante Stelle. Ein Rundungsfehler erzeugt eine ungenaue Komponente, obwohl der komplexe relative Fehler des Produkts weiterhin klein bleibt.

Der HP-15C ist so konstruiert, daß das Ergebnis bei allen komplexen Operationen in dem Sinn genau ist, daß der komplexe relative Fehler $E(Z, z)$ klein gehalten wird. Im allgemeinen gilt $E(Z, z) < 6 \times 10^{-10}$.

Wie wir oben gezeigt haben, garantiert dieser kleine relative Fehler nicht in beiden Komponenten 10 genaue Ziffern. Der Fehler wird bestimmt durch den Wert von $|z|$, und da dieser nicht sehr verschieden ist vom Wert der größeren Komponente von z , kann die kleinere Komponente weniger genaue Stellen haben. Es gibt eine einfache Methode, mit der Sie feststellen können, welche Stellen im allgemeinen noch genau sind. Drücken Sie dazu jede Komponente mit dem größtmöglichen Exponenten aus. In dieser Form sind ungefähr die ersten 10 Stellen jeder Komponente genau. Wenn zum Beispiel

$$Z = 1.234567890 \times 10^{-10} + i(2.222222222 \times 10^{-3})$$

ist, dann denken Sie sich Z als

$$0.0000001234567890 \times 10^{-3} + i(2.222222222 \times 10^{-3}).$$

Die genauen Ziffern sind

$$0.000000123 \times 10^{-3} + i(2.222222222 \times 10^{-3}).$$

Anwendungen

Die komplexen Operationen des HP-15C ermöglichen Ihnen das Bearbeiten von Aufgabenstellungen, die mit reellen Zahlen allein nicht gelöst werden können. Auf den folgenden Seiten sind mehrere Programmbeispiele aufgeführt, die die Nützlichkeit von komplexen Rechnungen – und damit des HP-15C – zeigen.

Speichern und Zurückrufen von komplexen Zahlen mittels einer Matrix

Dieses Programm verwendet den Stack und Matrix C, um komplexe Zahlen zu speichern und zurückzurufen. Es hat die folgenden Eigenschaften:

- Wenn Sie einen Index angeben, der größer ist als die Dimension der Matrix, zeigt Ihr Rechner **Error 3** an, und der Stack ist für eine erneute Eingabe bereit.
- Wenn Ihr Rechner nicht im Komplex-Modus ist, aktiviert das Programm den Komplex-Modus, und der Imaginärteil der Zahl wird Null gesetzt.
- Wenn Sie eine komplexe Zahl speichern, geht der Index verloren, und das T-Register wird in das Z-Register dupliziert.
- Das Programm «Store» verwendet die Taste **[D]**, die sich über der Taste **[STO]** befindet. Das Programm «Recall» verwendet die Taste **[E]**, die sich über der Taste **[RCL]** befindet.

Tastenfolge

Anzeige

[g] P/R		Programm-Modus.
[f] CLEAR [PRGM]	000–	
[f] [LBL] [D]	001–42,21,14	«Store» Programm.
[f] [MATRIX] 1	002–42,16, 1	Setzt $R_0 = R_1 = 1$.
[STO] 0	003– 44 0	$R_0 = k$.
[R↓]	004– 33	
0	005– 0	Gibt 0 in das reelle (und imaginäre) X-Register ein.
[+]	006– 40	Bewirkt ein Fallen des Stacks und gibt $a + ib$ in das X-Register.
[f] [USER] [STO] [C]	007u 44 13	Speichert a und erhöht die Indices (User-Modus).
[f] [USER]		
[f] [Re↔Im]	008– 42 30	

Tastenfolge
STO **C**
f **Re $\geq$ Im**
g **RTN**
f **LBL** **E**
STO 0

g **CLx**

2

STO 1

R ∇

0

+
RCL **C**
f **Re $\geq$ Im**
f **DSE** 1

g **CLx**
RCL **C**
g **RTN**
Anzeige
009– 44 13

 Speichert b (jetzt *kein* User-Modus).

010– 42 30

 Speichert $a + ib$ wieder in die X-Register.

011– 43 32
012–42,21,15

«Recall» Programm.

013– 44 0
 $R_0 = k$.

014– 43 35

Sperrt den Stack.

015– 2
016– 44 1

 Setzt $R_1 = 2$.

017– 33
018– 0
019– 40

 Wenn **Error 3** auftritt, ist der Stack für eine neue Eingabe bereit.

020– 45 13

 Ruft b (den Imaginärteil) zurück.

021– 42 30
022–42, 5, 1

 Erniedrigt auf $R_1 = 1$.

023– 43 35

Sperrt den Stack und löscht das reelle X-Register.

024– 45 13

 Ruft a (den Realteil) zurück.

025– 43 32

Beispiel: Speichern Sie mit obigem Programm $2 + 3i$ und $7 + 4i$ als Matrixelemente 1 und 2 ab. Dann rufen Sie sie zurück und addieren Sie sie. Wählen Sie dann die Dimension der Matrix **C** 5×2 , so daß Sie bis zu 5 komplexe Zahlen aufnehmen kann.

Nachdem Sie das obige Programm eingegeben haben:

Tastenfolge
g **P/R**

 5 **ENTER** 2

f **DIM** **C**

 2 **ENTER** 3 **f** **I**

 1 **f** **D**
Anzeige
2

Run-Modus.

Spezifiziert 5 Zeilen und 2 Spalten.

2.0000

 Dimensioniert Matrix **C**.

2.0000

 Gibt $2 + 3i$ ein.

2.0000

 Speichert die Zahl in **C** mit dem Index 1.

Tastenfolge7 **ENTER** 4 **f** **I**2 **f** **D**1 **f** **E**2 **f** **E****+****f** **ReIm****Anzeige****7.0000****7.0000****2.0000****7.0000****9.0000****7.0000**Gibt $7 + 4i$ ein.Speichert die Zahl in **C** mit dem Index 2.

Ruft die erste Zahl zurück.

Ruft die zweite Zahl zurück.

Realteil der Summe.

Imaginärteil der Summe.

Berechnung der n ten Wurzeln einer komplexen Zahl

Dieses Programm berechnet die n ten Wurzeln einer komplexen Zahl. Die Wurzeln werden z_k mit $k = 0, 1, 2, \dots, n-1$ bezeichnet. Sie können das Programm auch zur Berechnung von $z^{1/r}$ benutzen, wo r keine ganze Zahl zu sein braucht. Das Programm läuft ganz genauso ab, nur können für $k = 0, \pm 1, \pm 2, \dots$ unendlich viele Wurzeln z_k auftreten.

Tastenfolge**g** **P/R****f** **CLEAR** **PRGM****f** **LBL** **A****x $\geq$ y****Anzeige****000-****001-42,21,11****002- 34****003- 15****004- 43 36****005- 33****006-43, 4, 8****007- 14****008- 44 2****009- 42 30****010- 44 3****011- 3****012- 6****013- 0****014- 43 33****015- 10****016- 44 4****017- 0****018- 44 25**

Programm-Modus.

Speichert n in das X-Register, z in die Y-Register.Berechnet $1/n$.Lädt n zurück.

Aktiviert den Komplex-Modus.

Berechnet $z^{1/n}$.Speichert den Realteil von z_0 in R_2 .Speichert den Imaginärteil von z_0 in R_3 .Berechnet $360/n$.Speichert $360/n$ in R_4 .

Speichert Null in das Indexregister.

3

6

0

g **R ∇** **÷****STO** 4

0

STO **I**

Tastenfolge

[f] [LBL] 0

[RCL] 4

[RCL] [x] [I]

[f] [ReIm]

[g] [CLx]

1

[g] [DEG]

[f] [↔R]

[RCL] 2

[RCL] 3

[f] [I]

[X]

[RCL] [I]

[x↔y]

1

[STO] [+] [I]

[R↔]

[R/S]

[GTO] 0

Anzeige

019–42,21, 0

020– 45 4

021–45,20,25

022– 42 30

023– 43 35

024– 1

025– 43 7

026– 42 1

027– 45 2

028– 45 3

029– 42 25

030– 20

031– 45 25

032– 34

033– 1

034–44,40,25

035– 33

036– 31

037– 22 0

 Ruft $360/n$ zurück.

 Berechnet mit dem Indexregister $360k/n$.

 Speichert $1 + i(k360/n)$ in das X-Register.

Schaltet in den Altgrad-Modus.

 Berechnet $e^{ik360/n}$.

 Ruft den Realteil von z_0 zurück.

 Ruft den Imaginärteil von z_0 zurück.

 Bildet komplexes z_0 .

 Berechnet $z_0 e^{ik360/n}$, die k -te Wurzel.

 Ruft die Zahl k zurück.

 Speichert z_k in die X-Register, k in das Y-Register.

 Erhöht die Zahl k im Indexregister.

 Speichert z_k und k in die X- und Y-Register zurück.

Unterbricht die Rechnung.

Verzweigung für die nächste Wurzel.

Verwendete Labels: A und 0.

 Verwendete Register: R_2 , R_3 , R_4 und das Indexregister.

Um dieses Programm zu benutzen:

1. Geben Sie die Ordnung n in das Y-Register und die komplexe Zahl z in die X-Register ein.
2. Drücken Sie [f] [A], um die Hauptwurzel z_0 zu berechnen, die im reellen und imaginären X-Register abgelegt wird. (Drücken Sie [f] [(i)] und halten Sie die Taste [(i)] gedrückt, um den Imaginärteil anzuzeigen.)

3. Um die weiteren Wurzeln z_k zu berechnen:

- Drücken Sie $\boxed{\text{R/S}}$, um nacheinander die weiteren Wurzeln z_k zu berechnen. Jede Wurzel z_k wird in den komplexen X-Registern und ihre Nummer k im Y-Register abgelegt. Zwischen der Berechnung von Wurzeln können Sie andere Rechnungen durchführen, ohne dabei das Programm zu berühren (solange Sie nicht R_3 , R_4 , R_5 oder das Indexregister verändern).
- Speichern Sie die Nummer der Wurzel k im Indexregister (mit $\boxed{\text{STO}} \boxed{\text{I}}$), und drücken Sie dann $\boxed{\text{R/S}}$, um z_k zu berechnen. Jede komplexe Wurzel und ihre Nummer werden in die X- und Y-Register gespeichert. (Wenn Sie nochmals $\boxed{\text{R/S}}$ drücken, können Sie weitere Wurzeln mit höheren Nummern berechnen.)

Beispiel: Berechnen Sie mit dem obigen Program $(1)^{1/100}$. Ermitteln Sie für diesen Ausdruck z_0 , z_1 und z_{50} .

Tastenfolge	Anzeige	
$\boxed{\text{g}} \boxed{\text{P/R}}$		Run-Modus.
100 $\boxed{\text{ENTER}}$ 1	1	Gibt $n = 100$ und $z = 1$ (rein reell) ein.
$\boxed{\text{f}} \boxed{\text{A}}$	1.0000	Berechnet z_0 (Realteil).
$\boxed{\text{f}} \boxed{\text{(i)}}$ (gedrückt)	0.0000	Imaginärteil von z_0 .
$\boxed{\text{R/S}}$	0.9980	Berechnet z_1 (Realteil).
$\boxed{\text{f}} \boxed{\text{(i)}}$ (gedrückt)	0.0628	Imaginärteil von z_1 .
50 $\boxed{\text{STO}} \boxed{\text{I}}$	50.0000	Speichert die Nummer der Wurzel im Indexregister.
$\boxed{\text{R/S}}$	-1.0000	Berechnet z_{50} (Realteil).
$\boxed{\text{f}} \boxed{\text{(i)}}$ (gedrückt)	0.0000	Imaginärteil von z_{50} .

Berechnen der komplexen Nullstellen einer Gleichung

Das Newton'sche Iterationsverfahren ist eine häufig verwendete Methode zur numerischen Lösung der komplexen Gleichung $f(z) = 0$. Dieses Verfahren beginnt mit einer Anfangsnäherung z_0 für eine Wurzel und berechnet dann iterativ

$$z_{k+1} = z_k - f(z_k)/f'(z_k)$$

bis die Folge der Näherungen konvergiert.

Das folgende Beispiel zeigt, wie $\boxed{\text{SOLVE}}$ mit dem Newton'schen Iterationsverfahren zum Auffinden von komplexen Nullstellen benutzt werden kann.

(Eine andere Methode, die keinen Gebrauch vom Komplex-Modus macht, wird auf Seite 16 vorgestellt.)

Beispiel: Die Reaktion eines automatisch kontrollierten Systems auf kleine Störungen wird in einem Modell durch die folgende Verzögerungs-Differentialgleichung beschrieben:

$$\frac{d}{dt}w(t) + 9w(t) + 8w(t-1) = 0.$$

Wie stabil ist dieses System? Mit anderen Worten, wie schnell fallen die Lösungen dieser Gleichung ab?

Jede Lösung $w(t)$ kann als Summe in der Form

$$w(t) = \sum_k c(z)e^{zt}$$

mit den konstanten Koeffizienten $c(z)$ für jede Nullstelle z der zu der Verzögerungs-Differentialgleichung gehörigen charakteristischen Gleichung

$$z + 9 + 8e^{-z} = 0$$

ausgedrückt werden. Jede Nullstelle $z = x + iy$ trägt eine Komponente $e^{zt} = e^{xt}(\cos(yt) + i \sin(yt))$ zu $w(t)$ bei, deren Abfallrate um so schneller ist, wie x , der Realteil von z , negativer wird. Daher muß man zur Beantwortung der Frage alle Nullstellen z der charakteristischen Gleichung berechnen. Da diese Gleichung unendlich viele Nullstellen, von denen keine reell ist, besitzt, kann die Berechnung *aller* Nullstellen viel Arbeit bedeuten.

Bekanntlich werden jedoch für große ganzzahlige n die Nullstellen z durch $z \approx A(n) = -\ln((2n + \frac{1}{2})\pi/8) \pm i(2n + \frac{1}{2})\pi$ mit $n = 0, 1, 2, \dots$ angenähert. Je größer n ist, desto besser ist die Näherung. Daher brauchen Sie lediglich die wenigen Nullstellen mit kleinem $|z|$ zu berechnen, für die $A(n)$ keine gute Näherung ist.

Wenn Sie das Newton'sche Iterationsverfahren verwenden, welches $f(z)$ sollten Sie für diese Aufgabe wählen? Die Funktion $f(z) = z + 9 + 8e^{-z}$ ist nicht sehr günstig, da die Exponentialfunktion für größere negative Werte von $\operatorname{Re}(z)$ sehr schnell anwächst. Damit würde die Konvergenz sehr verlangsamt werden, wenn nicht zufällig der (geschätzte) Anfangswert z_0 sehr nahe an einer Nullstelle liegt. Außerdem wird dieses $f(z)$ unendlich oft gleich null, so daß es schwer ist zu entscheiden, wann alle gewünschten Nullstellen berechnet worden sind. Wenn Sie jedoch diese Gleichung zu

$$e^z = -8/(z + 9)$$

umschreiben und dann logarithmieren, erhalten Sie eine äquivalente Gleichung:

$$z = \ln(-8/(z+9)) \pm i2n\pi \quad \text{mit } n = 0, 1, 2, \dots$$

Diese Gleichung besitzt für jede ganze Zahl n nur zwei komplex konjugierte Nullstellen z . Verwenden Sie darum die äquivalente Funktion

$$f(z) = z - \ln(-8/(z+9)) \pm i2n\pi \quad \text{mit } n = 0, 1, 2, \dots$$

und wenden Sie dann folgende Newton'sche Iteration an

$$z_{k+1} = z_k - \ln(-8/(z_k+9)) \pm i2n\pi / (1 + 1/(z_k+9)).$$

Nehmen Sie als erste Näherung für z_0 $A(n)$, das wir oben schon verwendet haben. Mit etwas Algebra und der Tatsache, daß $\ln(\pm i) = \pm i\pi/2$ gilt, erhalten wir:

$$z_{k+1} = A(n) + ((z_k - A(n)) + (z_k + 9) \ln(i \operatorname{Im}(A(n))/(z_k + 9)))/(z_k + 10).$$

Im folgenden Programm wird $\operatorname{Re}(A(n))$ in R_0 und $\operatorname{Im}(A(n))$ in R_1 gespeichert. Beachten Sie, daß für jedes n nur eine Nullstelle des konjugierten Nullstellenpaares berechnet wird.

Tastenfolge

Anzeige

[g] [P/R]		Programm-Modus.
[f] CLEAR [PRGM]	000-	
[f] [LBL] [A]	001-42,21,11	Programm zur Berechnung von $A(n)$.
[g] [CF] 8	002-43, 5, 8	Spezifiziert reelle Arithmetik.
[ENTER]	003- 36	
[+]	004- 40	
[.]	005- 48	
5	006- 5	
[+]	007- 40	
[g] [π]	008- 43 26	
[x]	009- 20	Berechnet $(2n + \frac{1}{2})\pi$.
[ENTER]	010- 36	
[STO] 1	011- 44 1	
8	012- 8	
[÷]	013- 10	
[g] [LN]	014- 43 12	
[CHS]	015- 16	Berechnet $-\ln((2n + \frac{1}{2})\pi/8)$.
[STO] 0	016- 44 0	
[x↔y]	017- 34	
[f] [I]	018- 42 25	Bildet komplexes $A(n)$.

Tastenfolge

[g] [RTN]
 [f] [LBL] [B]

[ENTER]
 [ENTER]
 [RCL] 1
 [f] [Re $\rightarrow$ Im]
 [x $\rightarrow$ y]
 9
 [+]
 [$\div$]
 [g] [LSTx]
 [x $\rightarrow$ y]
 [g] [LN]
 [x]
 [x $\rightarrow$ y]
 [RCL] 1
 [f] [Re $\rightarrow$ Im]
 [RCL] [+ 0
 [-]
 [g] [LSTx]
 [R $\downarrow$]
 [+]
 [x $\rightarrow$ y]
 1
 0
 [+]
 [$\div$]
 [+]
 [g] [RTN]
 [f] [LBL] [C]

[ENTER]
 [e^x]
 9
 [g] [LSTx]
 [+]
 8
 [x $\rightarrow$ y]
 [$\div$]

Anzeige

019- 43 32
 020-42,21,12

Programm zur Berechnung von
 $z_k + 1$.

021- 36
 022- 36
 023- 45 1
 024- 42 30
 025- 34
 026- 9
 027- 40
 028- 10
 029- 43 36
 030- 34
 031- 43 12
 032- 20
 033- 34
 034- 45 1
 035- 42 30
 036-45,40, 0
 037- 30
 038- 43 36
 039- 33
 040- 40
 041- 34
 042- 1
 043- 0
 044- 40
 045- 10
 046- 40
 047- 43 32
 048-42,21,13

Erzeugt $i\text{Im}(A(n))$.

Programm zur Berechnung des
 Residuums $|e^z + 8/(z + 9)|$.

049- 36
 050- 12
 051- 9
 052- 43 36
 053- 40
 054- 8
 055- 34
 056- 10

Tastenfolge	Anzeige	
[+]	057– 40	
[g] [ABS]	058– 43 16	Berechnet $ e^z + 8/(z + 9) $.
[g] [RTN]	059– 43 32	

Verwendete Labels: A, B und C.

Verwendete Register: R₀ und R₁.

Lassen Sie jetzt das Programm ablaufen. Drücken Sie bei jeder Nullstelle solange die Taste **[B]**, bis sich der Realteil in der Anzeige nicht mehr ändert. (Sie könnten auch überprüfen, ob sich der Imaginärteil nicht mehr ändert.)

Tastenfolge	Anzeige	
[g] [P/R]		Run-Modus.
[f] [USER]		Aktiviert den User-Modus.
0 [A]	1.6279	Zeigt $\text{Re}(A(0)) = \text{Re}(z_0)$ an.
[B]	–0.1487	$\text{Re}(z_1)$.
[B]	–0.1497	$\text{Re}(z_2)$.
[B]	–0.1497	$\text{Re}(z)$.
[f] [(i)] (gedrückt)	2.8319	$\text{Im}(z)$.
[C]	1.0000 –10	Berechnet das Residuum.
[x≥y]	–0.1497	Speichert z in das X-Register zurück.

Wenn Sie dieses Programm von $n = 1$ bis 5 wiederholen, erhalten Sie die folgenden Ergebnisse. (Von jedem Paar komplexer Nullstellen ist nur eine aufgeführt.)

n	$A(n)$	Nullstelle z_k	Residuum
0	$1.6279 + i1.5708$	$-0.1497 + i2.8319$	1×10^{-10}
1	$0.0184 + i7.8540$	$-0.4198 + i8.6361$	6×10^{-10}
2	$-0.5694 + i14.1372$	$-0.7430 + i14.6504$	2×10^{-9}
3	$-0.9371 + i20.4204$	$-1.0236 + i20.7868$	5×10^{-10}
4	$-1.2054 + i26.7035$	$-1.2553 + i26.9830$	9×10^{-10}
5	$-1.4167 + i32.9867$	$-1.4486 + i33.2103$	2×10^{-9}

Mit wachsendem n kommt der erste Wert $A(n)$ dem wirklichen Wert z der Nullstelle immer näher. (Wenn Sie fertig sind, drücken Sie die Taste $\boxed{\text{f}} \boxed{\text{USER}}$, um den User-Modus auszuschalten.)

Da die Realteile aller Nullstellen negativ sind, ist das System stabil; jedoch ist die Stabilitätsgrenze (der Realteil mit dem kleinsten Absolutbetrag, nämlich -0.1497) so niedrig, daß die Stabilität des Systems gefährdet ist, wenn es starkem Rauschen ausgesetzt ist.

Konturintegrale

Mit $\boxed{\int}$ können Sie das Konturintegral $\int_C f(z) dz$ berechnen, wo C eine Kurve in der komplexen Ebene bezeichnet.

Parametrisieren Sie zunächst die Kurve C mit $z(t) = x(t) + iy(t)$ im Bereich $t_1 \leq t \leq t_2$. Sei $G(t) = f(z(t))z'(t)$. Dann gilt:

$$\begin{aligned}\int_C f(z) dz &= \int_{t_1}^{t_2} G(t) dt \\ &= \int_{t_1}^{t_2} \operatorname{Re}(G(t)) dt + i \int_{t_1}^{t_2} \operatorname{Im}(G(t)) dt.\end{aligned}$$

Diese Integrale sind genau von der Form, die $\boxed{\int}$ im Komplex-Modus berechnet. Da $G(t)$ eine komplexe Funktion einer reellen Variablen t ist, untersucht $\boxed{\int} G(t)$ in dem Intervall $t_1 \leq t \leq t_2$ und integriert $\operatorname{Re}(G(t))$ – diesen Wert legt die Funktion im reellen X-Register ab. Um den Imaginärteil zu erhalten, integrieren Sie eine Funktion, die $G(t)$ ausgewertet und dann $\operatorname{Im}(G(t))$ mit $\boxed{\operatorname{Re} \rightarrow \operatorname{Im}}$ in das reelle X-Register speichert.

Das folgende allgemeine Programm berechnet das komplexe Integral

$$I = \int_a^b f(z) dz$$

entlang der Geraden von a nach b , wo a und b komplexe Zahlen sind. Es setzt voraus, daß Ihr Unterprogramm zur Berechnung der komplexen Funktion $f(z)$ Label «B» hat und sich die Endpunkte a und b in den komplexen Y- und X-Registern befinden. Das Programm gibt Ihnen als Ergebnis Real- und Imaginärteil von I und deren Genauigkeit ΔI im X- und Y-Register zurück.

Tastenfolge

$\boxed{\text{g}} \boxed{\text{P/R}}$
 $\boxed{\text{f}} \boxed{\text{CLEAR}} \boxed{\text{PRGM}}$

Anzeige

000–

Programm-Modus.

Tastenfolge

f LBL A

x $\bar{z}$ y

-

STO 4

f Re $\bar{z}$ Im

STO 5

g LSTx

STO 6

f Re $\bar{z}$ Im

STO 7

0

ENTER

1

f $\int_f^x$ 0

STO 2

R ∇

STO 3

R ∇ f $\int_f^x$ 1

RCL 2

f I

x $\bar{z}$ y

RCL 3

f I

x $\bar{z}$ y

g RTN

f LBL 0

GSB 1

f Re $\bar{z}$ Im

g RTN

f LBL 1

Anzeige

001-42,21,11

002- 34

003- 30

004- 44 4

005- 42 30

006- 44 5

007- 43 36

008- 44 6

009- 42 30

010- 44 7

011- 0

012- 36

013- 1

014-42,20, 0

015- 44 2

016- 33

017- 44 3

018- 33

019-42,20, 1

020- 45 2

021- 42 25

022- 34

023- 45 3

024- 42 25

025- 34

026- 43 32

027-42,21, 0

028- 32 1

029- 42 30

030- 43 32

031-42,21, 1

Berechnet $b-a$.Speichert $\operatorname{Re}(b-a)$ in R_4 .Speichert $\operatorname{Im}(b-a)$ in R_5 .Ruft a zurück.Speichert $\operatorname{Re}(a)$ in R_6 .Speichert $\operatorname{Im}(a)$ in R_7 .Berechnet $\operatorname{Im}(I)$ und $\operatorname{Im}(\Delta I)$.Speichert $\operatorname{Im}(I)$ in R_2 .Speichert $\operatorname{Im}(\Delta I)$ in R_3 .Berechnet $\operatorname{Re}(I)$ und $\operatorname{Re}(\Delta I)$.Ruft $\operatorname{Im}(I)$ zurück.Bildet komplexes I .Ruft $\operatorname{Im}(\Delta I)$ zurück.Bildet komplexes ΔI .Speichert I in das X-Register zurück.Unterprogramm für $\operatorname{Im}(f(z)z'(t))$.Berechnet $f(z)z'(t)$.

Vertauscht Real- und Imaginärteil.

Unterprogramm, das $f(z)z'(t)$ berechnet.

Tastenfolge

RCL 4

RCL 5

f I

x

RCL 6

RCL 7

f I

+

GSB B

RCL 4

RCL 5

f I

x

g RTN

Anzeige

032- 45 4

033- 45 5

034- 42 25

035- 20

036- 45 6

037- 45 7

038- 42 25

039- 40

040- 32 12

041- 45 4

042- 45 5

043- 42 25

044- 20

045- 43 32

Bildet komplexes $b - a$.Berechnet $(b - a)t$.Bildet komplexes a .Berechnet $a + (b - a)t$.Berechnet $f(a + (b - a)t)$.Bildet komplexes $z'(t) = b - a$.Berechnet $f(z)z'(t)$.

Verwendete Labels: A, 0 und 1.

Verwendete Register: R₂, R₃, R₄, R₅, R₆ und R₇.

Um dieses Programm auszuführen:

1. Geben Sie das Unterprogramm mit dem Label «B» zur Berechnung der Funktion in den Programmspeicher ein.
2. Drücken Sie 7 **f** **DIM** **(i)**, um die Register R₀ bis R₇ freizuhalten. (Ihr Unterprogramm benötigt vielleicht weitere Register.)
3. Wählen Sie das Anzeigeformat für **f**.
4. Geben Sie die zwei Punkte in der komplexen Ebene ein, die die Endpunkte der Geraden sind, entlang der Ihre Funktion integriert werden soll. Der Anfangspunkt sollte in die Y- und der Endpunkt in die X-Register eingegeben werden.
5. Drücken Sie **f** **A**, um das komplexe Integral zu berechnen. Die Werte für das Integral werden in den X-Registern und die Werte für die Fehlerabschätzung in den Y-Registern gespeichert.

Da zwei Integrale berechnet werden müssen, läuft das Programm gewöhnlich länger als für ein reelles Integral, selbst wenn das Programm **f** nicht die gleiche Anzahl von Stützstellen für beide Integrale benutzt. Das einfachere Integral erfordert weniger Rechenaufwand als das schwierigere.

Beispiel: Berechnen Sie näherungsweise die folgenden Integrale:

$$I_1 = \int_1^{\infty} \frac{\cos x}{x + 1/x} dx \quad \text{und} \quad I_2 = \int_1^{\infty} \frac{\sin x}{x + 1/x} dx.$$

Im Übergang x gegen unendlich fallen diese Integranden sehr langsam ab, daher ist ein sehr großes Integrationsintervall erforderlich, und damit dauert die Ausführung lange. Sie können diese Rechnung aber erleichtern, indem Sie den Integrationsweg von der reellen Achse in die komplexe Ebene verlegen. Wie aus der Funktionentheorie bekannt ist, können diese Integrale kombiniert werden zu:

$$I_1 + iI_2 = \int_1^{1+i\infty} \frac{e^{iz}}{z + 1/z} dz.$$

Dieser komplexe Integrand, der entlang der Geraden mit $x=1$ und $y \geq 0$ zu berechnen ist, fällt für große y – genau wie e^{-y} – sehr schnell ab.

Um mit dem obigen Programm beide Integrale gleichzeitig zu berechnen, schreiben Sie ein Unterprogramm, das

$$f(z) = \frac{e^{iz}}{z + 1/z}.$$

berechnet.

Tastenfolge

[f] [LBL] [B]
 [1/x]
 [g] [LSTx]
 [+]
 [g] [LSTx]
 1
 [f] [Re≥Im]
 [x]
 [e^x]
 [x≥y]
 [÷]
 [g] [RTN]

Anzeige

046–42,21,12
 047– 15
 048– 43 36
 049– 40
 050– 43 36
 051– 1
 052– 42 30
 053– 20
 054– 12
 055– 34
 056– 10
 057– 43 32

Berechnet $z + 1/z$.

Bildet $0 + i$.

Berechnet e^{iz} .

Berechnet $f(z)$.

Berechnen Sie das komplexe Integral näherungsweise, indem Sie die Funktion von $1 + 0i$ bis $1 + 6i$ im Anzeigeformat [SCI] 2 integrieren, was Ihnen drei signifikante Ziffern liefert. (Jenseits von $1 + 6i$ ändert das Integral die ersten drei Ziffern nicht mehr.)

Tastenfolge	Anzeige		
[g] [P/R]			Run-Modus.
[f] [SCI] 2			Wählt das Format [SCI] 2.
1 [ENTER]	1.00	00	Gibt die erste Integrationsgrenze $1 + 0i$ ein.
1 [ENTER] 6	6		
[f] [I]	1.00	00	Gibt die zweite Integrationsgrenze $1 + 6i$ ein.
[f] [A]	-3.24	-01	Berechnet I und zeigt $\operatorname{Re}(I) = I_1$ an (nach etwa 9 Minuten).
[f] [(i)] (gedrückt)	3.82	-01	Zeigt $\operatorname{Im}(I) = I_2$.
[x$\rightrightarrows$y]	7.87	-04	Zeigt $\operatorname{Re}(\Delta I) = \Delta I_1$ an.
[f] [(i)] (gedrückt)	1.23	-03	Zeigt $\operatorname{Im}(\Delta I) = \Delta I_2$ an.
[f] [FIX] 4	0.0008		

Dieses Ergebnis I wird sehr viel schneller als die reellen Integrale I_1 und I_2 berechnet.

Komplexe Potentiale

In Anwendungen, in denen eine komplexe Potentialfunktion vorkommt, sind konforme Abbildungen ein nützliches Hilfsmittel. Die folgende Diskussion behandelt Aufgabenstellungen der Strömungslehre, doch können auch Probleme der Elektrostatik und Wärmeleitung analog behandelt werden.

Betrachten wir die Potentialfunktion $P(z)$. Die Gleichung $\operatorname{Im}(P(z)) = c$ definiert eine Schar von Kurven, die man *Stromlinien* des Flusses nennt. Das heißt, für einen beliebigen Wert von c liegen alle Werte von z , die die Gleichung erfüllen, auf einer Stromlinie mit dem Wert c . Um einige Punkte z_k auf der Stromlinie zu berechnen, geben Sie Werte x_k vor, und benutzen Sie dann **[SOLVE]**, das mit der folgenden Gleichung die zugehörigen Werte y_k findet:

$$\operatorname{Im}(P(x_k + iy_k)) = c.$$

Wenn die Werte x_k nicht zu weit auseinanderliegen, können Sie y_{k-1} als Anfangsnäherung für y_k wählen. So arbeiten Sie sich an der Stromlinie entlang und berechnen die komplexen Punkte $z_k = x_k + iy_k$. Auf ähnliche Weise können Sie die Äquipotentiallinien definieren, die durch $\operatorname{Re}(P(z)) = c$ gegeben sind.

Das folgende Programm berechnet Werte für y_k aus äquidistanten x_k . Sie müssen ein Unterprogramm mit dem Label «B» vorgeben, das $\text{Im}(P(z))$ im reellen X-Register ablegt. Das Programm benötigt Eingaben, die die Schrittweite h , die Anzahl n von Punkten auf der reellen Achse und $z_0 = x_0 + iy_0$, den Anfangspunkt auf der Stromlinie spezifizieren. Bevor Sie das Programm laufen lassen, müssen Sie also n , h und z_0 in die Z-, Y- und X-Register eingeben.

Das Programm berechnet die Werte z_k und speichert sie in der Form $a_{k1} = x_{k-1}$ und $a_{k2} = y_{k-1}$ mit $k = 1, 2, \dots, n$ in der Matrix A.

Tastenfolge	Anzeige	
[g] [P/R]		Programm-Modus.
[f] [CLEAR] [PRGM]	000–	
[f] [LBL] [A]	001–42,21,11	
[R↓]	002– 33	
[STO] 4	003– 44 4	Speichert h in R_4 .
[R↓]	004– 33	
2	005– 2	
[f] [DIM] [A]	006–42,23,11	Dimensioniert A als $n \times 2$ Matrix.
[g] [CLx]	007– 43 35	
[STO] [MATRIX] [A]	008–44,16,11	Setzt alle Matrixelemente von A gleich 0.
[STO] [I]	009– 44 25	Speichert 0 im Indexregister.
[f] [MATRIX] 1	010–42,16, 1	Setzt $R_0 = R_1 = 1$.
[g] [R↓]	011– 43 33	Ruft z_0 in die X-Register zurück.
[STO] 2	012– 44 2	Speichert x_0 in R_2 .
[f] [USER] [STO] [A]	013u 44 11	Setzt $a_{11} = x_0$.
[f] [USER]		
[f] [ReIm]	014– 42 30	
[STO] 3	015– 44 3	Speichert y_0 in R_3 .
[f] [USER] [STO] [A]	016u 44 11	Setzt $a_{12} = y_0$.
[f] [USER]		
[GTO] 1	017– 22 1	Verzweigt, wenn die Matrix A nicht voll ist ($n > 1$).
[f] [LBL] 0	018–42,21, 0	
[RCL] [MATRIX] [A]	019–45,16,11	Ruft den Deskriptor der Matrix A zurück.

Tastenfolge

g **RTN**
f **LBL** 1
f **Re $\rightarrow$ Im**
GSB **B**

STO 5
f **LBL** 2
1
STO **+** **I**

RCL 4
RCL **I**

$\times$
RCL 2
+

STO 6
RCL 3
ENTER

f **SOLVE** 3
GTO 4

1
STO **-** **I**
4
STO **$\div$** 4
STO **$\times$** **I**
GTO 2

f **LBL** 4
RCL 6
f **PSE**
f **USER** **STO** **A**
f **USER**

Anzeige

020– 43 32
021–42,21, 1
022– 42 30
023– 32 12

Speichert z_0 zurück.
Berechnet $\text{Im}(P(z_0))$ (oder $\text{Re}(P(z_0))$ für Äquipotentiallinien).

024– 44 5
025–42,21, 2
026– 1
027–44,40,25

Speichert c in R_5 .
Schleife, um y_k zu finden.
Erhöht den Zähler k im Indexregister.

028– 45 4
029– 45 25
030– 20

Ruft h zurück.
Ruft den Zähler k zurück.
Berechnet kh .

031– 45 2
032– 40

Ruft x_0 zurück.
Berechnet $x_k = x_0 + kh$.

033– 44 6
034– 45 3

Speichert x_k in R_6 .
Ruft y_{k-1} aus R_3 zurück.

035– 36

Dupliziert y_{k-1} für zweite Näherung.

036–42,10, 3
037– 22 4

Sucht y_k .
Verzweigung; y_k wurde gefunden.

038– 1
039–44,30,25

Verkleinert Schrittweite.
Verkleinert Zähler k .

040– 4
041–44,10, 4

Reduziert h um den Faktor 4.
Multipliziert den Zähler mit 4.

042–44,20,25
043– 22 2

Springt zurück, um y_k neu zu berechnen.

044–42,21, 4
045– 45 6

Fortsetzung bei gefundenem y_k .

046– 42 31
047u 44 11

Zeigt x_k an.
Setzt $a_{k+1,1} = x_k$.

Tastenfolge	Anzeige	
R↓	048– 33	
f PSE	049– 42 31	Zeigt y_k an.
STO 3	050– 44 3	Speichert y_k in R_3 .
f USER STO A	051 u 44 11	Setzt $a_{k+1,2} = y_k$.
f USER		
GTO 2	052– 22 2	Verzweigung für $k+1 < n$ (A ist noch nicht voll).
GTO 0	053– 22 0	Verzweigung für $k+1 = n$ (A ist voll).
f LBL 3	054–42,21, 3	Funktionsunterprogramm für SOLVE .
RCL 6	055– 45 6	Ruft x_k zurück.
x$\hat{=}$y	056– 34	Speichert momentane Näherung für y_k zurück.
f I	057– 42 25	Erzeugt Näherung $z_k = x_k + iy_k$.
GSB B	058– 32 12	Berechnet $\text{Im}(P(z_k))$ (oder $\text{Re}(P(z_k))$ für Äquipotentiallinien).
RCL 5	059– 45 5	Ruft c zurück.
-	060– 30	Berechnet $\text{Im}(P(z_k)) - c$.
g RTN	061– 43 32	

Verwendete Labels: A, B, 0, 1, 2, 3 und 4.

Verwendete Register: R_0 , R_1 , R_2 (x_0), R_3 (y_0), R_4 (h), R_5 (c), R_6 (x_k) und Indexregister (k).

Verwendete Matrix: A.

Eine besondere Eigenschaft dieses Programmes ist, daß es die Schrittweite verkleinert, wenn ein Wert x_k außerhalb des Wertebereichs der Stromlinie liegt (und daher **SOLVE** keine Nullstelle finden kann), so daß x_k nahe an die Grenze kommt, wo die Stromlinie umkehrt. Mit dieser Eigenschaft kann das Verhalten der Stromlinie untersucht werden, wenn y_k keine eindeutige Funktion von x_k ist. Die Werte z_k liegen auf einem Zweig der Stromlinie und nähern die Grenze an, wenn h klein genug gewählt wird. (Das zweite der folgenden Beispiele demonstriert diese Eigenschaft.)

Um dieses Programm zu verwenden:

1. Geben Sie Ihr Unterprogramm mit dem Label «B» in den Programmspeicher ein. Es sollte $\text{Im}(P(z))$ im reellen X-Register ablegen, wenn Sie Stromlinien berechnen, oder $\text{Re}(P(z))$, wenn Sie Äquipotentiallinien berechnen.
2. Drücken Sie 6 $\boxed{f}$ $\boxed{\text{DIM}}$ $\boxed{(i)}$, um die Register R_0 bis R_6 (und das Indexregister) freizuhalten. (Ihr Unterprogramm benötigt vielleicht weitere Register.)
3. Geben Sie die Werte n und h in die Z- und Y-Register ein, indem Sie n $\boxed{\text{ENTER}}$ h $\boxed{\text{ENTER}}$ drücken.
4. Geben Sie den komplexen Wert $z_0 = x_0 + iy_0$ in die X-Register ein, indem Sie x_0 $\boxed{\text{ENTER}}$ y_0 $\boxed{f}$ $\boxed{I}$ drücken.
5. Drücken Sie $\boxed{f}$ $\boxed{A}$, um die Werte x_k und y_k für $k = 1, \dots, n$ und den Deskriptor der Matrix **A** anzuzeigen. Die Werte für $k = 1, \dots, n$ sind in der Matrix **A** gespeichert.
6. Wenn Sie wollen, können Sie Werte der Matrix **A** zurückrufen.

Beispiel: Berechnen Sie die Stromlinie des Potentials $P(z) = 1/z + z$ durch den Punkt $z = -2 + 0.1i$.

Geben Sie zunächst das Unterprogramm «B» ein, das $\text{Im}(P(z))$ berechnet.

Tastenfolge	Anzeige	
$\boxed{f}$ $\boxed{\text{LBL}}$ $\boxed{B}$	062–42,21,12	
$\boxed{\text{ENTER}}$	063– 36	Dupliziert z .
$\boxed{1/x}$	064– 15	
$\boxed{+}$	065– 40	Berechnet $1/z + z$.
$\boxed{f}$ $\boxed{\text{Re} \rightarrow \text{Im}}$	066– 42 30	Speichert $\text{Im}(P(z))$ in das X-Register.
$\boxed{g}$ $\boxed{\text{RTN}}$	067– 43 32	

Berechnen Sie die Stromlinie, indem Sie $z_0 = -2 + 0.1i$, die Schrittweite $h = 0.5$ und als Anzahl von Punkten $n = 9$ verwenden.

Tastenfolge	Anzeige	
$\boxed{g}$ $\boxed{\text{P/R}}$		Run-Modus.
9 $\boxed{\text{ENTER}}$	9.0000	Gibt n ein.
$.5$ $\boxed{\text{ENTER}}$	0.5000	Gibt h ein.

Tastenfolge

2 **CHS** **ENTER**

.1 **f** **I**

f **A**

g **CF** 8

Anzeige

-2.0000

-2.0000

-1.5000

0.1343

⋮

2.0000

0.1000

A **9 2** Deskriptor der Matrix **A**.

A **9 2** Desaktiviert den Komplex-Modus.

Gibt z_0 ein.

x_1 .

y_1 .

⋮

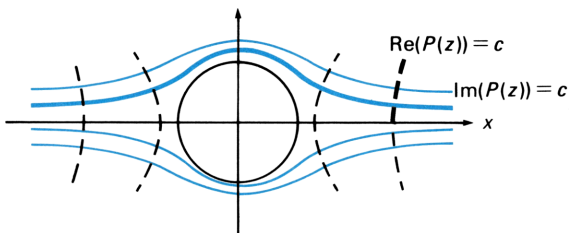
x_9 .

y_9 .

Die Matrix **A** enthält die folgenden Werte x_k und y_k :

x_k	y_k
-2.0	0.1000
-1.5	0.1343
-1.0	0.4484
-0.5	0.9161
0.0	1.0382
0.5	0.9161
1.0	0.4484
1.5	0.1343
2.0	0.1000

Die Stromlinie und Äquipotentiallinien der Geschwindigkeit sind in der folgenden Abbildung dargestellt. Die durchgezogene dicke Linie ist die berechnete Stromlinie.



Beispiel: Berechnen Sie für das Potential des vorigen Beispiels $P(z) = 1/z + z$ die Äquipotentiallinie der Geschwindigkeit vom Punkt $z = 2 + i$ aus weiter nach links.

Ändern Sie zunächst das Unterprogramm «B», so daß es $\operatorname{Re}(P(z))$ berechnet – d.h. lassen Sie die Anweisung `ReIm` aus «B» aus. Nehmen Sie die Werte $n = 6$ und $h = -0.5$. (Beachten Sie, daß h negativ ist und damit bestimmt wird, daß die x_k links von x_0 liegen.)

Die Tastenfolge ist hier zwar nicht angegeben, aber wir zeigen im folgenden die Ergebnisse, die das Programm berechnet und in der Matrix **A** speichert.

x_k	y_k
2.0000	1.0000
1.8750	0.2362
1.8672	0.1342
1.8652	0.0941
1.8647	0.0811
1.8646	0.0775

Das Ergebnis zeigt das Verhalten des oberen Zweiges der Kurve (die dicke gestrichelte Linie des vorigen Beispiels). Beachten Sie, wie die Schrittweite h automatisch verkleinert wird, um der Kurve zu folgen, wenn für $x < 1.86$ keine y -Werte gefunden werden, anstatt mit einer Fehlermeldung abubrechen.

Matrixoperationen

Die Methoden der Matrixalgebra ermöglichen die vereinfachte Formulierung komplizierter Probleme und erleichtern so ansonsten sehr mühsame Rechnungen. In diesem Abschnitt finden Sie Informationen darüber, wie Ihr HP-15C gewisse Matrizenoperationen ausführt und wie Sie diese in ihren Anwendungen verwenden können.

In diesem Abschnitt stellen wir mehrere Methoden zur numerischen Behandlung von Problemen der linearen Algebra zusammen, die nicht im einzelnen erläutert werden. Eine ausführlichere Darstellung können Sie in der einschlägigen Fachliteratur finden.*

Zum Verständnis der *LR*-Zerlegung

Ihr HP-15C kann lineare Gleichungssysteme lösen, Matrizen invertieren und Determinanten berechnen. Zur Durchführung dieser Berechnungen transformiert der HP-15C eine quadratische Matrix in ihre sogenannte *LR*-Zerlegung, die numerisch einfacher zu handhaben ist.

Der *LR*-Algorithmus zerlegt eine quadratische Matrix **A** in das Produkt **LR** zweier Matrizen **L** und **R**. **L** ist eine linke untere Dreiecksmatrix** mit 1-ern auf der Diagonalen und Elementen mit Werten zwischen -1 und +1 unter der Diagonalen. **R** ist eine rechte obere Dreiecksmatrix**. Beispielsweise:

$$\mathbf{A} = \begin{bmatrix} 2 & 3 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ .5 & 1 \end{bmatrix} \begin{bmatrix} 2 & 3 \\ 0 & -.5 \end{bmatrix} = \mathbf{LR}.$$

* Zwei solcher Fachbücher sind

Atkinson, Kendall E., *An Introduction to Numerical Analysis*, Wiley, 1978.

Kahan, W., «Numerical Linear Algebra», *Canadian Mathematical Bulletin*, Volume 9, 1966, pp. 756–801.

** Eine linke untere Dreiecksmatrix enthält über der Diagonalen nur Nullen. Eine rechte obere Dreiecksmatrix enthält unter der Diagonalen nur Nullen.

Es gibt Matrizen, die nicht in die LR -Form gebracht werden können. Beispielsweise gibt es *kein* Paar von linken unteren und rechten oberen Dreiecksmatrizen $\mathbf{L}$ und $\mathbf{R}$, für die gilt:

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ 1 & 2 \end{bmatrix} = \mathbf{LR}$$

Eine LR -Zerlegung kann jedoch immer durchgeführt werden, wenn die Zeilen der zu faktorisierenden Matrix vertauscht werden. Das Vertauschen von Zeilen der Matrix $\mathbf{A}$ kann durch das Produkt mit einer quadratischen Matrix $\mathbf{P}$ dargestellt werden. Die LR -Zerlegung erhält man dann in der Form $\mathbf{PA} = \mathbf{LR}$.

$$\mathbf{PA} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix} = \mathbf{LR}.$$

Zeilenvertauschungen können auch Rundungsfehler reduzieren, die bei der Berechnung der Zerlegung auftreten können.

Um die LR -Zerlegung durchzuführen, verwendet der HP-15C den Doolittle Algorithmus mit erhöht genauer Arithmetik. Die zerlegte Matrix wird in der Ergebnismatrix abgelegt und dabei in der Form

$$\begin{bmatrix} & \mathbf{R} \\ \mathbf{L} & \end{bmatrix}$$

gespeichert. Die Diagonalelemente von $\mathbf{L}$ brauchen nicht gespeichert zu werden, da sie immer gleich 1 sind. Die Zeilenvertauschungen sind auch in derselben Matrix (in einer kodierten, für Sie nicht sichtbaren Form) gespeichert. Die Zerlegung einer Matrix wird durch zwei Bindestriche im Deskriptor der Matrix gekennzeichnet.

Wenn Sie eine Determinante berechnen oder ein Gleichungssystem lösen, wird die LR -Zerlegung automatisch gespeichert. In einem späteren Rechengang kann die zerlegte Form einer Matrix sehr nützlich sein. In diesem Fall ist es wichtig, daß Sie die Information über Zeilenvertauschungen, die in der Matrix enthalten ist, nicht zerstören; daher dürfen Sie die Matrix, in der die Zerlegung gespeichert ist, nicht verändern!

Um beispielsweise die Determinante einer Matrix $\mathbf{A}$ zu berechnen, verwendet der HP-15C die Gleichung $\mathbf{A} = \mathbf{P}^{-1}\mathbf{LR}$, die Zeilenvertauschungen ermöglicht. Die Determinante ist dann gerade $(-1)^r$ mal des Produkt der Diagonalelemente von $\mathbf{R}$, wo r die Anzahl von Zeilenvertauschungen ist. Der HP-15C zerlegt erst die Matrix und berechnet dann dieses Produkt mit dem richtigen Vorzeichen. Wenn die Matrix bereits zerlegt ist, berechnet er nur das Produkt mit seinem Vorzeichen.

Eine obere oder untere Dreiecksmatrix ist leichter zu invertieren als eine allgemeine quadratische Matrix. Um beispielsweise die Inverse einer Matrix $\mathbf{A}$ zu berechnen, benutzt der HP-15C die folgende Beziehung:

$$\mathbf{A}^{-1} = (\mathbf{P}^{-1}\mathbf{LR})^{-1} = \mathbf{R}^{-1}\mathbf{L}^{-1}\mathbf{P}.$$

Um dies zu berechnen, zerlegt er erst die Matrix $\mathbf{A}$, invertiert dann sowohl $\mathbf{L}$ als auch $\mathbf{R}$, berechnet dann ihr Produkt $\mathbf{R}^{-1}\mathbf{L}^{-1}$ und vertauscht dann die Spalten des Ergebnisses. Dies erfolgt innerhalb der Ergebnismatrix, die $\mathbf{A}$ selbst sein kann. Wenn die Matrix $\mathbf{A}$ bereits zerlegt vorliegt, wird der erste Schritt ausgelassen. Mit dieser Methode kann der HP-15C eine Matrix invertieren, ohne zusätzliche Speicherregister zu benötigen.

Des weiteren ist es auch einfacher, ein Gleichungssystem $\mathbf{AX} = \mathbf{B}$ mit einer oberen oder unteren Dreiecksmatrix $\mathbf{A}$ zu lösen, als mit einer allgemeinen quadratischen Matrix $\mathbf{A}$. Mit $\mathbf{PA} = \mathbf{LR}$ ist die Gleichung äquivalent zu $\mathbf{LRX} = \mathbf{PB}$. Die Zeilen von $\mathbf{B}$ werden auf die gleiche Weise vertauscht, wie die Zeilen von $\mathbf{A}$ während der Zerlegung vertauscht wurden. Der HP-15C berechnet erst $\mathbf{Y}$ aus $\mathbf{LY} = \mathbf{PB}$ (Vorwärtssubstitution) und dann $\mathbf{X}$ aus $\mathbf{RX} = \mathbf{Y}$ (Rückwärtssubstitution). Die LR-Zerlegung wird gespeichert, damit Sie die Lösungen $\mathbf{X}$ für verschiedene Matrizen $\mathbf{B}$ finden können, ohne die Matrix $\mathbf{A}$ des Gleichungssystems neu eingeben zu müssen.

Die LR-Zerlegung ist ein wichtiger Zwischenschritt bei der Berechnung von Determinanten, der Invertierung von Matrizen und dem Lösen von linearen Gleichungssystemen. Anstelle der ursprünglichen Matrix kann die LR-Zerlegung als Eingabe für diese Rechnungen verwendet werden.

Schlechtkonditionierte Matrizen und die Konditionszahl

Um Fehler bei Berechnungen mit Matrizen diskutieren zu können, ist es nützlich, ein Maß für den Abstand zweier Matrizen zu definieren. Ein Maß für den Abstand der Matrizen $\mathbf{A}$ und $\mathbf{B}$ ist die *Norm* ihrer Differenz, die mit $\|\mathbf{A} - \mathbf{B}\|$ bezeichnet wird. Mit der Norm kann auch die *Konditionszahl* einer Matrix definiert werden, die angibt, wie groß der relative Fehler einer Berechnung im Vergleich zu dem relativen Fehler in der Matrix selbst ist.

Der HP-15C verwendet drei Normen. Die *Frobenius-Norm* einer Matrix $\mathbf{A}$ ($\|\mathbf{A}\|_F$) ist die Wurzel aus der Summe der Quadrate der Matrixelemente. Diese Norm ist das Analog für Matrizen zu der euklidischen Länge eines Vektors.

Eine weitere Norm des HP-15C ist die *Zeilensummennorm*. Die Zeilensummennorm einer $m \times n$ Matrix $\mathbf{A}$ ist die größte Summe der Absolutbeträge der Elemente einer Zeile und wird mit $\|\mathbf{A}\|_R$ bezeichnet:

$$\|\mathbf{A}\|_R = \max_{1 \leq i \leq m} \sum_{j=1}^n |a_{ij}|.$$

Die *Spaltensummennorm* einer Matrix wird mit $\|\mathbf{A}\|_C$ bezeichnet und wird $\|\mathbf{A}\|_C = \|\mathbf{A}^T\|_R$ berechnet. Die Spaltensummennorm ist die größte Summe der Absolutbeträge der Elemente einer Spalte.

Betrachten wir beispielsweise die Matrizen

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 9 \end{bmatrix} \quad \text{und} \quad \mathbf{B} = \begin{bmatrix} 2 & 2 & 2 \\ 4 & 5 & 6 \end{bmatrix}.$$

Dann ist

$$\mathbf{A} - \mathbf{B} = \begin{bmatrix} -1 & 0 & 1 \\ 0 & 0 & 3 \end{bmatrix}$$

und

$$\|\mathbf{A} - \mathbf{B}\|_F = \sqrt{11} \approx 3.3 \text{ (Frobenius-Norm),}$$

$$\|\mathbf{A} - \mathbf{B}\|_R = 3 \text{ (Zeilensummennorm) und}$$

$$\|\mathbf{A} - \mathbf{B}\|_C = 4 \text{ (Spaltensummennorm).}$$

Im weiteren Verlauf dieser Diskussion wird die Zeilensummennorm unterstellt. Bei Verwendung einer der anderen Normen erhält man ähnliche Ergebnisse.

Die *Konditionszahl* einer quadratischen Matrix $\mathbf{A}$ ist folgendermaßen definiert:

$$K(\mathbf{A}) = \|\mathbf{A}\| \|\mathbf{A}^{-1}\|.$$

Es gilt dann unabhängig von der verwendeten Norm $1 \leq K(\mathbf{A}) < \infty$. Die Konditionszahl ist nützlich, um Ungenauigkeiten in Rechnungen abzuschätzen. Eine Matrix $\mathbf{A}$ heißt *schlechtkonditioniert*, wenn $K(\mathbf{A})$ sehr groß ist.

Wenn die Matrixelemente Rundungs- oder andere Fehler enthalten, werden sich diese Fehler durch die nachfolgenden Matrixrechnungen fortpflanzen. Sie können sich dabei erheblich vergrößern. Nehmen wir beispielsweise an, $\mathbf{X}$ und $\mathbf{B}$ seien Vektoren ungleich null mit $\mathbf{A}\mathbf{X} = \mathbf{B}$, wo $\mathbf{A}$ eine beliebige quadratische Matrix ist. Angenommen, $\mathbf{A}$ sei um die Störung $\Delta\mathbf{A}$ ungenau, und wir wollen $\mathbf{B} + \Delta\mathbf{B} = (\mathbf{A} + \Delta\mathbf{A})\mathbf{X}$ berechnen. Dann gilt

$$\frac{(\|\Delta\mathbf{B}\| / \|\mathbf{B}\|)}{(\|\Delta\mathbf{A}\| / \|\mathbf{A}\|)} \leq K(\mathbf{A}),$$

wo das Gleichheitszeichen für irgendeine Störung $\Delta\mathbf{A}$ gilt. Dies ist ein Maß dafür, wie sehr sich der relative Fehler in $\mathbf{A}$ vergrößert, wenn er sich in das Produkt fortpflanzt.

Die Konditionszahl ist auch ein Maß dafür, um wieviel größer der relative Fehler der Lösung eines Gleichungssystems sein kann als der der eingegebenen Daten. Nehmen wir wieder an, $\mathbf{X}$ und $\mathbf{B}$ seien Vektoren ungleich null mit $\mathbf{A}\mathbf{X} = \mathbf{B}$ und irgendeiner Matrix $\mathbf{A}$. Nehmen wir nun an, die Matrix $\mathbf{B}$ sei (z.B. durch Rundungsfehler) um einen Betrag $\Delta\mathbf{B}$ ungenau. Für $\mathbf{X} + \Delta\mathbf{X}$ gelte $\mathbf{A}(\mathbf{X} + \Delta\mathbf{X}) = \mathbf{B} + \Delta\mathbf{B}$. Dann gilt

$$\frac{(\|\Delta\mathbf{X}\| / \|\mathbf{X}\|)}{(\|\Delta\mathbf{B}\| / \|\mathbf{B}\|)} \leq K(\mathbf{A}),$$

wo die Gleichheit für irgendeine Störung $\Delta\mathbf{B}$ gilt.

Nehmen wir nun stattdessen an, die Matrix $\mathbf{A}$ sei um $\Delta\mathbf{A}$ ungenau. Für $\mathbf{X} + \Delta\mathbf{X}$ gelte $(\mathbf{A} + \Delta\mathbf{A})(\mathbf{X} + \Delta\mathbf{X}) = \mathbf{B}$. Wenn $d(\mathbf{A}, \Delta\mathbf{A}) = K(\mathbf{A})\|\Delta\mathbf{A}\|/\|\mathbf{A}\| < 1$ ist, dann gilt

$$\frac{(\|\Delta\mathbf{X}\| / \|\mathbf{X}\|)}{(\|\Delta\mathbf{A}\| / \|\mathbf{A}\|)} \leq K(\mathbf{A}) / (1 - d(\mathbf{A}, \Delta\mathbf{A})).$$

Ebenso, wenn $\mathbf{A}^{-1} + \mathbf{Z}$ die Inverse der gestörten Matrix $\mathbf{A} + \Delta\mathbf{A}$ ist, dann gilt

$$\frac{(\|\mathbf{Z}\| / \|\mathbf{A}^{-1}\|)}{(\|\Delta\mathbf{A}\| / \|\mathbf{A}\|)} \leq K(\mathbf{A}) / (1 - d(\mathbf{A}, \Delta\mathbf{A})).$$

Weiterhin existieren Störungen $\Delta\mathbf{A}$, für die die Ungleichheiten zu Gleichheiten werden.

Alle obigen Beziehungen zeigen, wie der relative Fehler des Ergebnisses mit dem relativen Fehler der Matrix $\mathbf{A}$ über die Konditionszahl $K(\mathbf{A})$ zusammenhängt. Bei jeder Ungleichung gibt es Matrizen, für die die Gleichheit gilt. Bei einer großen Konditionszahl ist ein relativ großer Fehler im Ergebnis möglich.

Ungenauigkeiten in den Daten – manchmal nur sehr kleine relative Fehler – können dazu führen, daß die Lösung eines schlecht konditionierten Gleichungssystems sich erheblich von der Lösung des ursprünglichen Systems unterscheidet. Ebenso kann sich die Inverse einer schlecht konditionierten Matrix erheblich von der Inversen der ungestörten Matrix unterscheiden. Beide Unterschiede sind jedoch durch die Konditionszahl beschränkt; sie können *nur* dann relativ groß werden, wenn die Konditionszahl $K(\mathbf{A})$ groß ist.

Eine große Konditionszahl $K(\mathbf{A})$ einer nichtsingulären Matrix $\mathbf{A}$ ist auch ein Zeichen dafür, daß die Matrix $\mathbf{A}$ relativ dicht (in ihrer Norm) bei einer singulären Matrix liegt. Das heißt

$$1/K(\mathbf{A}) = \min(\|\mathbf{A} - \mathbf{S}\|/\|\mathbf{A}\|)$$

und

$$1/\|\mathbf{A}^{-1}\| = \min(\|\mathbf{A} - \mathbf{S}\|),$$

wo das Minimum über alle singulären Matrizen $\mathbf{S}$ gebildet wird. Das heißt, wenn $K(\mathbf{A})$ groß ist, dann ist der relative Abstand zwischen $\mathbf{A}$ und der nächsten singulären Matrix $\mathbf{S}$ klein. Wenn die Norm von $\mathbf{A}^{-1}$ groß ist, ist der Abstand zur nächsten singulären Matrix $\mathbf{S}$ auch klein.

Sei beispielsweise

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 1 & .9999999999 \end{bmatrix}.$$

Dann gilt

$$\mathbf{A}^{-1} = \begin{bmatrix} -9,999,999,999 & 10^{10} \\ 10^{10} & -10^{10} \end{bmatrix}$$

und $\|\mathbf{A}^{-1}\| = 2 \times 10^{10}$. Daher sollte ein Störung $\Delta\mathbf{A}$ existieren, mit $\|\Delta\mathbf{A}\| = 5 \times 10^{-11}$, so daß $\mathbf{A} + \Delta\mathbf{A}$ singulär wird. Tatsächlich, wenn

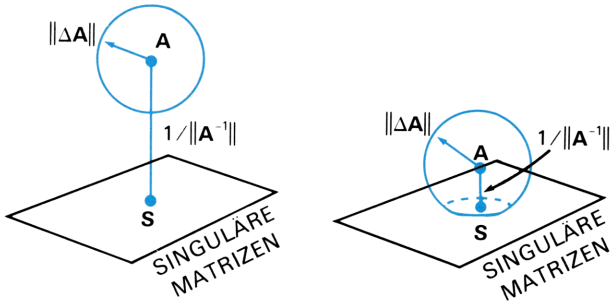
$$\Delta\mathbf{A} = \begin{bmatrix} 0 & -5 \times 10^{-11} \\ 0 & 5 \times 10^{-11} \end{bmatrix}$$

mit $\|\Delta\mathbf{A}\| = 5 \times 10^{-11}$ ist, dann gilt

$$\mathbf{A} + \Delta\mathbf{A} = \begin{bmatrix} 1 & .99999999995 \\ 1 & .99999999995 \end{bmatrix}$$

und $\mathbf{A} + \Delta\mathbf{A}$ ist singulär.

Die folgenden Abbildungen verdeutlichen diese Ideen. In den beiden Abbildungen sind die Matrizen $\mathbf{A}$ und $\mathbf{S}$ relativ zur «Ebene» aller singulären Matrizen im Raum aller Matrizen gezeichnet. Abstände werden mit der Norm gemessen. Um jede Matrix $\mathbf{A}$ gibt es eine Umgebung von Matrizen, die von $\mathbf{A}$ praktisch nicht unterscheidbar sind (z.B. solche innerhalb der Rundungsfehler von $\mathbf{A}$). Der Radius dieser Umgebung ist $\|\Delta\mathbf{A}\|$. Der Abstand einer nichtsingulären Matrix $\mathbf{A}$ zur nächstgelegenen singulären Matrix $\mathbf{S}$ beträgt $1/\|\mathbf{A}^{-1}\|$.



Im linken Diagramm ist $\|\Delta\mathbf{A}\| < 1/\|\mathbf{A}^{-1}\|$.

Wenn $\|\Delta\mathbf{A}\| \ll 1/\|\mathbf{A}^{-1}\|$ (oder $K(\mathbf{A})\|\mathbf{A}\|/\|\mathbf{A}\| \ll 1$) ist, dann gilt

Relative Variation in $\mathbf{A}^{-1} = \|\text{Änderung in } \mathbf{A}^{-1}\|/\|\mathbf{A}^{-1}\|$

$$\approx (\|\Delta\mathbf{A}\|/\|\mathbf{A}\|)K(\mathbf{A})$$

$$= \|\Delta\mathbf{A}\|/(1/\|\mathbf{A}^{-1}\|)$$

$$= (\text{Radius der Kugel})/(\text{Abstand von der Ebene})$$

Im rechten Diagramm ist $\|\Delta\mathbf{A}\| > 1/\|\mathbf{A}^{-1}\|$. In diesem Fall existiert eine von $\mathbf{A}$ nicht unterscheidbare singuläre Matrix, und der Versuch, die Inverse $\mathbf{A}^{-1}$ von $\mathbf{A}$ zu berechnen, kann daher zu sinnlosen Ergebnissen führen.

Genauigkeit von numerischen Lösungen linearer Gleichungssysteme

In der vorangegangenen Diskussion ging es darum, wie Ungenauigkeiten der Eingabedaten die Lösungen von linearen Gleichungssystemen und Matrixinversionen beeinflussen. Doch selbst wenn die Daten genau sind, werden numerisch berechnete Lösungen und Inverse ungenau.

Betrachten wir die Auflösung des linearen Gleichungssystems $\mathbf{AX} = \mathbf{B}$ nach der theoretischen Lösung $\mathbf{X}$. Aufgrund von Rundungsfehlern während der Rechnungen ist im allgemeinen die berechnete Lösung $\mathbf{Z}$ nicht die Lösung des ursprünglichen Systems $\mathbf{AX} = \mathbf{B}$, sondern die Lösung eines gestörten Systems $(\mathbf{A} + \Delta\mathbf{A})\mathbf{Z} = \mathbf{B}$. Für die Störung $\Delta\mathbf{A}$ gilt $\|\Delta\mathbf{A}\| \leq \epsilon \|\mathbf{A}\|$, mit einem im allgemeinen sehr kleinen ϵ . In vielen Fällen macht $\Delta\mathbf{A}$ weniger als die 10te Stelle bei jedem Element von $\mathbf{A}$ aus.

Für eine berechnete Lösung $\mathbf{Z}$ bezeichnet man $\mathbf{R} = \mathbf{B} - \mathbf{AZ}$ als *Residuum*. Dann gilt $\|\mathbf{R}\| \leq \epsilon \|\mathbf{A}\| \|\mathbf{Z}\|$. Das für eine berechnete Lösung zu erwartende Residuum ist also klein. Doch obwohl das Residuum $\mathbf{R}$ im allgemeinen klein ist, braucht der Fehler $\mathbf{Z} - \mathbf{X}$ nicht klein zu sein, wenn $\mathbf{A}$ schlecht konditioniert ist:

$$\|\mathbf{Z} - \mathbf{X}\| \leq \epsilon \|\mathbf{A}\| \|\mathbf{A}^{-1}\| \|\mathbf{Z}\| = \epsilon K(\mathbf{A}) \|\mathbf{Z}\|.$$

Eine nützliche Daumenregel für die Genauigkeit einer berechneten Lösung ist

$$\left(\begin{array}{c} \text{Anzahl von} \\ \text{korrekten} \\ \text{Dezimalstellen} \end{array} \right) \geq \left(\begin{array}{c} \text{Anzahl von in} \\ \text{der Rechnung mitge-} \\ \text{führten Dezimalstellen} \end{array} \right) - \log(\|\mathbf{A}\| \|\mathbf{A}^{-1}\|) - \log(10n)$$

wo n die Dimension der Matrix $\mathbf{A}$ ist. Für den HP-15C, der mit 10 Dezimalstellen rechnet, gilt

$$(\text{Anzahl von korrekten Dezimalstellen}) \geq 9 - \log(\|\mathbf{A}\| \|\mathbf{A}^{-1}\|) - \log(n).$$

Für viele Anwendungen reicht diese Genauigkeit aus. Ist eine höhere Genauigkeit erforderlich, kann die berechnete Lösung $\mathbf{Z}$ im allgemeinen durch *Nachiteration* verbessert werden.

Bei der Nachiteration wird zunächst eine Lösung zu einem Gleichungssystem berechnet und dann ihre Genauigkeit dadurch verbessert, daß die Lösung mit Hilfe des Residuums (zu dieser Lösung) modifiziert wird.

Um eine Nachiteration durchzuführen, berechnen Sie zunächst eine Lösung $\mathbf{Z}$ des ursprünglichen Gleichungssystems $\mathbf{AX} = \mathbf{B}$. $\mathbf{Z}$ wird dann als Näherung für

$\mathbf{X}$ mit dem Fehler $\mathbf{E} = \mathbf{X} - \mathbf{Z}$ betrachtet. Dann erfüllt $\mathbf{E}$ das lineare Gleichungssystem $\mathbf{A}\mathbf{E} = \mathbf{A}\mathbf{X} - \mathbf{A}\mathbf{Z} = \mathbf{R}$, wo $\mathbf{R}$ das Residuum für $\mathbf{Z}$ ist. Der nächste Schritt besteht darin, das Residuum zu berechnen und dann $\mathbf{A}\mathbf{E} = \mathbf{R}$ nach $\mathbf{E}$ aufzulösen. Die berechnete Lösung $\mathbf{F}$ wird als Näherung für $\mathbf{E} = \mathbf{X} - \mathbf{Z}$ betrachtet und wird zu $\mathbf{Z}$ addiert, um eine zweite Näherung für $\mathbf{X}$ zu erhalten: $\mathbf{F} + \mathbf{Z} \approx (\mathbf{X} - \mathbf{Z}) + \mathbf{Z} = \mathbf{X}$.

Damit $\mathbf{F} + \mathbf{Z}$ eine bessere Näherung für $\mathbf{X}$ als $\mathbf{Z}$ ist, muß das Residuum $\mathbf{R} = \mathbf{B} - \mathbf{A}\mathbf{Z}$ mit erhöhter Genauigkeit berechnet werden. Die Operation **MATRIX** 6 des HP-15C ermöglicht dies. Um die beiden Lösungen $\mathbf{Z}$ und $\mathbf{F}$ zu finden, wird die Matrix $\mathbf{A}$ des Gleichungssystems benutzt. Die LR -Zerlegung, die schon bei der Berechnung von $\mathbf{Z}$ durchgeführt wurde, kann nun auch dazu benutzt werden, $\mathbf{F}$ zu berechnen, was die Rechenzeit verkürzt. Die Nachiteration kann wiederholt werden, doch der größte Teil der Verbesserung tritt nach der ersten Iteration auf.

(In den Anwendungsbeispielen am Ende dieses Abschnitts finden Sie ein Programm, das eine Nachiteration durchführt.)

Vereinfachung von schwierigen Gleichungssystemen

Es ist schwierig, ein lineares Gleichungssystem $\mathbf{E}\mathbf{X} = \mathbf{B}$ numerisch genau zu lösen, wenn $\mathbf{E}$ schlecht konditioniert (beinahe singulär) ist. Wenn $\mathbf{E}$ hinreichend schlecht konditioniert ist, kann selbst Nachiteration die berechnete Lösung nicht weiter verbessern. Es gibt jedoch praktische Beispiele, in denen es gelingt, mit wenig zusätzlichem Aufwand schwierig zu lösende Gleichungssysteme in leichter zu lösende mit derselben Lösung umzuwandeln. Zwei Möglichkeiten dies zu tun, sind Skalieren und Vorkonditionieren.

Skalierung

Schlechte Skalierung ist häufig die Ursache für unbefriedigende Ergebnisse bei dem Versuch, schlecht konditionierte Matrizen numerisch zu invertieren oder Gleichungssysteme mit schlecht konditionierten Matrizen des Systems numerisch zu lösen. Doch es ist eine Ursache, die Sie leicht finden und beheben können.

Angenommen, Sie haben aus der Matrix $\mathbf{A}$ eine Matrix $\mathbf{E}$ mit $\mathbf{E} = \mathbf{L}\mathbf{A}\mathbf{R}$ erhalten, wo $\mathbf{L}$ und $\mathbf{R}$ skalierende Diagonalmatrizen sind, deren Diagonalelemente alle ganze Potenzen von 10 sind. Dann sagt man, $\mathbf{E}$ sei aus $\mathbf{A}$ durch *Skalierung* hervorgegangen. $\mathbf{L}$ skaliert die Zeilen und $\mathbf{R}$ die Spalten von $\mathbf{A}$. $\mathbf{E}^{-1} = \mathbf{R}^{-1}\mathbf{A}^{-1}\mathbf{L}^{-1}$ kann entweder aus $\mathbf{A}^{-1}$ durch Skalieren oder aus $\mathbf{E}$ durch Invertieren ermittelt werden.

Sei beispielsweise die folgende Matrix $\mathbf{A}$ gegeben:

$$\mathbf{A} = \begin{bmatrix} 3 \times 10^{-40} & 1 & 2 \\ 1 & 1 & 1 \\ 2 & 1 & -1 \end{bmatrix}.$$

Der HP-15C berechnet $\mathbf{A}^{-1}$ mit 10stelliger Genauigkeit zu

$$\mathbf{A}^{-1} \approx \begin{bmatrix} -2 & 3 & -1 \\ 3 & -4 & 2 \\ -1 & 2 & -1 \end{bmatrix}.$$

Nun sei

$$\mathbf{L} = \mathbf{R} = \begin{bmatrix} 10^{20} & 0 & 0 \\ 0 & 10^{-20} & 0 \\ 0 & 0 & 10^{-20} \end{bmatrix}$$

so daß

$$\mathbf{E} = \begin{bmatrix} 3 & 1 & 2 \\ 1 & 10^{-40} & 10^{-40} \\ 2 & 10^{-40} & -10^{-40} \end{bmatrix}.$$

ist. $\mathbf{E}$ ist der singulären Matrix

$$\mathbf{S} = \begin{bmatrix} 3 & 1 & 2 \\ 1 & 0 & 0 \\ 2 & 0 & 0 \end{bmatrix}$$

sehr nahe, und $\|\mathbf{E} - \mathbf{S}\|/\|\mathbf{E}\| = \frac{1}{3} \times 10^{-40}$. Damit ist $K(\mathbf{S}) \geq 3 \times 10^{40}$, so daß es nicht verwundert, daß das berechnete $\mathbf{E}^{-1}$

$$\mathbf{E}^{-1} \approx \begin{bmatrix} -6.67 \times 10^{-11} & 1 & 10^{-10} \\ 0.8569 & 8.569 \times 10^9 & -4.284 \times 10^9 \\ 0.07155 & -4.284 \times 10^9 & 2.142 \times 10^9 \end{bmatrix}$$

sich von dem wahren Wert

$$\mathbf{E}^{-1} = \begin{bmatrix} -2 \times 10^{-40} & 3 & -1 \\ 3 & -4 \times 10^{40} & 2 \times 10^{40} \\ -1 & 2 \times 10^{40} & -10^{40} \end{bmatrix}.$$

erheblich unterscheidet. Wie schlecht die berechnete Inverse ist, können Sie sehen, wenn Sie sie mit der ursprünglichen Matrix multiplizieren.

Das Problem ist, daß $\mathbf{E}$ schlecht skaliert ist. Bei einer gut skalierten Matrix wie $\mathbf{A}$ haben ihrer Zeilen und Spalten (in der Norm) etwa die gleiche Größenordnung, *und* das gleiche gilt für ihre Inverse. Die Normen der Zeilen und Spalten von $\mathbf{E}$ haben etwa die gleiche Größenordnung wie die von $\mathbf{A}$, aber die erste Zeile und Spalte von $\mathbf{E}^{-1}$ haben eine viel kleinere Norm als die anderen. Daher sollten die Zeilen und Spalten von $\mathbf{E}$ vor dem Invertieren der Matrix skaliert werden. Das heißt, daß die oben besprochenen Diagonalmatrizen $\mathbf{L}$ und $\mathbf{R}$ so gewählt werden sollten, daß $\mathbf{LER}$ und $(\mathbf{LER})^{-1} = \mathbf{R}^{-1}\mathbf{E}^{-1}\mathbf{L}^{-1}$ nicht so schlecht skaliert sind.

Im allgemeinen kennen Sie natürlich den wahren Wert der inversen Matrix $\mathbf{E}^{-1}$ nicht im voraus. Daher muß das Erkennen einer schlechten Skalierung der Matrix $\mathbf{E}$ und die Wahl der Skaliermatrizen $\mathbf{L}$ und $\mathbf{R}$ von $\mathbf{E}$ und dem *berechneten* $\mathbf{E}^{-1}$ ausgehen. Das oben berechnete $\mathbf{E}^{-1}$ erweist sich als schlecht skaliert, und man könnte versuchen

$$\mathbf{L} = \mathbf{R} = \begin{bmatrix} 10^{-5} & 0 & 0 \\ 0 & 10^5 & 0 \\ 0 & 0 & 10^5 \end{bmatrix}.$$

zu wählen. Mit diesen Skaliermatrizen wird

$$\mathbf{LER} = \begin{bmatrix} 3 \times 10^{-10} & 1 & 2 \\ 1 & 10^{-30} & 10^{-30} \\ 2 & 10^{-30} & -10^{-30} \end{bmatrix},$$

was zwar noch immer schlecht skaliert ist, aber nicht so schlecht, daß Ihr HP-15C damit nicht fertig würde. Die berechnete Inverse ist

$$(\mathbf{LER})^{-1} = \begin{bmatrix} -2 \times 10^{-30} & 3 & -1 \\ 3 & -4 \times 10^{30} & 2 \times 10^{30} \\ -1 & 2 \times 10^{30} & -10^{30} \end{bmatrix}.$$

Dieses Ergebnis ist auf 10 Stellen genau, obwohl Sie das nicht wissen können. Das Ergebnis kann aber auch so überprüft werden, daß mit der berechneten Inversen $(\mathbf{LER})^{-1}(\mathbf{LER}) = (\mathbf{LER})(\mathbf{LER})^{-1} = \mathbf{I}$ (Einheitsmatrix) bis auf 10 Dezimalstellen gilt.

Damit erhält man

$$\mathbf{E}^{-1} = \mathbf{R}(\mathbf{LER})^{-1}\mathbf{L} = \begin{bmatrix} -2 \times 10^{-40} & 3 & -1 \\ 3 & -4 \times 10^{40} & 2 \times 10^{40} \\ -1 & 2 \times 10^{40} & -10^{40} \end{bmatrix},$$

was auf 10 Dezimalstellen stimmt.

Wenn auch $(\mathbf{LER})^{-1}$ sich noch als schlecht erweist, können Sie den Skaliervorgang wiederholen, mit $\mathbf{LER}$ anstelle von $\mathbf{E}$ und mit neuen Skaliermatrizen, die Sie aus $\mathbf{LER}$ und dem berechneten $\mathbf{LER}^{-1}$ bestimmen müssen.

Sie können den Skaliervorgang auch dann anwenden, wenn Sie beispielsweise ein Gleichungssystem $\mathbf{EX} = \mathbf{B}$ mit schlecht skaliertem $\mathbf{E}$ lösen wollen. Wenn Sie $\mathbf{X}$ berechnen, ersetzen Sie das Gleichungssystem $\mathbf{EX} = \mathbf{B}$ durch ein System $(\mathbf{LER})\mathbf{Y} = \mathbf{LB}$, aus dem Sie $\mathbf{Y}$ bestimmen. Die diagonalen Skaliermatrizen $\mathbf{L}$ und $\mathbf{R}$ werden wieder so gewählt, daß die Matrix $\mathbf{LER}$ gut skaliert wird. Berechnen Sie erst $\mathbf{Y}$ aus dem neuen System und dann die gewünschte Lösung $\mathbf{X} = \mathbf{RY}$.

Vorkonditionierung

Eine andere Methode, ein schwieriges Gleichungssystem $\mathbf{EX} = \mathbf{B}$ durch ein einfacheres $\mathbf{AX} = \mathbf{D}$ zu ersetzen, ist die Vorkonditionierung.

Angenommen, $\mathbf{E}$ sei schlecht konditioniert (beinahe singulär). Sie können dies feststellen, indem Sie die Inverse $\mathbf{E}^{-1}$ berechnen und überprüfen, ob $1/\|\mathbf{E}^{-1}\|$ sehr klein im Vergleich zu $\|\mathbf{E}\|$ ist (oder äquivalent dazu, ob $\mathbf{E}$ eine große Konditionszahl $K(\mathbf{E})$ hat). Dann hat fast jeder Zeilenvektor $\mathbf{u}^T$ die Eigenschaft, daß $\|\mathbf{u}^T\|/\|\mathbf{u}^T\mathbf{E}^{-1}\|$ sehr klein ist im Vergleich zu $\|\mathbf{E}\|$, wo $\mathbf{E}^{-1}$ die berechnete Inverse ist. Das liegt daran, daß für die meisten Zeilenvektoren $\mathbf{u}^T\|\mathbf{u}^T\mathbf{E}^{-1}\|$ nicht wesentlich kleiner als $\|\mathbf{u}^T\|\|\mathbf{E}^{-1}\|$ ist und $\|\mathbf{E}^{-1}\|$ groß ist. Wählen Sie nun einen solchen Zeilenvektor $\mathbf{u}^T$ und berechnen Sie $\mathbf{v}^T = \mathbf{a}\mathbf{u}^T\mathbf{E}^{-1}$. Wählen Sie den Skalar a so, daß der Zeilenvektor $\mathbf{r}^T$, den Sie durch Runden aller Elemente von $\mathbf{v}^T$ auf eine ganze Zahl zwischen -100 und 100 erhalten, sich nicht wesentlich von $\mathbf{v}^T$ unterscheidet. Dann ist $\mathbf{r}^T$ ein Zeilenvektor mit ganzzahligen Elementen mit Beträgen kleiner als 100 . $\|\mathbf{r}^T\mathbf{E}\|$ ist klein im Vergleich zu $\|\mathbf{r}^T\|\|\mathbf{E}\|$ – je kleiner, desto besser.

Definieren Sie nun eines der betragsmäßig größten Elemente von $\mathbf{r}^T$ als das k -te Element. Ersetzen Sie die k -te Zeile von $\mathbf{E}$ durch $\mathbf{r}^T$ und die k -te Zeile von $\mathbf{B}$ durch $\mathbf{r}^T \mathbf{B}$. Wenn keine Rundungen bei der Berechnung dieser neuen Zeilen erforderlich waren, sollte die neue Matrix $\mathbf{A}$ des Gleichungssystems besser konditioniert (weniger singulär) als $\mathbf{E}$ sein; doch das neue Gleichungssystem hat noch dieselbe Lösung $\mathbf{X}$.

Dieses Verfahren funktioniert am besten, wenn $\mathbf{E}$ und $\mathbf{A}$ so skaliert sind, daß alle Zeilen von $\mathbf{E}$ und $\mathbf{A}$ ungefähr die gleiche Norm haben. Sie können dieses erreichen, indem Sie die Zeilen der Gleichungssysteme $\mathbf{E}\mathbf{X} = \mathbf{B}$ und $\mathbf{A}\mathbf{X} = \mathbf{D}$ mit passenden Zehnerpotenzen multiplizieren. Sollte $\mathbf{A}$ zwar gut skaliert, doch noch «zu singulär» sein, müssen Sie eine weitere Vorkonditionierung durchführen.

Als Beispiel für die Vorkonditionierung wollen wir das Gleichungssystem $\mathbf{E}\mathbf{X} = \mathbf{B}$ betrachten, mit

$$\mathbf{E} = \begin{bmatrix} x & y & y & y & y \\ y & x & y & y & y \\ y & y & x & y & y \\ y & y & y & x & y \\ y & y & y & y & x \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

wo $x = 8000.00002$ und $y = -1999.99998$ sei. Wenn Sie dieses Gleichungssystem direkt zu lösen versuchen, berechnet Ihr HP-15C als Lösung $\mathbf{X}$ und Inverse $\mathbf{E}^{-1}$

$$\mathbf{X} \approx \begin{bmatrix} 2014.6 \\ 2014.6 \\ 2014.6 \\ 2014.6 \\ 2014.6 \end{bmatrix} \text{ und } \mathbf{E}^{-1} \approx 2014.6 \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Wenn Sie dies einsetzen, erhalten Sie

$$\mathbf{E}\mathbf{X} \approx \begin{bmatrix} 1.00146 \\ 0.00146 \\ 0.00146 \\ 0.00146 \\ 0.00147 \end{bmatrix}.$$

Sie können nun (mit **MATRIX** 7) überprüfen, daß $1/\|\mathbf{E}^{-1}\| \approx 9.9 \times 10^{-5}$ ist, was sehr klein im Vergleich zu $\|\mathbf{E}\| \approx 1.6 \times 10^4$ ist (oder daß die Konditionszahl groß ist nämlich $\|\mathbf{E}\|\|\mathbf{E}^{-1}\| \approx 1.6 \times 10^8$).

Wählen Sie einen beliebigen Zeilenvektor

$$\mathbf{u}^T = (1, 1, 1, 1, 1) \text{ und berechnen Sie}$$

dann

$$\mathbf{u}^T \mathbf{E}^{-1} \approx 10073 (1, 1, 1, 1, 1).$$

Mit $a = 10^{-4}$ wird

$$\mathbf{v}^T = a \mathbf{u}^T \mathbf{E}^{-1} \approx 1.0073 (1, 1, 1, 1, 1)$$

$$\mathbf{r}^T = (1, 1, 1, 1, 1)$$

$$\|\mathbf{r}^T \mathbf{E}\| \approx 5 \times 10^{-4}$$

$$\|\mathbf{r}^T\| \|\mathbf{E}\| \approx 8 \times 10^4.$$

Wie erwartet, ist $\|\mathbf{r}^T \mathbf{E}\|$ klein verglichen mit $\|\mathbf{r}^T\| \|\mathbf{E}\|$.

Ersetzen wir nun die erste Zeile von $\mathbf{E}$ durch

$$10^7 \mathbf{r}^T \mathbf{E} = (1000, 1000, 1000, 1000, 1000)$$

und die erste Zeile von $\mathbf{B}$ durch $10^7 \mathbf{r}^T \mathbf{B} = 10^7$. Damit erhalten wir ein neues Gleichungssystem $\mathbf{A}\mathbf{X} = \mathbf{D}$, mit

$$\mathbf{A} = \begin{bmatrix} 1000 & 1000 & 1000 & 1000 & 1000 \\ y & x & y & y & y \\ y & y & x & y & y \\ y & y & y & x & y \\ y & y & y & y & x \end{bmatrix} \text{ und } \mathbf{D} = \begin{bmatrix} 10^7 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

Beachten Sie, wie $\mathbf{r}^T \mathbf{E}$ mit 10^7 skaliert wurde, so daß alle Zeilen von $\mathbf{A}$ und $\mathbf{E}$ etwa dieselbe Norm haben. Mit diesem neuen Gleichungssystem berechnet Ihr HP-15C die Lösung

$$\mathbf{X} = \begin{bmatrix} 2000.000080 \\ 1999.999980 \\ 1999.999980 \\ 1999.999980 \\ 1999.999980 \end{bmatrix}, \text{ mit } \mathbf{A}\mathbf{X} = \begin{bmatrix} 10^7 \\ -10^{-5} \\ -9 \times 10^{-6} \\ 0 \\ 0 \end{bmatrix}.$$

Die Lösung unterscheidet sich von der vorigen und ist auf 10 Dezimalstellen genau.

Manchmal werden die Elemente einer fast singulären Matrix $\mathbf{E}$ mit einer Formel berechnet, zu der Rundungen so viele Fehler beitragen, daß die berechnete Inverse $\mathbf{E}^{-1}$ falsch sein muß, selbst wenn bei ihrer Berechnung exakte Arithmetik verwendet wurde. In diesem Fall nützt Vorkonditionierung nur, wenn sie so auf die Formel angewandt wird, daß die modifizierte Zeile von $\mathbf{A}$ exakt berechnet werden kann. Mit anderen Worten müssen Sie die Formel mit der Vorkonditionierung genau in eine neue und bessere umrechnen, wenn Ihnen das Verfahren nützen soll.

Berechnungen mit kleinsten Quadraten

Matrixoperationen kommen häufig in Rechnungen mit *kleinsten Quadraten* vor. Die typische kleinste Quadrate Aufgabe besteht aus einer $n \times p$ Matrix $\mathbf{X}$ beobachter Daten und einem Vektor $\mathbf{y}$ aus n Meßwerten, zu denen ein Vektor $\mathbf{b}$ mit p Komponenten gesucht ist, der den Ausdruck

$$\|\mathbf{r}\|_F^2 = \sum_{i=1}^n r_i^2$$

in dem $\mathbf{r} = \mathbf{y} - \mathbf{X}\mathbf{b}$ der Residuumsvektor ist, minimiert.

Normalgleichungen

Aus dem vorigen Ausdruck erhalten wir

$$\|\mathbf{r}\|_F^2 = (\mathbf{y} - \mathbf{X}\mathbf{b})^T (\mathbf{y} - \mathbf{X}\mathbf{b}) = \mathbf{y}^T \mathbf{y} - 2\mathbf{b}^T \mathbf{X}^T \mathbf{y} + \mathbf{b}^T \mathbf{X}^T \mathbf{X} \mathbf{b}.$$

Das Lösen der kleinste Quadrate Aufgabe ist äquivalent dazu, eine Lösung $\mathbf{b}$ der *Normalgleichungen*

$$\mathbf{X}^T \mathbf{X} \mathbf{b} = \mathbf{X}^T \mathbf{y}.$$

zu finden. Diese Normalgleichungen sind jedoch für Rundungsfehler sehr empfindlich. (Die Orthogonalzerlegung auf S. 113 dagegen ist für Rundungsfehler relativ unempfindlich.)

Die *gewichtete Methode der kleinsten Quadrate* ist eine Verallgemeinerung der gewöhnlichen Methode der kleinsten Quadrate. Hier wollen Sie

$$\|\mathbf{W}\mathbf{r}\|_F^2 = \sum_{i=1}^n w_i^2 r_i^2$$

minimieren, wo $\mathbf{W}$ eine $n \times n$ Diagonalmatrix mit positiven Diagonalelementen $w_1, w_2, \dots, w_n$ ist. Dann gilt

$$\|\mathbf{W}\mathbf{r}\|_F^2 = (\mathbf{y} - \mathbf{X}\mathbf{b})^T \mathbf{W}^T \mathbf{W} (\mathbf{y} - \mathbf{X}\mathbf{b})$$

und jede Lösung $\mathbf{b}$ erfüllt auch die gewichteten Normalgleichungen

$$\mathbf{X}^T \mathbf{W}^T \mathbf{W} \mathbf{X} \mathbf{b} = \mathbf{X}^T \mathbf{W}^T \mathbf{W} \mathbf{y}.$$

Dies sind die Normalgleichungen, wenn man $\mathbf{X}$ und $\mathbf{y}$ durch $\mathbf{W}\mathbf{X}$ und $\mathbf{W}\mathbf{y}$ ersetzt. Daher sind auch diese Gleichungen für Rundungsfehler empfindlich.

Bei der *Methode der kleinsten Quadrate mit linearer Nebenbedingung* ist ein $\mathbf{b}$ gesucht, das

$$\|\mathbf{r}\|_F^2 = \|\mathbf{y} - \mathbf{X}\mathbf{b}\|_F^2$$

minimiert, unter der Nebenbedingung, daß

$$\mathbf{C}\mathbf{b} = \mathbf{d} \quad \left(\sum_{j=1}^k c_{ij} b_j = d_i \text{ mit } i = 1, 2, \dots, m \right).$$

Dies ist äquivalent dazu, eine Lösung $\mathbf{b}$ der *erweiterten Normalgleichungen*

$$\begin{bmatrix} \mathbf{X}^T \mathbf{X} & \mathbf{C}^T \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{b} \\ \mathbf{l} \end{bmatrix} = \begin{bmatrix} \mathbf{X}^T \mathbf{y} \\ \mathbf{d} \end{bmatrix}$$

zu finden, wo $\mathbf{l}$, ein Vektor aus Lagrange'schen Multiplikatoren, ein Teil der Lösung ist, aber weiter nicht mehr verwendet wird. Auch die erweiterten Normalgleichungen sind wieder sehr empfindlich für Rundungsfehler. Beachten Sie, daß auch wieder durch Ersetzen von $\mathbf{X}$ und $\mathbf{y}$ durch $\mathbf{W}\mathbf{X}$ und $\mathbf{W}\mathbf{y}$ gewichtet werden kann.

Betrachten wir als Beispiel dafür, wie die Normalgleichungen für die Lösung von kleinste Quadrate Aufgaben numerisch ungeeignet sein können, die Werte

$$\mathbf{X} = \begin{bmatrix} 100,000. & -100,000. \\ 0.1 & 0.1 \\ 0.2 & 0.0 \\ 0.0 & 0.2 \end{bmatrix} \quad \text{und} \quad \mathbf{y} = \begin{bmatrix} 0.1 \\ 0.1 \\ 0.1 \\ 0.1 \end{bmatrix}.$$

Dann gilt

$$\mathbf{X}^T \mathbf{X} = \begin{bmatrix} 10,000,000,000.05 & -9,999,999,999.99 \\ -9,999,999,999.99 & 10,000,000,000.05 \end{bmatrix}$$

und

$$\mathbf{X}^T \mathbf{y} = \begin{bmatrix} 10,000.03 \\ -9,999.97 \end{bmatrix}.$$

Wenn wir jedoch auf 10 Stellen runden, erhalten wir

$$\mathbf{X}^T \mathbf{X} \approx \begin{bmatrix} 10^{10} & -10^{10} \\ -10^{10} & 10^{10} \end{bmatrix},$$

was wir auch erhalten würden, wenn $\mathbf{X}$ auf fünf signifikante Stellen relativ zu dem größten Element gerundet wäre:

$$\mathbf{X} = \begin{bmatrix} 100,000 & -100,000 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

Ihr HP-15C löst $\mathbf{X}^T \mathbf{X} \mathbf{b} = \mathbf{X}^T \mathbf{y}$ (die singuläre Matrix wird gestört wie auf S. 118 beschrieben) und erhält

$$\mathbf{b} = \begin{bmatrix} 0.060001 \\ 0.060000 \end{bmatrix}$$

mit

$$\mathbf{X}^T \mathbf{y} - \mathbf{X}^T \mathbf{X} \mathbf{b} = \begin{bmatrix} 0.03 \\ 0.03 \end{bmatrix}.$$

Die exakte kleinste Quadrate Lösung ist jedoch

$$\mathbf{b} = \begin{bmatrix} 0.5000005 \\ 0.4999995 \end{bmatrix}$$

obwohl die berechnete und die exakte Lösung die berechneten Normalgleichungen gleich gut erfüllen.

Die Normalgleichungen sollten nur dann verwendet werden, wenn alle Elemente von $\mathbf{X}$ kleine ganze Zahlen (zwischen etwa -3000 und 3000) sind oder wenn Sie wissen, daß keine Veränderungen der Spalten $\mathbf{x}_j$ von $\mathbf{X}$ um bis zu $\|\mathbf{x}_j\|/10^4$ diese Spalten linear abhängig machen könnten.

Orthogonal-Faktorisierung

Die folgende Methode der Orthogonal-Faktorisierung löst die kleinste Quadrate-Aufgabe, ist aber weniger für Rundungsfehler empfindlich als die Normalgleichungsmethode. Sie können diese Methode verwenden, wenn die Normalgleichungen ungeeignet sind.

Jede $n \times p$ Matrix $\mathbf{X}$ kann in $\mathbf{X} = \mathbf{Q}^T \mathbf{U}$ zerlegt werden, wo $\mathbf{Q}$ eine orthogonale Matrix mit der Eigenschaft $\mathbf{Q}^T = \mathbf{Q}^{-1}$ und $\mathbf{U}$ eine obere $n \times p$ -Dreiecksmatrix ist. Das wesentliche Merkmal von orthogonalen Matrizen ist, daß sie Längen in dem Sinn erhalten, daß

$$\begin{aligned} \|\mathbf{Q}\mathbf{r}\|_F^2 &= (\mathbf{Q}\mathbf{r})^T (\mathbf{Q}\mathbf{r}) \\ &= \mathbf{r}^T \mathbf{Q}^T \mathbf{Q} \mathbf{r} \\ &= \mathbf{r}^T \mathbf{r} \\ &= \|\mathbf{r}\|_F^2. \end{aligned}$$

gilt. Daher hat $\mathbf{r} = \mathbf{y} - \mathbf{X}\mathbf{b}$ dieselbe Länge wie

$$\mathbf{Q}\mathbf{r} = \mathbf{Q}\mathbf{y} - \mathbf{Q}\mathbf{X}\mathbf{b} = \mathbf{Q}\mathbf{y} - \mathbf{U}\mathbf{b}.$$

Die obere Dreiecksmatrix $\mathbf{U}$ und das Produkt $\mathbf{Qy}$ können als

$$\mathbf{U} = \begin{bmatrix} \hat{\mathbf{U}} \\ \mathbf{0} \end{bmatrix} \begin{matrix} (p \text{ Zeilen}) \\ (n-p \text{ Zeilen}) \end{matrix} \quad \text{und} \quad \mathbf{Qy} = \begin{bmatrix} \mathbf{g} \\ \mathbf{f} \end{bmatrix} \begin{matrix} (p \text{ Zeilen}) \\ (n-p \text{ Zeilen}) \end{matrix}$$

geschrieben werden. Dann gilt

$$\begin{aligned} \|\mathbf{r}\|_F^2 &= \|\mathbf{Qr}\|_F^2 \\ &= \|\mathbf{Qy} - \mathbf{Ub}\|_F^2 \\ &= \|\mathbf{g} - \hat{\mathbf{U}}\mathbf{b}\|_F^2 + \|\mathbf{f}\|_F^2 \\ &\geq \|\mathbf{f}\|_F^2 \end{aligned}$$

wo die Gleichheit für $\mathbf{g} - \hat{\mathbf{U}}\mathbf{b} = \mathbf{0}$ gilt. Mit anderen Worten: Die Lösung der gewöhnlichen kleinste Quadrate Aufgabe ist eine beliebige Lösung von $\hat{\mathbf{U}}\mathbf{b} = \mathbf{g}$ und die kleinste Quadratsumme ist $\|\mathbf{f}\|_F^2$. Dies ist die Grundlage von allen numerisch guten kleinste Quadrate Algorithmen.

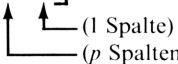
Die kleinste Quadrate Aufgabe ohne Nebenbedingungen können Sie in zwei Stufen lösen:

1. Führen Sie die Orthogonal-Faktorisierung mit der erweiterten $n \times (p+1)$ -Matrix

$$\begin{bmatrix} \mathbf{X} & \mathbf{y} \end{bmatrix} = \mathbf{Q}^T \mathbf{V}$$

durch, wo $\mathbf{Q}^T = \mathbf{Q}^{-1}$ gilt, und behalten Sie nur die obere Dreiecksmatrix $\mathbf{V}$, die Sie nun folgendermaßen aufteilen können:

$$\mathbf{V} = \begin{bmatrix} \hat{\mathbf{U}} & \mathbf{g} \\ \mathbf{0} & q \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{matrix} (p \text{ Zeilen}) \\ (1 \text{ Zeile}) \\ (n-p-1 \text{ Zeile}) \end{matrix}$$



Sie brauchen nur die ersten $p+1$ Zeilen (und Spalten) von $\mathbf{V}$ zu behalten. (Beachten Sie, daß dieses $\mathbf{Q}$ nicht identisch ist mit dem $\mathbf{Q}$ weiter oben, da dieses auch $\mathbf{y}$ transformieren soll.)

2. Bestimmen Sie aus dem folgenden Gleichungssystem **b**:

$$\begin{bmatrix} \hat{\mathbf{U}} & \mathbf{g} \\ \mathbf{0} & q \end{bmatrix} \begin{bmatrix} \mathbf{b} \\ -1 \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ -q \end{bmatrix}.$$

(Sollte $q = 0$ sein, ersetzen Sie es durch eine kleine Zahl, z.B. 10^{-99} .) In der Lösungsmatrix erscheint ohne zusätzliche Rechnungen automatisch die -1 .

Wenn keine Rundungsfehler auftreten gilt $q = \pm \|\mathbf{y} - \mathbf{X}\mathbf{b}\|_F$; dies könnte ungenau sein, wenn $|q|$ zu klein, z.B. kleiner als $\|\mathbf{y}\|/10^6$, ist. Falls Sie eine bessere Näherung für $\|\mathbf{y} - \mathbf{X}\mathbf{b}\|_F$ brauchen, können Sie sie direkt aus $\mathbf{X}, \mathbf{y}$ und der berechneten Lösung **b** bestimmen.

Bei der gewichteten Methode der kleinsten Quadrate ersetzen Sie **X** und **y** durch **Wx** und **Wy**, wobei **W** die Diagonalmatrix mit den Gewichtungsfaktoren ist.

Bei der kleinsten Quadrate Aufgabe mit linearer Nebenbedingung müssen Sie berücksichtigen, daß Nebenbedingungen nicht immer miteinander verträglich sind. Zusätzlich können sie wegen Rundungsfehlern nicht immer genau erfüllt werden. Daher müssen Sie eine Toleranz t vorgeben, so daß für $\|\mathbf{C}\mathbf{b} - \mathbf{d}\| < t$ die Nebenbedingungen erfüllt sein sollen. Gewiß gilt bei 10stelliger Rechengenauigkeit $t > \|\mathbf{d}\|/10^{10}$, und in manchen Fällen muß eine viel größere Toleranz verwendet werden.

Nachdem Sie t gewählt haben, wählen Sie einen Gewichtungsfaktor w , der $w > \|\mathbf{y}\|/t$ erfüllt. Wählen Sie der Einfachheit halber w als eine Zehnerpotenz, die etwas größer als $\|\mathbf{y}\|/t$ ist. Dann gilt $w\|\mathbf{C}\mathbf{b} - \mathbf{d}\| > \|\mathbf{y}\|$, es sei denn $\|\mathbf{C}\mathbf{b} - \mathbf{d}\| < t$.

Die Nebenbedingung könnte jedoch immer noch aus einem der zwei folgenden Gründe nicht erfüllt sein:

- Es existiert kein **b** mit $\|\mathbf{C}\mathbf{b} - \mathbf{d}\| < t$.
- Die führenden Spalten von **C** sind beinahe linear abhängig.

Um zu kontrollieren, ob der erste Fall eingetreten ist, überprüfen Sie, ob es für die Nebenbedingungen allein eine Lösung gibt. Nachdem Sie $[\mathbf{wC} \ \mathbf{wd}]$ zu $\mathbf{Q}[\mathbf{U} \ \mathbf{g}]$ faktorisiert haben, berechnen Sie **b** aus dem folgenden Gleichungssystem

$$\begin{array}{l} (k \text{ Zeilen}) \\ (p+1-k \text{ Zeilen}) \end{array} \begin{bmatrix} \mathbf{U} & \mathbf{g} \\ \mathbf{0} & \text{diag}(q) \end{bmatrix} \begin{bmatrix} \mathbf{b} \\ -1 \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ -q \end{bmatrix} \begin{array}{l} (p \text{ Zeilen}) \\ (1 \text{ Zeile}) \end{array}$$

wo q irgendeine kleine Zahl ungleich null ist. Wenn für die berechnete Lösung **b** $\mathbf{Cb} \approx \mathbf{d}$ gilt, sind die Nebenbedingungen miteinander verträglich.

Der zweite Fall kommt selten vor und kann vermieden werden. In diesem Fall ist mindestens eines der Elemente in der Diagonalen von $\mathbf{U}$ viel kleiner als das größte darüberliegende Element derselben Spalte. $\mathbf{U}$ ist die bei der Orthogonal-Faktorisierung $\mathbf{wC} = \mathbf{QU}$ erzeugte Matrix.

Um diesen Fall zu vermeiden, ordnen Sie die Spalten von $\mathbf{wC}$ und $\mathbf{X}$ und ebenso die Elemente (Zeilen) von $\mathbf{b}$ um. Wie umgeordnet werden soll, kann leicht bestimmt werden, wenn das störende Diagonalelement von $\mathbf{U}$ auch viel kleiner als irgendein Element weiter hinten in derselben Zeile ist. Vertauschen Sie einfach die entsprechenden Spalten in Ihren ursprünglichen Daten und faktorisieren Sie die gewichteten Gleichungen mit den Nebenbedingungen. Wiederholen Sie nötigenfalls diesen Vorgang.

Wenn beispielsweise die Faktorisierung von $\mathbf{wC}$

$$\mathbf{U} = \begin{bmatrix} 1.0 & 2.0 & 0.5 & -1.5 & 0.3 \\ 0 & 0.02 & 0.5 & 3.0 & 0.1 \\ 0 & 0 & 2.5 & 1.5 & -1.2 \end{bmatrix},$$

ergibt, dann ist das zweite Diagonalelement viel kleiner als das Element 2.0 darüber. Dies zeigt, daß die erste und zweite Spalte der ursprünglichen Nebenbedingungen beinahe linear abhängig sind. Außerdem ist das Diagonalelement viel kleiner als das weiter hinten stehende Element derselben Zeile mit dem Wert 3.0. Daher sollten in den ursprünglichen Daten die zweite und vierte Zeile vertauscht und die Faktorisierung wiederholt werden.

Es ist immer empfehlenswert, die Verträglichkeit der Nebenbedingungen zu überprüfen. Bei der Gelegenheit können Sie auch die kleinen Elemente in der Diagonalen von $\mathbf{U}$ überprüfen.

Addieren Sie schließlich unter Verwendung von $\mathbf{U}$ und $\mathbf{g}$ als k erste Zeilen zu $\mathbf{X}$ und $\mathbf{y}$ korrespondierende Zeilen. (Mehr hierzu finden Sie unter *Kleinste Quadrate* Lösungen mit aufeinanderfolgenden Zeilen auf Seite 140.) Nun können Sie die kleinste Quadrate Aufgabe ohne Nebenbedingungen lösen, wobei jetzt

$$\mathbf{X} \rightarrow \begin{bmatrix} \mathbf{wC} \\ \mathbf{X} \end{bmatrix} \quad \text{und} \quad \mathbf{y} \rightarrow \begin{bmatrix} \mathbf{wd} \\ \mathbf{y} \end{bmatrix}.$$

Solange die berechnete Lösung die Bedingung $\|\mathbf{Cb} - \mathbf{d}\| < t$ erfüllt, minimiert sie auch $\|\mathbf{y} - \mathbf{Xb}\|$ mit der Nebenbedingung $\mathbf{Cb} \approx \mathbf{d}$.

Singuläre und fastsinguläre Matrizen

Eine Matrix ist genau dann singulär, wenn ihre Determinante gleich null ist. Die Determinante einer Matrix berechnet sich zu $(-1)^r$ mal dem Produkt der Diagonalelemente von $\mathbf{R}$, wobei $\mathbf{R}$ die obere Diagonalmatrix der LR -Zerlegung der Matrix und r die Anzahl der Zeilenvertauschungen während der Zerlegung ist. Somit ist eine Matrix theoretisch dann singulär, wenn wenigstens eines der Diagonalelemente von $\mathbf{R}$ (der Pivotelemente) null ist, ansonsten ist sie nichtsingulär.

Da der HP-15C nur mit einer endlichen Anzahl von Stellen rechnet, können Sie auf diese Weise nicht immer zwischen singulären und fastsingulären Matrizen unterscheiden. Betrachten wir beispielsweise die singuläre Matrix

$$\mathbf{B} = \begin{bmatrix} 3 & 3 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{1}{3} & 1 \end{bmatrix} \begin{bmatrix} 3 & 3 \\ 0 & 0 \end{bmatrix} = \mathbf{LR}$$

Mit 10stelliger Rechengenauigkeit wird diese Matrix in die nichtsinguläre Matrix

$$\mathbf{LR} = \begin{bmatrix} 1 & 0 \\ .3333333333 & 1 \end{bmatrix} \begin{bmatrix} 3 & 3 \\ 0 & 10^{-10} \end{bmatrix},$$

zerlegt. Die singuläre Matrix $\mathbf{B}$ kann von der nichtsingulären Matrix

$$\mathbf{D} = \begin{bmatrix} 3 & 3 \\ .9999999999 & 1 \end{bmatrix}$$

nicht unterschieden werden, da für beide dieselbe LR -Zerlegung berechnet wird. Andererseits ist die Matrix

$$\mathbf{A} = \begin{bmatrix} 3 & 3 \\ 1 & .9999999999 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{1}{3} & 1 \end{bmatrix} \begin{bmatrix} 3 & 3 \\ 0 & -10^{-10} \end{bmatrix} = \mathbf{LR}$$

nichtsingulär. Mit 10stelliger Rechengenauigkeit wird $\mathbf{A}$ in

$$\mathbf{LR} = \begin{bmatrix} 1 & 0 \\ .3333333333 & 1 \end{bmatrix} \begin{bmatrix} 3 & 3 \\ 0 & 0 \end{bmatrix}.$$

zerlegt. Diese Zerlegung würde fälschlicherweise anzeigen, daß $\mathbf{A}$ singulär sei. Wiederum kann die nichtsinguläre Matrix $\mathbf{A}$ nicht von der singulären Matrix

$$\mathbf{C} = \begin{bmatrix} 3 & 3 \\ .9999999999 & .9999999999 \end{bmatrix}$$

unterschieden werden, da für beide dieselbe LR -Zerlegung berechnet wird.

Wenn Sie mit Ihrem HP-15C eine inverse Matrix berechnen oder ein Gleichungssystem lösen wollen, sollten Sie beachten, daß er für manche singuläre und nichtsinguläre Matrizen dieselbe LR -Zerlegung berechnet. Aus diesem Grund stellt der HP-15C *bei jeder Rechnung* sicher, daß *keines* der Pivotelemente irgendeiner zerlegten Matrix null ist. Er tut dies, indem er nötigenfalls die Pivotelemente um einen Betrag, der gewöhnlich kleiner als der Rundungsfehler ist, abändert. Daher können Sie Matrizen invertieren und Gleichungssysteme lösen, ohne von Pivotelementen gleich null gestört zu werden. Dies ist in den Anwendungen von großer Bedeutung, wie z.B. der Berechnung von Eigenvektoren mit der Methode der inversen Iteration (siehe Seite 155).

Rundungsfehler und gegebenenfalls absichtliche Störungen bewirken, daß alle Pivotelemente der berechneten Zerlegung von null verschieden sind und daß $\mathbf{A}$ die gleiche Zerlegung wie eine nichtsinguläre Matrix $\mathbf{A} + \Delta\mathbf{A}$ hat, die mit dem ursprünglichen $\mathbf{A}$ identisch ist oder sich von ihm nur unwesentlich unterscheidet. Insbesondere ist die Spaltensummennorm $\|\Delta\mathbf{A}\|_C$ (bis auf 10 Stellen) gegen $\|\mathbf{A}\|_C$ vernachlässigbar, es sei dann, in irgendeiner Spalte von $\mathbf{A}$ haben alle Elemente einen kleineren Absolutbetrag als 10^{-89} .

Ihr HP-15C berechnet die Determinante einer quadratischen Matrix als das Produkt der (gegebenenfalls abgeänderten) vorher berechneten Pivotelemente, unter Erhaltung des Vorzeichens. Die berechnete Determinante ist also die Determinante der Matrix $\mathbf{A} + \Delta\mathbf{A}$, in ihrer LR -Zerlegung. Sie kann nur dann null werden, wenn der Betrag dieses Produkts kleiner als 10^{-99} wird (Underflow).

Anwendungen

Die folgenden Programmbeispiele zeigen, wie Sie mit Hilfe der Matrixoperationen anspruchsvolle Aufgabenstellungen lösen können.

Konstruktion einer Einheitsmatrix

Dieses Programm erzeugt eine Einheitsmatrix I_n in der Matrix, deren Deskriptor sich im Indexregister befindet. Das Programm geht davon aus, daß diese bereits auf $n \times n$ dimensioniert ist. Führen Sie das Programm aus, indem Sie **[GSB]** 8 drücken. Die Diagonalelemente der Ergebnismatrix sind 1; alle übrigen Elemente Null.

Tastenfolge

[g] **[P/R]**
[f] **[CLEAR]** **[PRGM]**
[f] **[LBL]** 8
[f] **[MATRIX]** 1
[f] **[LBL]** 9
[RCL] 0
[RCL] 1
[g] **[TEST]** 6
[g] **[CLx]**
[g] **[TEST]** 5
[EEX]

[f] **[USER]** **[STO]** (i)
[f] **[USER]**

[GTO] 9
[g] **[RTN]**
[g] **[P/R]**

Anzeige

000–
 001– 42,21, 8
 002– 42,16, 1
 003– 42,21, 9
 004– 45 0
 005– 45 1
 006– 43,30, 6
 007– 43 35
 008– 43,30, 5
 009– 26
 010u 44 24
 011– 22 9
 012– 43 32

Programm-Modus.

Setzt $i = j = 1$.

Prüft, ob $i \neq j$.

Prüft, ob $i = j$.

Falls $i = j$, wird das Element gleich 1 gesetzt.

Beim letzten Element wird der folgende Schritt übersprungen.

Run-Modus.

Verwendete Labels: 8 und 9.

Verwendete Register: R_0 , R_1 und das Indexregister.

Einmalige Nachiteration

Das folgende Programm berechnet $\mathbf{X}$ aus dem Gleichungssystem $\mathbf{AX} = \mathbf{B}$ und führt dann eine Nachiteration durch, um die Lösung zu verbessern. Das Programm verwendet vier Matrizen:

Matrix	A	B	C	D
Eingabe	System Matrix	Rechte Seite		
Ausgabe	System Matrix	Verbesserte Lösung	Unverbesserte Lösung	<i>LR</i> -Zerlegung von A

Tastenfolge**Anzeige****[g]** **[P/R]**

Programm-Modus.

[f] **CLEAR** **[PRGM]**

000–

[f] **[LBL]** **A**

001–42,21,11

[RCL] **[MATRIX]** **[A]**

002–45,16,11

[STO] **[MATRIX]** **[D]**

003–44,16,14

Speichert die Matrix des Gleichungssystems in **D**.**[RCL]** **[MATRIX]** **[B]**

004–45,16,12

[RCL] **[MATRIX]** **[D]**

005–45,16,14

[f] **[RESULT]** **[C]**

006–42,26,13

[÷]

007– 10

Berechnet erste Lösung **C**.**[f]** **[RESULT]** **[B]**

008–42,26,12

[f] **[MATRIX]** **6**

009–42,16, 6

Berechnet das Residuum **B**.**[RCL]** **[MATRIX]** **[D]**

010–45,16,14

[÷]

011– 10

Berechnet die Korrektur **B**.**[RCL]** **[MATRIX]** **[C]**

012–45,16,13

[+]

013– 40

Berechnet die nachiterierte Lösung **B**.**[g]** **[RTN]**

014– 43 32

[g] **[P/R]**

Run-Modus.

Verwendete Labels: **A**.Verwendete Matrizen: **A**, **B**, **C** und **D**.

Um dieses Programm auszuführen, müssen Sie:

1. Die Matrix **A** entsprechend der Matrix des Gleichungssystems dimensionieren und deren Elemente in **A** speichern.
2. Die Matrix **B** entsprechend der Matrix auf der rechten Seite der Gleichung dimensionieren und deren Elemente in **B** speichern.
3. **[GSB]** **[A]** drücken, um die verbesserte Lösung in Matrix **B** zu berechnen.

Beispiel: Berechnen Sie mit dem Nachiterationsprogramm die Inverse der Matrix **A** mit

$$\mathbf{A} = \begin{bmatrix} 33 & 16 & 72 \\ -24 & -10 & -57 \\ -8 & -4 & -17 \end{bmatrix}.$$

Die theoretische Inverse von **A** ist

$$\mathbf{A}^{-1} = \begin{bmatrix} -29/3 & -8/3 & -32 \\ 8 & 5/2 & 51/2 \\ 8/3 & 2/3 & 9 \end{bmatrix}.$$

Bestimmen Sie die Inverse, indem Sie aus $\mathbf{A}\mathbf{X} = \mathbf{B}$ $\mathbf{X}$ berechnen, wo **B** eine 3×3 Einheitsmatrix ist.

Geben Sie zunächst das vorige Programm ein. Geben Sie dann im Run-Modus die Elemente der Matrix **A** (des Gleichungssystems) und der Matrix **B** (der rechten Seite, hier die Einheitsmatrix) ein. Drücken Sie **[GSB]** **[A]**, um das Programm laufen zu lassen.

Rufen Sie die Elemente der ersten Lösung, der Matrix **C**, zurück.

$$\mathbf{C} = \begin{bmatrix} -9.666666881 & -2.666666726 & -32.00000071 \\ 8.000000167 & 2.500000046 & 25.50000055 \\ 2.666666728 & 0.6666666836 & 9.000000203 \end{bmatrix}.$$

Diese Lösung ist auf sieben Stellen genau. Die Genauigkeit ist damit weit innerhalb der auf Seite 103 angegebenen Grenzen:

$$(\text{Anzahl von genauen Stellen}) \geq 9 - \log(\|\mathbf{A}\| \|\mathbf{C}\|) - \log(3) \approx 4.8.$$

Rufen Sie die Elemente der verbesserten Lösung, Matrix **B**, zurück:

$$\mathbf{B} = \begin{bmatrix} -9.666666667 & -2.666666667 & -32.00000000 \\ 8.000000000 & 2.500000000 & 25.50000000 \\ 2.666666667 & 0.666666667 & 9.000000000 \end{bmatrix}.$$

In diesem Fall ergibt eine Nachiteration 10 korrekte Stellen.

Lösen eines nichtlinearen Gleichungssystems

Betrachten wir ein System p nichtlinearer Gleichungen mit p Unbekannten:

$$f_i(x_1, x_2, \dots, x_p) = 0 \quad \text{mit } i = 1, 2, \dots, p$$

für das die Lösung $x_1, x_2, \dots, x_p$ gesucht ist.

Sei

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{bmatrix}, \mathbf{f}(\mathbf{x}) = \begin{bmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ \vdots \\ f_p(\mathbf{x}) \end{bmatrix}, \text{ und } \mathbf{F}(\mathbf{x}) = \begin{bmatrix} F_{11}(\mathbf{x}) \dots F_{1p}(\mathbf{x}) \\ F_{21}(\mathbf{x}) \dots F_{2p}(\mathbf{x}) \\ \vdots \quad \quad \quad \vdots \\ F_{p1}(\mathbf{x}) \dots F_{pp}(\mathbf{x}) \end{bmatrix},$$

wo

$$F_{ij}(\mathbf{x}) = \frac{\partial}{\partial x_j} f_i(\mathbf{x}) \quad \text{mit } i, j = 1, 2, \dots, p.$$

ist. Das obige Gleichungssystem kann als $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ ausgedrückt werden. Das Newton'sche Lösungsverfahren beginnt mit einer geschätzten Anfangsnäherung $\mathbf{x}^{(0)}$ für eine Nullstelle $\mathbf{x}$ und berechnet dann

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - (\mathbf{F}(\mathbf{x}^{(k)}))^{-1} \mathbf{f}(\mathbf{x}^{(k)}) \quad \text{mit } k = 0, 1, 2, \dots,$$

bis die Folge der Näherungen $\mathbf{x}^{(k+1)}$ konvergiert.

Das Programm im folgenden Beispiel führt eine Iteration des Newtonverfahrens durch. Es wird

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \mathbf{d}^{(k)}$$

berechnet, wo $\mathbf{d}^{(k)}$ die Lösung des $(p \times p)$ linearen Gleichungssystems

$$\mathbf{F}(\mathbf{x}^{(k)}) \mathbf{d}^{(k)} = \mathbf{f}(\mathbf{x}^{(k)})$$

ist. Das Programm zeigt am Ende jeder Iteration die euklidischen Normen von $\mathbf{f}(\mathbf{x}^{(k)})$ und die Verbesserung $\mathbf{d}^{(k)}$ an.

Beispiel: Gegeben sei eine normalverteilte Zufallsvariable y mit unbekanntem Mittelwert m und unbekannter Varianz v^2 . Konstruieren Sie einen erwartungstreuen Test für die Hypothese $v^2 = v_0^2$ (gegen die Antithese $v^2 \neq v_0^2$), wo v_0^2 ein beliebig gegebener Wert ist.

Für eine beliebige Stichprobe y , die aus den Werten $y_1, y_2, \dots, y_n$ besteht, widerlegt ein erwartungstreuer Test die Hypothese, falls

$$s_n < x_1 v_0^2 \quad \text{oder} \quad s_n > x_2 v_0^2$$

wo

$$s_n = \sum_{i=1}^n (y_i - \bar{y})^2 \quad \text{und} \quad \bar{y} = \frac{1}{n} \sum_{i=1}^n y_i, \quad ,$$

mit Konstanten x_1 und x_2 .

Sei a die Signifikanz des Tests ($0 < a < 1$). Sie können x_1 und x_2 dann finden, indem Sie das Gleichungssystem $f_1(\mathbf{x}) = f_2(\mathbf{x}) = 0$ lösen, mit

$$f_1(\mathbf{x}) = (n-1) \ln(x_2/x_1) + x_1 - x_2$$

$$f_2(\mathbf{x}) = \int_{x_1}^{x_2} (w/2)^m \exp(-w/2) dw - 2(1-a)\Gamma(m+1).$$

Hier gilt $x_2 > x_1 > 0$, a und n sind gegeben ($n > 1$), und $m = (n-1)/2 - 1$. Ein Anfangswert für (x_1, x_2) ist

$$x_1^{(0)} = \chi_{n-1, a/2}^2 \quad \text{und} \quad x_2^{(0)} = \chi_{n-1, 1-a/2}^2$$

wo $\chi_{d,p}^2$ der p -te Teil der Chi-Quadrat Verteilung mit d Freiheitsgraden ist.

In diesem Beispiel ist

$$\mathbf{F}(\mathbf{x}) = \begin{bmatrix} 1 - (n-1)/x_1 & (n-1)/x_2 - 1 \\ -(x_1/2)^m \exp(-x_1/2) & (x_2/2)^m \exp(-x_2/2) \end{bmatrix}.$$

Geben Sie folgendes Programm ein:

Tastenfolge

Anzeige

g P/R
f CLEAR PRGM
f LBL A
2
ENTER
f DIM C

1
f DIM B

000-
001-42,21,11
002- 2
003- 36
004-42,23,13

005- 1
006-42,23,12

Programm-Modus.

Dimensioniert die Matrix F
2 × 2.

Dimensioniert die Matrix f
2 × 1.

Tastenfolge

GSB **B**
RCL **MATRIX** **A**
RCL **MATRIX** **B**
RCL **MATRIX** **C**
f **RESULT** **D**
 $\div$
f **RESULT** **A**
 $-$
g **LSTx**
f **MATRIX** **8**
RCL **MATRIX** **B**
f **MATRIX** **8**
g **RTN**
f **LBL** **B**

f **MATRIX** **1**
f **USER** **RCL** **A**
f **USER**
STO **4**
f **USER** **RCL** **A**
f **USER**
STO **5**
STO **5**
 $-$
RCL **5**
RCL $\div$ **4**
g **LN**
RCL **2**
1
 $-$
 $\times$
 $+$
STO **B**
1

Anzeige

007- **32 12**
008- **45,16,11**
009- **45,16,12**
010- **45,16,13**
011- **42,26,14**
012- **10**
013- **42,26,11**
014- **30**
015- **43 36**
016- **42,16, 8**
017- **45,16,12**
018- **42,16, 8**
019- **43 32**
020- **42,21,12**

021- **42,16,1**
022u **45 11**

023- **44 4**
024u **45 11**

025- **44 5**
026- **44 5**
027- **30**
028- **45 5**
029- **45,10, 4**
030- **43 12**
031- **45 2**
032- **1**
033- **30**
034- **20**
035- **40**
036- **44 12**
037- **1**

Berechnet $\mathbf{f}$ und $\mathbf{F}$.Berechnet $\mathbf{d}^{(k)}$.Berechnet $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \mathbf{d}^{(k)}$.Berechnet $\|\mathbf{d}^{(k)}\|_F$.Berechnet $\|\mathbf{f}(\mathbf{x}^{(k)})\|_F$.Routine zur Berechnung von $\mathbf{f}$ und $\mathbf{F}$.Speichert $x_1^{(k)}$ in R_4 .

Überspringt beim letzten Element den nächsten Schritt.

Speichert $x_2^{(k)}$ in R_5 .Berechnet $x_1 - x_2$.Berechnet $\ln(x_2/x_1)$.Berechnet $(n-1) \ln(x_2/x_1)$.Berechnet f_1 .Speichert f_1 in $\mathbf{B}$.

Tastenfolge

[RCL] 2
 1
 [-]
 [RCL] ÷ 4
 [-]
 [f] [USER] [STO] [C]
 [f] [USER]
 [RCL] 2
 1
 [-]
 [RCL] ÷ 5
 1
 [-]
 [f] [USER] [STO] [C]
 [f] [USER]
 [RCL] 4
 [RCL] 5
 [f] [$\int_x^x$] [C]
 [RCL] 3
 1
 [-]
 2
 [x]
 [RCL] 2
 3
 [-]
 2
 ÷
 [f] [x!]
 [x]
 [+]
 [STO] [B]
 [RCL] 4
 [GSB] [C]
 [CHS]
 [f] [USER] [STO] [C]
 [f] [USER]

Anzeige

038- 45 2
 039- 1
 040- 30
 041-45,10, 4 Berechnet $(n-1)/x_1$.
 042- 30 Berechnet F_{11} .
 043u 44 13 Speichert F_{11} in C.
 044- 45 2
 045- 1
 046- 30
 047-45,10, 5 Berechnet $(n-1)/x_2$.
 048- 1
 049- 30 Berechnet F_{12} .
 050u 44 13 Speichert F_{12} in C.
 051- 45 4
 052- 45 5
 053-42,20,13 Berechnet das Integral.
 054- 45 3
 055- 1
 056- 30
 057- 2
 058- 20 Berechnet $2(a-1)$.
 059- 45 2
 060- 3
 061- 30
 062- 2
 063- 10 Berechnet m .
 064- 42 0 Berechnet $\Gamma(m+1)$.
 065- 20
 066- 40 Berechnet f_2 .
 067- 44 12 Speichert f_2 in B.
 068- 45 4
 069- 32 13
 070- 16 Berechnet F_{21} .
 071u 44 13 Speichert F_{21} in C.

Tastenfolge	Anzeige	
RCL 5	072– 45 5	
GSB C	073– 32 13	Berechnet F_{22} .
f USER STO C	074u 44 13	Speichert F_{22} in C .
f USER		
g RTN	075– 43 32	Überspringt diese Zeile.
g RTN	076– 43 32	
f LBL C	077–42,21,13	Routine zur Auswertung des Integranden.
2	078– 2	
$\div$	079– 10	
CHS	080– 16	
e^x	081– 12	Berechnet $e^{-x/2}$.
g LSTx	082– 43 36	
CHS	083– 16	
RCL 2	084– 45 2	
3	085– 3	
$-$	086– 30	
2	087– 2	
$\div$	088– 10	Berechnet m .
y^x	089– 14	
$\times$	090– 20	Berechnet $(x/2)^m e^{-x/2}$.
g RTN	091– 43 32	

Verwendete Labels: A, B und C.

Verwendete Register: R_0 (Zeile), R_1 (Spalte), R_2 (n), R_3 (a), R_4 ($x_1^{(k)}$) und R_5 ($x_2^{(k)}$).

Verwendete Matrizen: $\mathbf{A}(\mathbf{x}^{(k+1)})$, $\mathbf{B}(\mathbf{f}(\mathbf{x}^{(k)}))$, $\mathbf{C}(\mathbf{F}(\mathbf{x}^{(k)}))$ und $\mathbf{D}(\mathbf{d}^{(k)})$.

Lassen Sie nun das Programm laufen. Wählen Sie zum Beispiel die Werte $n = 11$ und $a = 0.05$. Die Anfangsnäherungen seien $x_1^{(0)} = 3.25$ und $x_2^{(0)} = 20.5$. Denken Sie daran, daß das Anzeigeformat die Genauigkeit der Berechnung des Integrals beeinflusst.

Tastenfolge	Anzeige	
g P/R		Run-Modus.
5 f DIM (i)	5.0000	Reserviert R_0 bis R_5 .
11 STO 2	11.0000	Speichert n in R_2 .

Tastenfolge	Anzeige	
.05 [STO] 3	0.0500	Speichert a in R_3 .
2 [ENTER] 1	1	
[f] [DIM] [A]	1.0000	Dimensioniert A auf 2×1 .
[f] [USER]	1.0000	Aktiviert den User-Modus.
[f] [MATRIX] 1	1.0000	
3.25 [STO] [A]	3.2500	Speichert $x_1^{(0)}$ der Chi-Quadrat Verteilung.
20.5 [STO] [A]	20.5000	Speichert $x_2^{(0)}$ der Chi-Quadrat Verteilung.
[f] [SCI] 4	2.0500 01	Wählt das Anzeigeformat.
[A]	1.1677 00	Zeigt die Norm von $f(x^{(0)})$ an.
[R↕]	1.0980 00	Zeigt die Norm der Verbesserung $d^{(0)}$ an.
[RCL] [A]	3.5519 00	Ruft $x_1^{(1)}$ zurück.
[RCL] [A]	2.1556 01	Ruft $x_2^{(1)}$ zurück.

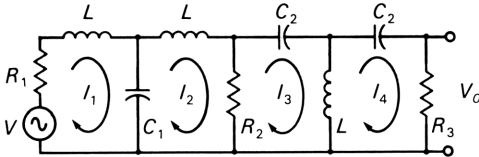
Wenn Sie die letzten vier Schritte wiederholen, erhalten Sie die folgenden Ergebnisse:

k	$\ f(x^{(k)})\ _F$	$\ d^{(k)}\ _F$	$x_1^{(k+1)}$	$x_2^{(k+1)}$
0	1.168	1.098	3.5519	21.556
1	1.105×10^{-1}	1.740×10^{-1}	3.5169	21.726
2	1.918×10^{-3}	2.853×10^{-3}	3.5162	21.729
3	6.021×10^{-7}	9.542×10^{-7}	3.5162	21.729

Für den statistischen Test ist diese Genauigkeit hinreichend. (Drücken Sie **[f]** **[FIX]** 4, um das Anzeigeformat zurückzustellen, und **[f]** **[USER]**, um den User-Modus zu deaktivieren.)

Lösen eines großen komplexen Gleichungssystems

Beispiel: Für das unten gezeigte Filternetzwerk ist die Ausgangsspannung bei einer Kreisfrequenz $\omega = 15 \times 10^3$ rad/s gesucht.



$V = 10$ Volt
 $R_1 = 100$ Ohm
 $R_2 = 10^6$ Ohm
 $R_3 = 10^5$ Ohm

$L = 10^{-2}$ Henry
 $C_1 = 25 \times 10^{-8}$ Farad
 $C_2 = 25 \times 10^{-6}$ Farad

Beschreiben Sie das Netzwerk mit den Maschenströmen:

$$\begin{bmatrix} (R_1 + i\omega L - i/\omega C_1) & (i/\omega C_1) & 0 & 0 \\ (i/\omega C_1) & (R_2 + i\omega L - i/\omega C_1) & (-R_2) & 0 \\ 0 & (-R_2) & (R_2 - i/\omega C_2 + i\omega L) & (-i\omega L) \\ 0 & 0 & (-i\omega L) & (R_3 + i\omega L - i/\omega C_2) \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \\ I_3 \\ I_4 \end{bmatrix} = \begin{bmatrix} V \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Bestimmen Sie aus diesem komplexen Gleichungssystem I_1, I_2, I_3 und I_4 . Dann gilt $V_O = (R_3)(I_4)$.

Dieses Gleichungssystem ist zu groß, um mit dem Standardverfahren zur Lösung eines komplexen Gleichungssystems gelöst zu werden. Daher verwenden wir hierfür ein alternatives Verfahren (das im Benutzerhandbuch näher beschrieben wird). Geben Sie hierzu in Matrix A die Matrix des Gleichungssystems ein und berechnen Sie ihre Inverse. Beachten Sie, daß $\omega L = 150$, $1/\omega C_1 = 800/3$ und $1/\omega C_2 = 8/3$ gilt.

Tastenfolge

CLEAR

Anzeige

000–

Programm-Modus.
 Löscht den Programmspeicher.

Tastenfolge	Anzeige	
g P/R		Run-Modus.
0 f DIM (i)	0.0000	Stellt auf maximalen Matrix-speicher.
f MATRIX 0	0.0000	Dimensioniert alle Matrizen 0×0 .
4 ENTER 8	8	
f DIM A	8.0000	Dimensioniert Matrix A 4×8 .
f MATRIX 1	8.0000	
f USER	8.0000	Aktiviert User-Modus.
100 STO A	100.0000	Speichert $\text{Re}(a_{11})$.
150 ENTER	150.0000	
800 ENTER 3 ÷	266.6667	
- STO A	-116.6667	Speichert $\text{Im}(a_{11})$.
⋮		
150 ENTER	150.0000	
8 ENTER 3 ÷	2.6667	
- STO A	147.3333	Speichert $\text{Im}(a_{44})$.
RCL MATRIX A	A 4 8	
f P_{y,x}	A 8 4	Transformiert $\mathbf{A}^C$ in $\mathbf{A}^P$.
f MATRIX 2	A 8 8	Transformiert $\mathbf{A}^P$ in $\tilde{\mathbf{A}}$.
STO RESULT	A 8 8	
f 1/x	A 8 8	Berechnet Inverse von $\tilde{\mathbf{A}}$ in A .

Nun können Sie die zweite Hälfte der Zeilen von **A** löschen, um den zum Speichern der Matrix **B** benötigten Platz zu schaffen.

Tastenfolge	Anzeige	
4 ENTER 8	8	
f DIM A	8.0000	Dimensioniert die Matrix A wieder 4×8 .
4 ENTER 2	2	
f DIM B	2.0000	Dimensioniert Matrix B 4×2 .

Tastenfolge

f MATRIX 1

10 STO B

RCL MATRIX A

RCL MATRIX B

f P y,x

f MATRIX 2

f RESULT C

x

f MATRIX 4

f MATRIX 2

1 ENTER 8

f DIM C

RCL RESULT

f MATRIX 4

g C y,x

Anzeige

2.0000

10.0000

A 4 8

b 4 2

b 8 1

b 8 2

b 8 2

C 4 2

C 2 4

C 2 8

8

8.0000

C 1 8

C 8 1

C 4 2

Speichert $\text{Re}(I)$. (Die anderen Elemente sind 0.)Transformiert $\mathbf{B}^C$ in $\mathbf{B}^P$.Transformiert $\mathbf{B}^P$ in $\tilde{\mathbf{B}}$.

Berechnet Lösung in C.

Berechnet Transponierte.

Transformiert C in $\tilde{\mathbf{C}}$.Dimensioniert die Matrix C wieder auf 1×8 .

Berechnet Transponierte.

Transformiert $\mathbf{C}^P$ in $\mathbf{C}^C$.

Die Matrix C enthält die gesuchten Werte I_1 , I_2 , I_3 und I_4 in kartesischer Darstellung. Ihre Phasendarstellung kann einfach berechnet werden:

Tastenfolge

f MATRIX 1

f SCI 4

RCL C

RCL C

x $\div$ y g $\Rightarrow$ Px $\div$ y

RCL C

RCL C

x $\div$ y g $\Rightarrow$ Px $\div$ y

RCL C

RCL C

x $\div$ y g $\Rightarrow$ P

Anzeige

C 4 2

C 4 2

1.9950 -04

4.0964 -03

4.1013 -03

8.7212 01

-1.4489 -03

-3.5633 -02

3.5662 -02

-9.2328 01

-1.4541 -03

-3.5633 -02

3.5662 -02

Setzt R_0 und R_1 zurück.Ruft $\text{Re}(I_1)$ zurück.Ruft $\text{Im}(I_1)$ zurück.Zeigt $|I_1|$ an.Zeigt $\text{Arg}(I_1)$ in Grad an.Zeigt $|I_2|$ an.Zeigt $|I_3|$ an.

Tastenfolge**Anzeige**

$x \geq y$	-9.2337	01	
RCL C	5.3446	-05	
RCL C	-2.2599	-06	
$x \geq y$ g $\rightarrow P$	5.3494	-05	Zeigt $ I_4 $ an.
$x \geq y$	-2.4212	00	
$x \geq y$ EEX 5 $\times$	5.3494	00	Berechnet $ V_0 = (R_3) I_4 $.
f FIX 4	5.3494		
f USER	5.3494		Desaktiviert den User-Modus.

Das Ergebnis ist $5.3494 \angle -2.4212^\circ$.

Methode der kleinsten Quadrate mit Normalgleichungen

In der Statistikkultur wird die Methode der kleinsten Quadrate ohne Nebenbedingungen als *multiple lineare Regression* bezeichnet. Sie benutzt das lineare Modell

$$y = \sum_{j=1}^p b_j x_j + r.$$

wo $b_1, \dots, b_p$ die gesuchten Parameter, $x_1, \dots, x_p$ die unabhängigen Variablen sind, y die abhängige Variable und r der zufällige Fehler mit dem Erwartungswert $E(r) = 0$ und der Varianz σ^2 ist.

Nach n Messungen von y und $x_1, x_2, \dots, x_p$ kann diese Aufgabe als

$$\mathbf{y} = \mathbf{X}\mathbf{b} + \mathbf{r}$$

ausgedrückt werden, wo $\mathbf{y}$ ein Vektor der Dimension n , $\mathbf{X}$ eine Matrix der Dimension $n \times p$ und $\mathbf{r}$ ein Vektor der Dimension n mit den unbekannten zufälligen Fehlern, für die $E(\mathbf{r}) = 0$ und $\text{Cov}(\mathbf{r}) = E(\mathbf{r}\mathbf{r}^T) = \sigma^2 \mathbf{I}_n$ gilt, ist.

Wenn das Modell korrekt ist und $\mathbf{X}^T \mathbf{X}$ eine Inverse hat, dann hat die berechnete Kleinste Quadrate Lösung $\hat{\mathbf{b}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$ folgende Eigenschaften:

- $E(\hat{\mathbf{b}}) = \mathbf{b}$, so daß $\hat{\mathbf{b}}$ ein erwartungstreuer Schätzer für $\mathbf{b}$ ist.
- $\text{Cov}(\hat{\mathbf{b}}) = E((\hat{\mathbf{b}} - \mathbf{b})(\hat{\mathbf{b}} - \mathbf{b})^T) = \sigma^2 (\mathbf{X}^T \mathbf{X})^{-1}$, die Kovarianzmatrix des Schätzers $\hat{\mathbf{b}}$.

- $E(\hat{\mathbf{r}}) = \mathbf{0}$, wo $\hat{\mathbf{r}} = \mathbf{y} - \mathbf{X}\hat{\mathbf{b}}$ der Vektor des Residuums ist.
- $E(\|\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\|_F^2) = (n - p)\sigma^2$, so daß $\sigma^2 = \|\hat{\mathbf{r}}\|_F^2/(n - p)$ ein erwartungstreuer Schätzer für σ^2 ist. Sie können eine Abschätzung für $\text{Cov}(\hat{\mathbf{b}})$ finden, indem Sie σ^2 durch $\hat{\sigma}^2$ ersetzen.

Die gesamte Quadratsumme $\|\mathbf{y}\|_F^2$ kann folgendermaßen aufgespalten werden:

$$\begin{aligned}\|\mathbf{y}\|_F^2 &= \mathbf{y}^T \mathbf{y} \\ &= (\mathbf{y} - \mathbf{X}\hat{\mathbf{b}} + \mathbf{X}\hat{\mathbf{b}})^T (\mathbf{y} - \mathbf{X}\hat{\mathbf{b}} + \mathbf{X}\hat{\mathbf{b}}) \\ &= (\mathbf{y} - \mathbf{X}\hat{\mathbf{b}})^T (\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}) - 2\hat{\mathbf{b}}^T \mathbf{X}^T (\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}) + (\mathbf{X}\hat{\mathbf{b}})^T (\mathbf{X}\hat{\mathbf{b}}) \\ &= \|\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\|_F^2 + \|\mathbf{X}\hat{\mathbf{b}}\|_F^2 \\ &= \left(\begin{array}{c} \text{Residuen-} \\ \text{quadratsumme} \end{array} \right) + \left(\begin{array}{c} \text{Regressions-} \\ \text{quadratsumme} \end{array} \right).\end{aligned}$$

Wenn das Modell korrekt ist, gilt

$$E(\|\mathbf{X}\hat{\mathbf{b}}\|_F^2/p) = \sigma^2 + \|\mathbf{X}\mathbf{b}\|_F^2/p > \sigma^2$$

und

$$E(\|\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\|_F^2/(n - p)) = \sigma^2$$

mit $\mathbf{b} \neq \mathbf{0}$. Wenn das vereinfachte Modell $y = r$ korrekt ist, sind diese beiden Erwartungswerte gleich σ^2 .

Sie können die Hypothese, daß das vereinfachte Modell gilt (gegenüber der Alternative, daß das ursprüngliche Modell gilt), prüfen, indem Sie den F -Ratio

$$F = \frac{\|\mathbf{X}\hat{\mathbf{b}}\|_F^2/p}{\|\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\|_F^2/(n - p)}.$$

berechnen. In dem Fall, daß das ursprüngliche Modell gilt ($b \neq 0$), ist F größer, als wenn das vereinfachte Modell gilt ($\mathbf{b} = \mathbf{0}$). Wenn F hinreichend groß ist, müssen Sie die Hypothese verwerfen.

Wenn das Zufallsglied normalverteilt ist, gehorcht der F -Ratio einer zentralen F Verteilung mit p und mit $(n - p)$ Freiheitsgraden, falls $\mathbf{b} = \mathbf{0}$, und einer nichtzentralen Verteilung, falls $\mathbf{b} \neq \mathbf{0}$. Ein statistischer Test soll die Hypothese verwerfen (wobei die Wahrscheinlichkeit, die Hypothese fälschlicherweise zu verwerfen, α ist), falls der F -Ratio größer als der 100α prozentige Anteil der zentralen F Verteilung mit p und $(n - p)$ Freiheitsgraden ist; andernfalls soll er die Hypothese bestätigen.

Das folgende Programm paßt das lineare Modell mit der Methode der kleinsten Quadrate an eine Menge von n Datenwerten $x_{i1}, x_{i2}, \dots, x_{ip}, y_i$ an. Die Parameter $b_1, b_2, \dots, b_p$ werden aus der Lösung $\hat{\mathbf{b}}$ des Normalgleichungssystems $\mathbf{X}^T \mathbf{X} \mathbf{b} = \mathbf{X}^T \mathbf{y}$ geschätzt. Das Programm schätzt auch σ^2 und die Parameterkovarianzmatrix $\text{Cov}(\hat{\mathbf{b}})$. Weiterhin berechnet es die Regressions- und Residuenquadratsummen (*Reg SS* und *Res SS*) und die Residuen.

Das Programm benötigt zwei Matrizen:

Matrix **A**: $n \times p$ mit Zeilen $i(x_{i1}, x_{i2}, \dots, x_{ip})$
und $i = 1, 2, \dots, n$.

Matrix **B**: $n \times 1$ mit dem Element $i(y_i)$ und $i = 1, 2, \dots, n$.

Das Programm gibt aus:

Matrix **A**: unverändert.

Matrix **B**: $n \times 1$ mit den Residuen aus der Anpassung.

$(y_i - \hat{b}_1 x_{i1} - \dots - \hat{b}_p x_{ip})$ mit $i = 1, 2, \dots, n$, wo $\hat{b}_i$ die Schätzung für b_i ist.

Matrix **C**: $p \times p$ Kovarianzmatrix der Schätzungen für die Parameter.

Matrix **D**: $p \times 1$ mit den Schätzungen für die Parameter $\hat{b}_1, \dots, \hat{b}_p$.

T-Register: enthält eine Schätzung für σ^2 .

Y-Register: enthält die Regressionsquadratsumme (*Reg SS*).

X-Register: enthält die Residuenquadratsumme (*Res SS*).

Die folgende Varianzanalyse-Tabelle (ANOVA) teilt die totale Quadratsumme (*Tot SS*) in die Regressions- und Residuenquadratsummen auf. Mit Hilfe dieser Tabelle können Sie den *F*-Ratio berechnen.

ANOVA

Quelle	Freiheits- grade	Quadrat- summe	Quadrat. Mittel	F-Ratio
Regression	p	<i>Reg SS</i>	$\frac{(\text{Reg SS})}{p}$	$\frac{(\text{Reg MS})}{(\text{Res MS})}$
Residuum	$n - p$	<i>Res SS</i>	$\frac{(\text{Res SS})}{(n - p)}$	
Summe	n	<i>Tot SS</i>		

Die berechnete Regressionsquadratsumme ist *nicht* an das Mittel *angepaßt*, da das Modell vielleicht keinen konstanten Term enthält. Um einen konstanten Term mitzuberücksichtigen, schließen Sie in das Modell eine Variable identisch gleich eins ein. Der dazugehörige Parameter ist dann der konstante Term.

Um eine *an das Mittel angepasste* Regressionsquadratsumme für ein Modell, das einen konstanten Term enthält, zu berechnen, benutzen Sie zunächst das Programm, das das Modell anpaßt und die unangepaßte Regressionsquadratsumme findet. Passen Sie dann das einfachere Modell $y = b_1 + r$ dadurch an, daß Sie alle Variablen außer der einen, die gleich eins ist (z.B. b_1) weglassen und nun die Regressionsquadratsumme $(Reg\ SS)_C$ für dieses Modell finden. Die an das Mittel angepasste Regressionsquadratsumme $(Reg\ SS)_A = Reg\ SS - (Reg\ SS)_C$.

Damit wird die ANOVA-Tabelle zu:

ANOVA				
Quelle	F.G.	Q.S.	Q.M.	F
Regressions-Konstante	$p - 1$	$(Reg\ SS)$	$\frac{(Reg\ SS)_A}{(p - 1)}$	$\frac{(Reg\ MS)_A}{(Res\ MS)}$
Konstante	1	$(Reg\ SS)_C$	$(Reg\ SS)_C$	
Residuum	$n - p$	Res SS	$\frac{(Res\ SS)}{(n - p)}$	
Summe	n	Tot SS		

Jetzt können Sie mit dem F -Ratio prüfen, ob das allgemeinere Modell die Daten erheblich besser anpaßt als das einfachere Modell $y = b_1 + r$.

Angenommen, Sie wollen eine Reihe von Regressionen durchführen und zwischen den einzelnen Regressionen die unabhängigen Variablen wegfällen lassen. Um dies tun zu können, ordnen Sie die Variablen andersherum, als sie aus dem Modell entfernt werden sollen. Sie können dann entfernt werden, indem die Matrix **A** transponiert wird, **A** auf weniger Zeilen neu dimensioniert und dann **A** wieder zurücktransponiert wird.

Für jede einzelne Regression benötigen Sie die ursprünglichen abhängigen Variablen. Wenn in Matrix **E** nicht genügend Platz vorhanden ist, um die ursprünglichen Werte abzuspeichern, können Sie sie aus der Ausgabe der Regressionsanpassung berechnen. Es folgt ein Unterprogramm, das diese Berechnung durchführt.

Dieses Unterprogramm hat folgende Eigenschaften:

- Wenn das gesamte Programm in den Programmspeicher eingegeben wird, müssen die Werte von n und p die Bedingungen $n \geq p$ und $(n+p)(p+1) \leq 56$ erfüllen. Das heißt,

wenn p	1	2	3	4
dann ist $n_{\max}$	27	16	11	7

Es wird hierbei vorausgesetzt, daß nur die Datenspeicher R_0 und R_1 zugeordnet werden. Wenn das Unterprogramm «B» weggelassen wird, dann gilt $n \geq p$ und $(n+p)(p+1) \leq 58$. Das heißt,

wenn p	1	2	3	4
dann ist $n_{\max}$	28	17	11	7

- Obwohl das Unterprogramm «B» die Residuumsfunktion mit ihrer höheren Genauigkeit verwendet, stimmen die berechneten Werte der abhängigen Variablen nicht unbedingt genau mit den ursprünglichen Werten überein. Sie stimmen jedoch im allgemeinen für statistische Tests und Abschätzungen gut genug überein. Ist eine höhere Genauigkeit erforderlich, können die ursprünglichen Werte in Matrix **B** neu eingegeben werden.

Tastenfolge

g **P/R**
f **CLEAR** **PRGM**
f **LBL** **A**

RCL **MATRIX** **B**
f **MATRIX** **8**
g **x²**
RCL **MATRIX** **A**

ENTER
f **RESULT** **C**
f **MATRIX** **5**

g **LSTx**
RCL **MATRIX** **B**
f **RESULT** **D**
f **MATRIX** **5**

x $\hat{=}$ y

Anzeige

000–
001–**42,21,11**

002–**45,16,12**
003–**42,16, 8**
004– **43 11**

005–**45,16,11**

006– **36**
007–**42,26,13**
008–**42,16, 5**

009– **43 36**
010–**45,16,12**
011–**42,26,14**
012–**42,16, 5**

013– **34**

Programm-Modus.

Programm zur Anpassung des Modells.

Berechnet $Tot\ SS$.

Berechnet $C = A^T A$.

Berechnet $D = A^T B$.

Tastenfolge

Anzeige

$\div$	014– 10	Berechnet die Parameter in D .
RCL MATRIX A	015–45,16,11	
$x \geq y$	016– 34	
f RESULT B	017–42,26,12	
f MATRIX 6	018–42,16, 6	Berechnet die Residuen der Anpassung in B .
f MATRIX 8	019–42,16, 8	
g x^2	020– 43 11	Berechnet <i>Res SS</i> .
RCL DIM A	021–45,23,11	
-	022– 30	
$\div$	023– 10	Berechnet Abschätzung für σ^2 .
ENTER	024– 36	
ENTER	025– 36	
RCL MATRIX C	026–45,16,13	
f RESULT C	027–42,26,13	
$\div$	028– 10	Berechnet Kovarianzmatrix in C .
g R↑	029– 43 33	
RCL MATRIX B	030–45,16,12	
f MATRIX 8	031–42,16, 8	
g x^2	032– 43 11	
-	033– 30	Berechnet <i>Reg SS</i> .
g LSTx	034– 43 36	Ruft <i>Res SS</i> zurück.
g RTN	035– 43 32	
f LBL B	036–42,21,12	Unterprogramm, das Werte für abhängige Variablen wieder berechnet.
RCL MATRIX A	037–45,16,11	
RCL MATRIX D	038–45,16,14	
CHS	039– 16	
f RESULT B	040–42,26,12	
f MATRIX 6	041–42,16, 6	Berechnet B = B + AD .
RCL MATRIX D	042–45,16,14	
CHS	043– 16	
g RTN	044– 43 32	

Verwendete Labels: A und B.

Verwendete Register: R_0 und R_1 .

Verwendete Matrizen: **A**, **B**, **C** und **D**.

Um dieses Programm zu benutzen:

1. Drücken Sie 1 $\boxed{f}$ $\boxed{DIM}$ $\boxed{(i)}$, um die Register R_0 und R_1 zu reservieren.
2. Dimensionieren Sie Matrix **A** nach der Anzahl n der Meßwerte und der Anzahl p der Parameter, indem Sie n $\boxed{ENTER}$ p $\boxed{f}$ $\boxed{DIM}$ $\boxed{A}$ drücken.
3. Dimensionieren Sie Matrix **B** nach der Anzahl n der Meßwerte (und eine Spalte), indem Sie n $\boxed{ENTER}$ 1 $\boxed{f}$ $\boxed{DIM}$ $\boxed{B}$ drücken.
4. Drücken Sie $\boxed{f}$ $\boxed{MATRIX}$ 1 , um die Register R_0 und R_1 zu initialisieren.
5. Drücken Sie $\boxed{f}$ $\boxed{USER}$, um den User-Modus zu aktivieren.
6. Speichern Sie die Werte der p Variablen jeder Einzelmessung in einer Zeile der Matrix **A**. Wiederholen Sie dies für alle n Messungen.
7. Speichern Sie die Werte der abhängigen Variablen in Matrix **B**.
8. Drücken Sie $\boxed{A}$, um $Res\ SS$ zu berechnen und anzuzeigen. Im Y-Register befindet sich $Reg\ SS$ und im T-Register die Abschätzung für σ^2 .
9. Drücken Sie $\boxed{RCL}$ $\boxed{D}$, um jede der Schätzungen für die p Parameter anzuzeigen.
10. Drücken Sie nötigenfalls $\boxed{B}$, um die Werte der abhängigen Variablen in Matrix **B** neu zu berechnen.

Beispiel: Vergleichen Sie zwei Regressionsmodelle für den jährlichen Anstieg der Verbraucherpreisindex (VPI), die auf dem jährlichen Anstieg des Herstellerpreisindex (HPI) und der Arbeitslosenzahl (AZ) basieren.

$$y = b_1 + b_2x_2 + b_3x_3 + r \quad \text{und} \quad y = b_1 + b_2x_2 + r,$$

wo y , x_2 , und x_3 für VPI, HPI und AZ (in Prozent) stehen. Verwenden Sie die folgenden Daten aus den USA:

Jahr	VPI	HPI	AZ
1969	5.4	3.9	3.5
1970	5.9	3.7	4.9
1971	4.3	3.3	5.9
1972	3.3	4.5	5.6
1973	6.2	13.1	4.9
1974	11.0	18.9	5.6
1975	9.1	9.2	8.5
1976	5.8	4.6	7.7
1977	6.5	6.1	7.0
1978	7.6	7.8	6.0
1979	11.5	19.3	5.8

Tastenfolge

Anzeige

g P/R

Run-Modus.

f MATRIX 0

11 ENTER 3 f DIM A

3.0000

Dimensioniert **A** 11×3 .

11 ENTER 1 f DIM B

1.0000

Dimensioniert **B** 11×1 .

f MATRIX 1

1.0000

f USER

1.0000

1 STO A

1.0000

Gibt Werte der unabhängigen Variablen ein.

3.9 STO A

3.9000

3.5 STO A

3.5000

⋮

⋮

1 STO A

1.0000

19.3 STO A

19.3000

5.8 STO A

5.8000

5.4 STO B

5.4000

Gibt Werte der abhängigen Variablen ein.

5.9 STO B

5.9000

⋮

⋮

11.5 STO B

11.5000

A f FIX 9

13.51217504

Res SS für das gesamte Modell.

R $\downarrow$

587.9878252

Reg SS für das gesamte Modell.

R $\downarrow$ R $\downarrow$

1.689021880

Schätzung für σ^2 .

Tastenfolge

RCL D

RCL D

RCL D

B

RCL MATRIX A

f MATRIX 4

2 ENTER 11

f DIM A

RCL MATRIX A

f MATRIX 4

A

R↵

R↵ R↵

RCL D

RCL D

B

RCL MATRIX A

f MATRIX 4

1 ENTER 11

f DIM A

RCL MATRIX A

f MATRIX 4

A

R↵

R↵ R↵

RCL D

f USER

f FIX 4

Anzeige

1.245864326

0.379758235

0.413552218

d 3 1

A 11 3

A 3 11

11

11.00000000

A 2 11

A 11 2

16.78680552

584.7131947

1.865200613

3.701730745

0.380094935

d 2 1

A 11 2

A 2 11

11

11.00000000

A 1 11

A 11 1

68.08545454

533.4145457

6.808545454

6.963636364

6.963636364

6.9636

Schätzung für b_1 .Schätzung für b_2 .Schätzung für b_3 .

Berechnet die Werte der abhängigen Variablen neu.

Streicht die letzte Spalte von A.

Neue Matrix A.

 $Res\ SS$ für das reduzierte Modell. $Reg\ SS$ für das reduzierte Modell.Schätzung für σ^2 .Schätzung für b_1 .Schätzung für b_2 .

Berechnet die Werte der abhängigen Variablen neu.

Streicht die nächste Spalte von A.

Neue Matrix A.

 $Res\ SS$. $Reg\ SS$ für Konstante.Schätzung für σ^2 .Schätzung für b_1 .

Desaktiviert den User-Modus.

$Reg\ SS$ für die Variable HPI, angepaßt an den konstanten Wert, ergibt
 ($Reg\ SS$ für reduziertes Modell) – ($Reg\ SS$ für Konstante) = 51.29864900.

Reg SS für die Variable AZ, angepaßt an die Variable HPI und den konstanten Wert, gibt

$$(\text{Reg SS für gesamtes Modell}) - (\text{Reg SS für reduziertes Modell}) = 3.274630500.$$

Stellen Sie nun folgende ANOVA Tabelle auf:

Quelle	Freiheits-grade	Quadrat-summe	Quadratisches Mittel	F-Ratio
AZ HPI, Konstante	1	3.2746305	3.2746305	1.939
HPI Konstante	1	51.2986490	51.2986490	30.37
Konstante	1	533.4145457	533.4145457	315.8
Residuum (gesamtes Modell)	8	13.5121750	1.68902188	
Summe	11	601.5000002		

Der F -Ratio der Arbeitslosenzahl, an den Anstieg des Herstellerpreisindex und die Konstante angepaßt, ist für 10 Prozent Signifikanz ($\alpha = 0.1$) statistisch nicht signifikant. Daher verbessert ein Einbezug der Arbeitslosenzahl in das Modell die VPI-Anpassung nicht wesentlich.

Der F -Ratio des Herstellerpreisindex, an die Konstante angepaßt, ist jedoch für 0.1 Prozent Signifikanz ($\alpha = 0.001$) signifikant. Daher verbessert ein Einbezug des Herstellerpreisindex in das Modell die VPI-Anpassung.

Methode der kleinsten Quadrate mit sukzessiven Zeilen

Dieses Programm verwendet orthogonale Faktorisierung, um kleinste Quadrate Aufgaben zu lösen. Das heißt es findet die Parameter $b_1, \dots, b_p$, die die Quadratsumme $\|\mathbf{r}\|_F^2 = (\mathbf{y} - \mathbf{X}\mathbf{b})^T(\mathbf{y} - \mathbf{X}\mathbf{b})$ für das Modell

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \dots & x_{np} \end{bmatrix} \quad \text{und} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}.$$

minimiert. Das Programm führt dies für sukzessive Werte von n durch, doch die Lösung $\mathbf{b} = \mathbf{b}^{(n)}$ ist nur für $n \geq p$ sinnvoll.

Die erweiterte $n \times (p+1)$ Matrix $[\mathbf{X} \ \mathbf{y}]$ kann in $\mathbf{Q}^T \mathbf{V}$ faktorisiert werden, wo $\mathbf{Q}$ eine orthogonale Matrix,

$$\mathbf{V} = \begin{bmatrix} \hat{\mathbf{U}} & \mathbf{g} \\ \mathbf{0} & q \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \quad \begin{array}{l} (p \text{ Zeilen}) \\ (1 \text{ Zeile}) \\ (n-p-1 \text{ Zeilen}) \end{array}$$

und $\hat{\mathbf{U}}$ eine obere Dreiecksmatrix ist. Wenn diese Faktorisierung durch Einschließen von n Zeilen $\mathbf{r}_m = (x_{m1}, x_{m2}, \dots, x_{mp}, y_m)$ mit $m = 1, 2, \dots, n$ in $[\mathbf{X} \ \mathbf{y}]$ erreicht wurde, wird auf $n+1$ Zeilen erweitert, indem die Zeile $\mathbf{r}_{n+1}$ in $[\mathbf{X} \ \mathbf{y}]$ hinzugefügt wird:

$$\begin{bmatrix} \mathbf{X} & \mathbf{y} \\ \mathbf{r}_{n+1} \end{bmatrix} = \begin{bmatrix} \mathbf{Q}^T & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{V} \\ \mathbf{r}_{n+1} \end{bmatrix}.$$

Die Zeilen in $\mathbf{V}$ mit nur dem Element null werden weggelassen.

Multiplizieren Sie die $(p+2) \times (p+1)$ Matrix

$$\mathbf{A} = \begin{bmatrix} \hat{\mathbf{U}} & \mathbf{g} \\ \mathbf{0} & q \\ \mathbf{r}_{n+1} \end{bmatrix} \quad \begin{array}{l} (p \text{ Zeilen}) \\ (1 \text{ Zeile}) \\ (1 \text{ Zeile}) \end{array}$$

mit einem Produkt von elementaren Orthogonalmatrizen, die sich alle nur in zwei Zeilen und zwei Spalten von der Einheitsmatrix $\mathbf{I}_{p+2}$ unterscheiden. Für die Werte $k = 1, 2, \dots, p+1$ wirkt die k -te Orthogonalmatrix auf die k -te und die letzte Zeile, läßt das k -te Element der letzten Zeile weg und ändert die

nachfolgenden Elemente in der letzten Zeile. Die k -te Orthogonalmatrix hat die Form

$$\begin{bmatrix} 1 & & & & & & & & & \\ & \ddots & & & & & & & & \\ & & \ddots & & & & & & & \\ & & & 1 & & & & & & \\ & & & & c & & & & & \\ & & & & & 1 & & & & \\ & & & & & & \ddots & & & \\ & & & & & & & \ddots & & \\ & & 0 & & & & & & \ddots & \\ & & & & & & & & & 1 \\ & & & & & & & & & & c \\ & & & & & & & & & & & -s \\ & & & & & & & & & & & & 1 \\ & & & & & & & & & & & & & c \end{bmatrix}$$

wo $c = \cos(\theta)$, $s = \sin(\theta)$ und $\theta = \tan^{-1}(a_{p+2,k}/a_{kk})$ gilt. Nachdem die Matrix $\mathbf{A}$ mit $p+1$ solcher Faktoren multipliziert worden ist, nimmt sie die Form

$$\mathbf{A}^* = \begin{bmatrix} \mathbf{U}^* & \mathbf{g}^* \\ \mathbf{0} & q^* \\ \mathbf{0} & 0 \end{bmatrix} \begin{matrix} (p \text{ Zeilen}) \\ (1 \text{ Zeile}) \\ (1 \text{ Zeile}) \end{matrix}$$

an, wo $\mathbf{U}^*$ ebenfalls eine obere Dreiecksmatrix ist. Die Lösung $\mathbf{b}^{(n+1)}$ des erweiterten Systems mit $n+1$ Zeilen erhalten Sie, wenn Sie das Gleichungssystem

$$\begin{bmatrix} \mathbf{U}^* & \mathbf{g}^* \\ \mathbf{0} & q^* \end{bmatrix} \begin{bmatrix} \mathbf{b}^{(n+1)} \\ -1 \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ -q^* \end{bmatrix}.$$

lösen. Zusätzliche Zeilen mit weiteren Meßwerten können in das Gleichungssystem eingeschlossen werden, indem die letzte Zeile in $\mathbf{A}^*$ durch $\mathbf{r}_{n+2}$ ersetzt und die Faktorisierung wiederholt wird. Sie können beliebig viele Zeilen hinzufügen, ohne den hierfür erforderlichen Speicherbedarf zu vergrößern.

Das folgende Programm beginnt mit $n = 0$ und $\mathbf{A} = \mathbf{0}$. Geben Sie der Reihe nach die Zeilen $\mathbf{r}_m$ mit $m = 1, 2, \dots$ ein. Sobald $\hat{\mathbf{U}} \neq 0$ ist, können Sie nach Eingabe aller Zeilen die momentane Lösung $\mathbf{b}$ berechnen.

Außerdem können Sie mit diesem Programm gewichtete Kleinste Quadrate Aufgaben und Kleinste Quadrate Aufgaben mit linearen Nebenbedingungen lösen. Führen Sie dafür die hierzu erforderlichen Substitutionen, die weiter oben in diesem Abschnitt unter Orthogonal-Faktorisierung beschrieben sind, durch.

Tastenfolge

[g] [P/R]
[f] [CLEAR] [PRGM]
[f] [LBL] [A]

[STO] 2
1
[STO] 1
[f] [LBL] 4
[RCL] [DIM] [A]
 $x \div y$
[STO] 0
[f] [LBL] 5
[RCL] 1
[R/S]
[RCL] 2
[x]
[f] [USER] [STO] [A]
[f] [USER]
[GTO] 5
[GTO] 4
[f] [LBL] [B]

[RCL] [DIM] [A]

 $x \div y$
[STO] 2
[f] [MATRIX] 1
[f] [LBL] 1

[g] [CF] 0
[RCL] 2
[RCL] 0
[RCL] [g] [A]
[RCL] [A]
[g] [TEST] 2
[g] [SF] 0

Anzeige

000–
001–42,21,11

002– 44 2
003– 1
004– 44 1
005–42,21, 4
006–45,23,11
007– 34
008– 44 0
009–42,21, 5
010– 45 1
011– 31
012– 45 2
013– 20
014u 44 11

015– 22 5
016– 22 4
017–42,21,12

018–45,23,11

019– 34
020– 44 2
021–42,16, 1
022–42,21, 1

023–43, 5, 0
024– 45 2
025– 45 0
026–45,43,11
027– 45 11
028–43,30, 2
029–43, 4, 0

Programm-Modus.

Programm zur Eingabe der neuen Zeile.

Speichert Gewicht in R_2 .

Speichert $l=1$ in R_1 .

Speichert $k=p+2$ in R_0 .

Programm zur Aktualisierung der Matrix A.

Ruft die Dimensionen $p+2$ und $p+1$ zurück.

Speichert $p+2$ in R_2 .

Setzt $k=l=1$.

Verzweigung, um i -te Zeile zu aktualisieren.

Ruft $a_{p+2,k}$ zurück.

Ruft a_{kk} zurück.

Prüft, ob $a_{kk} < 0$ ist.

Setzt Flag 0, falls Diagonalelement negativ.

Tastenfolge

Anzeige

[g] [ABS]

030- 43 16

[g] [↔P]

031- 43 1

Berechnet θ .

[g] [CLx]

032- 43 35

1

033- 1

[f] [↔R]

034- 42 1

Berechnet $x = \cos \theta$ und $y = \sin \theta$.

[g] [F7] 0

035-43, 6, 0

[CHS]

036- 16

Setzt $x = c$ und $y = s$.

[f] [I]

037- 42 25

Bildet $s + ic$.

[R↔]

038- 33

[f] [LBL] 2

039-42,21, 2

Unterprogramm zur Rotation der k -ten Zeile.

[g] [R↔]

040- 43 33

[RCL] [A]

041- 45 11

Ruft a_{kl} zurück.

[RCL] 2

042- 45 2

[RCL] 1

043- 45 1

[RCL] [g] [A]

044-45,43,11

Ruft $a_{p+2,l}$ zurück.

[f] [I]

045- 42 25

Bildet $a_{kl} - ia_{p+2,l}$.

[x]

046- 20

[RCL] 2

047- 45 2

[RCL] 1

048- 45 1

[STO] [g] [A]

049-44,43,11

Speichert neues a_{kl} .

[f] [Re↔Im]

050- 42 30

[f] [USER] [STO] [A]

051u 44 11

Speichert neues $a_{p+2,l}$, erhöht R_0 und R_1 .

[f] [USER]

[RCL] 1

052- 45 1

Ruft l (Spalte) zurück.

[RCL] 0

053- 45 0

Ruft k (Zeile) zurück.

[g] [x≤y]

054- 43 10

Prüft, ob $k \leq l$.

[GTO] 2

055- 22 2

Rücksprung, bis Spalte auf 1 gesetzt ist.

[g] [CF] 8

056-43, 5, 8

Schaltet den Komplex-Modus aus.

[STO] 1

057- 44 1

Speichert k in R_1 (l).

[RCL] 2

058- 45 2

g $x \leq y$

g RTN

GTO 1

f LBL C

RCL DIM A

ENTER

f DIM A

STO 0

STO 1

1

f DIM C

0

STO MATRIX C

EEX

9

9

CHS

RCL A

g $x=0$

R↓

CHS

RCL 0

1

STO g C

RCL MATRIX C

RCL MATRIX A

f RESULT C

÷

RCL 0

1

+

RCL 0

f DIM A

1

059– 43 10

060– 43 32

061– 22 1

062–42,21,13

063–45,23,11

064– 36

065–42,23,11

066– 44 0

067– 44 1

068– 1

069–42,23,13

070– 0

071–44,16,13

072– 26

073– 9

074– 9

075– 16

076– 45 11

077– 43 20

078– 33

079– 16

080– 45 0

081– 1

082–44,43,13

083–45,16,13

084–45,16,11

085–42,26,13

086– 10

087– 45 0

088– 1

089– 40

090– 45 0

091–42,23,11

092– 1

Prüft, ob $p + 2 \leq k$.

Kehrt zur letzten Zeile zurück.

Neuer Schleifendurchlauf.

Programm zur Berechnung der momentanen Lösung.

Eliminiert die letzte Zeile in A.

Speichert $p + 1$ in R_0 .Speichert $p + 1$ in R_1 .Dimensioniert Matrix C $(p + 1) \times 1$.

Setzt Matrix C gleich 0.

Bildet 10^{-99} .Ruft $q = a_{p+1,p+1}$ zurück.Prüft, ob $q = 0$.Nimmt 10^{-99} , falls $q = 0$.Setzt $c_{p+1,1} = -q$.Speichert $A^{-1}C$ in C.Dimensioniert Matrix A $(p + 2) \times (p + 1)$.

Tastenfolge

Anzeige

[−]	093− 30	
1	094− 1	
[f] [DIM] [C]	095−42,23,13	Dimensioniert Matrix C $p \times 1$.
[RCL] [A]	096− 45 11	Ruft q zurück.
[f] [MATRIX] 1	097−42,16, 1	Setzt $k = l = 1$.
[g] [RTN]	098− 43 32	

Verwendete Labels: A, B, C und 1 bis 5.

Verwendete Register: R_0 , R_1 und R_2 ($p+2$ und w).

Verwendete Matrizen: **A** (Arbeitsmatrix) und **C** (Geschätzte Parameterwerte).

Verwendete Flags: 0 und 8.

Nach Speichern dieses Programms verfügt der HP-15C noch über genügend Speicherplatz, um mit bis zu $p=4$ Parametern zu arbeiten. Wenn die Programme «A» und «C» weggelassen werden, können Sie mit $p=5$ Parametern arbeiten. In beiden Fällen können Sie beliebig viele Zeilen eingeben.

Um dieses Programm zu benutzen:

1. Drücken Sie **2 [f] [DIM] [(i)]**, um die Speicher R_0 bis R_2 zu reservieren.
2. Drücken Sie **[f] [USER]**, um den User-Modus zu aktivieren.
3. Geben Sie $(p+2)$ und $(p+1)$ in den Stack ein, und drücken Sie dann **[f] [DIM] [A]**, um Matrix **A** zu dimensionieren. Die Dimensionen hängen von der Anzahl p der Parameter, die Sie verwenden, ab.
4. Drücken Sie **[STO] [MATRIX] [A]**, um Matrix **A** zu initialisieren.
5. Geben Sie die Gewichte w_k für die momentanen Zeilen ein, und drücken Sie dann **[A]**. In der Anzeige sollte 1.0000 erscheinen, um anzuzeigen, daß das Programm auf die Eingabe des ersten Elementes der Zeile wartet. (Für gewöhnliche Kleinste Quadrate Aufgaben wählen Sie für jede Zeile $w_k = 1$.)
6. Geben Sie die Elemente der m -ten Zeile der Matrix **A** ein, indem Sie x_m **[R/S]** x_{m2} **[R/S]** ... x_{mp} **[R/S]** y_m **[R/S]** drücken. Nachdem ein Element eingegeben ist, sollte in der Anzeige die Nummer des nächsten einzugebenden Elements erscheinen. (Wenn Ihnen bei der Eingabe der Elemente ein Fehler unterläuft, gehen Sie zurück zu Schritt 5 und 6 der betreffenden Zeile.)
7. Drücken Sie **[B]**, um die Faktorisierung zu aktualisieren, so daß die in den letzten zwei Schritten eingegebene Zeile berücksichtigt wird.

8. Sie können nun $\boxed{\text{C}} \boxed{\text{g}} \boxed{x^2}$ drücken, um die Residuenquadratsumme q^2 zu berechnen und anzuzeigen und die momentane Lösung $\mathbf{b}$ zu berechnen. Drücken Sie dann $\boxed{\text{RCL}} \boxed{\text{C}} p$ mal, um der Reihe nach $b_1, b_2, \dots, b_p$ anzuzeigen.

9. Wiederholen Sie die Schritte 5 bis 8 für alle weiteren Zeilen.

Beispiel: Verwenden Sie dieses Programm und die VPI-Daten des vorigen Beispiels, um das Modell

$$y = b_1 + b_2 x_2 + b_3 x_3 + r$$

anzupassen, wobei y, x_2 und x_3 für VPI, HPI und AZ (alle in Prozent) stehen. Diese Aufgabe verwendet $p=3$ Parameter, so daß Matrix $\mathbf{A}$ auf 5×4 dimensioniert sein sollte.

Die Zeilen der Matrix $\mathbf{A}$ sind $(1, x_{m2}, x_{m3}, y_m)$ mit $m = 1, 2, \dots, 11$. Jede Zeile hat das Gewicht $w_m = 1$.

Tastenfolge	Anzeige	
$\boxed{\text{g}} \boxed{\text{P/R}}$		Run-Modus.
2 $\boxed{\text{f}} \boxed{\text{DIM}} \boxed{(i)}$	2.0000	Reserviert R_0 bis R_2 .
$\boxed{\text{f}} \boxed{\text{USER}}$	2.0000	Aktiviert User-Modus.
$\boxed{\text{f}} \boxed{\text{MATRIX}} 0$	2.0000	Löscht den Matrixspeicher.
5 $\boxed{\text{ENTER}} 4$	4	
$\boxed{\text{f}} \boxed{\text{DIM}} \boxed{\text{A}}$	4.0000	Dimensioniert Matrix $\mathbf{A}$ 5×4 .
0 $\boxed{\text{STO}} \boxed{\text{MATRIX}} \boxed{\text{A}}$	0.0000	Speichert null in allen Elementen.
1 $\boxed{\text{A}}$	1.0000	Gibt Gewicht für Zeile 1 ein.
1 $\boxed{\text{R/S}}$	2.0000	Gibt $x_{1,1}$ ein.
3.9 $\boxed{\text{R/S}}$	3.0000	Gibt $x_{1,2}$ ein.
3.5 $\boxed{\text{R/S}}$	4.0000	Gibt $x_{1,3}$ ein.
5.4 $\boxed{\text{R/S}}$	1.0000	Gibt y_1 ein.
$\boxed{\text{B}}$	5.0000	Aktualisiert die Faktorisierung.
:	:	
1 $\boxed{\text{A}}$	1.0000	Gibt Gewicht für Zeile 11 ein.
1 $\boxed{\text{R/S}}$	2.0000	Gibt $x_{11,1}$ ein.
19.3 $\boxed{\text{R/S}}$	3.0000	Gibt $x_{11,2}$ ein.
5.8 $\boxed{\text{R/S}}$	4.0000	Gibt $x_{11,3}$ ein.
11.5 $\boxed{\text{R/S}}$	1.0000	Gibt y_{11} ein.
$\boxed{\text{B}}$	5.0000	Aktualisiert die Faktorisierung.

Tastenfolge

Anzeige

C	3.6759	Berechnet augenblickliche Schätzungen und q .
f FIX 9	3.675891055	
g x²	13.51217505	Berechnet Residuensumme der Quadrate q^2 .
RCL C	1.245864306	Zeigt $b_1^{(1)}$ an.
RCL C	0.379758235	Zeigt $b_2^{(1)}$ an.
RCL C	0.413552221	Zeigt $b_3^{(1)}$ an.

Diese Schätzungen stimmen (auf 3 Einheiten in der neunten Stelle) mit den Ergebnissen des vorigen Beispiels überein, das Normalgleichungen verwendete. Außerdem können Sie zusätzliche Daten einschließen und die Schätzungen für die Parameter aktualisieren. Geben Sie beispielsweise zusätzlich folgende Daten für 1968 ein: VPI = 4.2, HPI = 2.5 und AZ = 3.6.

Tastenfolge

Anzeige

1 A	1.000000000	Gibt Gewicht für die neue Zeile ein.
1 R/S	2.000000000	Gibt $x_{12,1}$ ein.
2.5 R/S	3.000000000	Gibt $x_{12,2}$ ein.
3.6 R/S	4.000000000	Gibt $x_{12,3}$ ein.
4.2 R/S	1.000000000	Gibt y_{12} ein.
B	5.000000000	Aktualisiert Faktorisierung.
C	3.700256908	
g x²	13.69190119	Berechnet Residuenquadratsumme.
RCL C	1.581596327	Zeigt $b_1^{(12)}$ an.
RCL C	0.373826487	Zeigt $b_2^{(12)}$ an.
RCL C	0.370971848	Zeigt $b_3^{(12)}$ an.
f FIX 4	0.3710	
f USER	0.3710	Desaktiviert User-Modus.

Eigenwerte einer symmetrischen reellen Matrix

Die Eigenwerte einer quadratischen Matrix A sind die Nullstellen λ_j ihrer charakteristischen Gleichung

$$\det(A - \lambda I) = 0.$$

Wenn $\mathbf{A}$ reell und symmetrisch ($\mathbf{A} = \mathbf{A}^T$) ist, sind ihre Eigenwerte λ_j alle reell und haben orthogonale Eigenvektoren $\mathbf{q}_j$. Dann gilt

$$\mathbf{A}\mathbf{q}_j = \lambda_j\mathbf{q}_j$$

und

$$\mathbf{q}_j^T \mathbf{q}_k = \begin{cases} 0 & \text{falls } j \neq k. \\ 1 & \text{falls } j = k. \end{cases}$$

Die Eigenvektoren ($\mathbf{q}_1, \mathbf{q}_2, \dots$) bilden die Spalten einer orthogonalen Matrix $\mathbf{Q}$, die die Gleichungen

$$\mathbf{Q}^T \mathbf{A} \mathbf{Q} = \text{diag}(\lambda_1, \lambda_2, \dots)$$

und

$$\mathbf{Q}^T = \mathbf{Q}^{-1}$$

erfüllt. Eine orthogonale Variablentransformation $\mathbf{x} = \mathbf{Q}\mathbf{z}$, der einer Drehung der Koordinatenachsen entspricht, bringt eine Familie von quadratischen Flächen ($\mathbf{x}^T \mathbf{A} \mathbf{x} = \text{konstant}$) auf die Form

$$\mathbf{z}^T (\mathbf{Q}^T \mathbf{A} \mathbf{Q}) \mathbf{z} = \sum_j^k \lambda_j z_j^2 = \text{konstant}.$$

Aus dieser Form der Gleichung können Sie erkennen, um was für Flächen es sich handelt (Ellipsoide, Hyperboloide, Paraboloid, Kegel, Zylinder, Ebenen), da die Halbachsen der Fläche entlang der Achsen des neuen Koordinatensystems liegen.

Das folgende Programm beginnt mit einer gegebenen Matrix $\mathbf{A}$, die symmetrisch sei (wenn sie es nicht ist, muß sie durch die Matrix $(\mathbf{A} + \mathbf{A}^T)/2$ ersetzt werden, die symmetrisch ist).

Aus dieser symmetrischen Matrix $\mathbf{A}$ konstruiert das Programm eine schiefsymmetrische Matrix (d.h. eine Matrix, für die gilt $\mathbf{B} = -\mathbf{B}^T$), wobei die folgende Gleichung verwendet wird:

$$b_{ij} = \begin{cases} \tan(\frac{1}{4} \tan^{-1}(2a_{ij}/(a_{ii} - a_{jj}))) & \text{falls } i \neq j \text{ und } a_{ij} \neq 0 \\ 0 & \text{falls } i = j \text{ oder } a_{ij} = 0. \end{cases}$$

Dann muß $\mathbf{Q} = 2(\mathbf{I} + \mathbf{B})^{-1} - \mathbf{I}$ eine orthogonale Matrix sein, deren Spalten die Eigenwerte von $\mathbf{A}$ annähern; je kleiner alle Elemente von $\mathbf{B}$ sind, desto besser ist die Näherung. Daher muß $\mathbf{Q}^T \mathbf{A} \mathbf{Q}$ sich weniger von einer Diagonalmatrix unterscheiden als $\mathbf{A}$, aber dieselben Eigenwerte haben. Wenn $\mathbf{Q}^T \mathbf{A} \mathbf{Q}$ nicht nahe genug diagonal ist, wird es anstelle von $\mathbf{A}$ für eine Wiederholung der Rechnung verwendet.

Auf diese Weise werden sukzessive orthogonale Transformationen $Q_1, Q_2, Q_3, \dots$ auf A angewandt, um eine Folge $A_1, A_2, A_3, \dots$ zu erzeugen, wo

$$A_j = (Q_1 Q_2 \dots Q_j)^T A Q_1 Q_2 \dots Q_j$$

gilt und jedes A_j näher diagonal als das vorige ist.

Dieser Vorgang führt normalerweise zu schiefsymmetrischen Matrizen A_j , deren Elemente alle klein sind und die schnell zu einer Diagonalmatrix A konvergieren. Wenn jedoch einige der Eigenwerte der Matrix A sehr nahe beieinander, aber weit weg von den anderen liegen, ist die Konvergenz langsam; glücklicherweise kommt dies selten vor.

Nach jeder Iteration bleibt das Programm stehen und zeigt

$$\frac{1}{2} \sum_j |\text{nichtdiagonale Elemente von } A_j| / \|A_j\|_F$$

an. Diese Größe kann als ein Maß für die Diagonalität von A_j betrachtet werden. Wenn dieser Wert nicht vernachlässigbar klein ist, drücken Sie $\boxed{R/S}$, um A_{j+1} zu berechnen; wenn er vernachlässigbar ist, dann nähern die Diagonalelemente von A_j die Eigenwerte von A an. Für 2×2 Matrizen benötigt das Programm nur eine Iteration und für 3×3 Matrizen selten mehr als sechs. Für 4×4 Matrizen braucht das Programm etwas länger und benötigt allen verfügbaren Speicherplatz; im allgemeinen kommt es mit 6 oder 7 Iterationen aus. Wenn einige Eigenwerte jedoch sehr nahe beieinander, aber relativ weit von den übrigen entfernt liegen, werden unter Umständen 10 bis 16 Iterationen benötigt.

Tastenfolge

g **P/R**
f **CLEAR** **PRGM**
f **LBL** **A**
RCL **MATRIX** **A**
STO **MATRIX** **B**
STO **MATRIX** **C**
f **MATRIX** **4**
RCL **MATRIX** **B**
STO **RESULT**
+

Anzeige

000–
 001–42,21,11
 002–45,16,11
 003–44,16,12
 004–44,16,13
 005–42,16, 4
 006–45,16,12
 007– 44 26
 008– 40

Programm-Modus.

Dimensioniert **B**.
 Dimensioniert **C**.
 Transponiert **A**.

Tastenfolge

2
 ÷
 STO MATRIX A
 f MATRIX 8
 STO 2
 g CLx
 STO 3

 STO MATRIX C
 f MATRIX 1
 f LBL 0

 RCL 0
 RCL 1
 g TEST 5
 GTO 3
 g TEST 7
 GTO 1
 x↔y
 RCL g B
 CHS
 f USER STO B
 f USER
 GTO 0
 f LBL 1
 RCL g A
 g ABS
 STO + 3

 g LSTx
 ENTER
 +
 RCL 0
 ENTER
 RCL g A
 RCL 1
 ENTER

Anzeige

009– 2
 010– 10
 011–44,16,11
 012–42,16, 8
 013– 44 2
 014– 43 35
 015– 44 3

 016–44,16,13
 017–42,16, 1
 018–42,21, 0

 019– 45 0
 020– 45 1
 021–43,30, 5
 022– 22 3
 023–43,30, 7
 024– 22 1
 025– 34
 026–45,43,12
 027– 16
 028u 44 12

 029– 22 0
 030–42,21, 1
 031–45,43,11
 032– 43 16
 033–44,40, 3

 034– 43 36
 035– 36
 036– 40
 037– 45 0
 038– 36
 039–45,43,11
 040– 45 1
 041– 36

Berechnet $A = (A + A^T)/2$.Berechnet $\|A\|_F$.Speichert $\|A\|_F$ in R_2 .

Initialisiert Summe der Nicht-diagonalelemente.

Setzt $C = 0$.Setzt $R_0 = R_1 = 1$.Programm zur Konstruktion von Q .

Prüft, ob Zeile = Spalte.

Prüft, ob Spalte > Zeile.

Setzt $b_{ij} = -b_{ji}$.

Programm für Spalte > Zeile.

Berechnet $|a_{ij}|$.

Summiert Nichtdiagonalelemente.

Berechnet $2a_{ij}$.Ruft a_{ji} zurück.

Tastenfolge

RCL g A

-

g TEST 3

GTO 2

CHS

 $x \geq y$

CHS

 $x \geq y$

f LBL 2

g $\rightarrow P$

g CLx

4

 $\div$

TAN

f USER STO B

f USER

GTO 0

f LBL 3

1

STO C

f USER STO B

f USER

GTO 0

RCL 3

RCL $\div$ 2

R/S

2

RCL MATRIX B

 $\div$

RCL MATRIX C

-

RCL MATRIX A

f RESULT C

f MATRIX 5

Anzeige

042-45,43,11

043- 30

044-43,30, 3

045- 22 2

046- 16

047- 34

048- 16

049- 34

050-42,21, 2

051- 43 1

052- 43 35

053- 4

054- 10

055- 25

056u 44 12

057- 22 0

058-42,21, 3

059- 1

060- 44 13

061u 44 12

062- 22 0

063- 45 3

064-45,10, 2

065- 31

066- 2

067-45,16,12

068- 10

069-45,16,13

070- 30

071-45,16,11

072-42,26,13

073-42,16, 5

Ruft a_{jj} zurück.Berechnet $a_{ii} - a_{jj}$.Prüft, ob $x \geq 0$.Hält den Drehwinkel zwischen -90° und 90° .

Berechnet den Drehwinkel.

Berechnet b_{ij} .Programmteil für
Zeile = Spalte.Setzt $c_{ii} = 1$.Setzt $b_{ii} = 1$.Berechnet den Nicht-
diagonalfaktor.

Zeigt den Faktor an.

Berechnet $\mathbf{B} = 2$
 $(\mathbf{I} + \text{Schiefe})^{-1} - \mathbf{I}$.Berechnet $\mathbf{C} = \mathbf{B}^T \mathbf{A}$.

Tastenfolge

RCL MATRIX B
 f RESULT A
 x
 GTO A

Anzeige

074-45,16,12
 075-42,26,11
 076- 20
 077- 22 11

Berechnet $\mathbf{A} = \mathbf{B}^T \mathbf{A} \mathbf{B}$.

Verwendete Labels: A, 0, 1, 2 und 3.

Verwendete Speicher: R_0 , R_1 , R_2 (Summe der Nichtdiagonalelemente) und R_3 ($\|\mathbf{A}_j\|_F$).Verwendete Matrizen: $\mathbf{A}$ ($\mathbf{A}_j$), $\mathbf{B}$ ($\mathbf{Q}_j$) und $\mathbf{C}$.

Um das Programm zu benutzen:

1. Drücken Sie 4 f DIM (i), um die Speicher R_0 bis R_4 zu reservieren.
2. Drücken Sie f USER, um den User-Modus zu aktivieren.
3. Dimensionieren und geben Sie die Elemente der Matrix $\mathbf{A}$ mit f DIM A und STO A ein. Die Dimensionierung kann maximal 4×4 sein, vorausgesetzt, es ist genügend Speicherplatz für die Matrizen $\mathbf{B}$ und $\mathbf{C}$ derselben Dimension vorhanden.
4. Drücken Sie A, um den Nichtdiagonalfaktor zu berechnen und anzuzeigen.
5. Drücken Sie mehrmals R/S, bis der angezeigte Faktor vernachlässigbar klein ist, etwa 10^{-8} oder kleiner.
6. Drücken Sie mehrmals RCL A, um die Elemente der Matrix $\mathbf{A}$ zu betrachten. Die Diagonalelemente sind die Eigenwerte.

Beispiel: Was für eine Fläche zweiter Ordnung wird durch die folgende Gleichung beschrieben?

$$\mathbf{x}^T \mathbf{A} \mathbf{x} = [x_1 \quad x_2 \quad x_3] \begin{bmatrix} 0 & 1 & 2 \\ 1 & 2 & 3 \\ 2 & 3 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$= 2x_1x_2 + 4x_1x_3 + 2x_2^2 + 6x_2x_3 + 4x_3^2$$

$$= 7$$

Tastenfolge

g P/R
 4 f DIM (i)
 f USER

Anzeige

4.0000
 4.0000

Run-Modus.

Ordnet Speicher zu.

Aktiviert User-Modus.

Tastenfolge	Anzeige	
3 ENTER f DIM A	3.0000	Dimensioniert A 3×3 .
f MATRIX 1	3.0000	Setzt R_0 und R_1 gleich 1.
0 STO A	0.0000	Gibt a_{11} ein.
1 STO A	1.0000	Gibt a_{12} ein.
⋮		
3 STO A	3.0000	Gibt a_{32} ein.
4 STO A	4.0000	Gibt a_{33} ein.
A	0.8660	Berechnet Faktor – zu groß.
R/S	0.2304	Immer noch zu groß.
R/S	0.1039	Immer noch zu groß.
R/S	0.0060	Immer noch zu groß.
R/S	3.0463 –05	Immer noch zu groß.
R/S	5.8257 –10	Nun ist der Faktor vernachlässigbar klein.
RCL A	–0.8730	Ruft $a_{11} = \lambda_1$ zurück.
RCL A	–9.0006 –10	Ruft a_{12} zurück.
RCL A	–2.0637 –09	Ruft a_{13} zurück.
RCL A	–9.0006 –10	Ruft a_{21} zurück.
RCL A	9.3429 –11	Ruft $a_{22} = \lambda_2$ zurück.
RCL A	1.0725 –09	Ruft a_{23} zurück.
RCL A	–2.0637 –09	Ruft a_{31} zurück.
RCL A	1.0725 –09	Ruft a_{32} zurück.
RCL A	6.8730	Ruft $a_{33} = \lambda_3$ zurück.
f USER	6.8730	Desaktiviert User-Modus.

Im neuen Koordinatensystem wird die Gleichung der quadratischen Fläche näherungsweise zu

$$-0.8730z_1^2 + 0z_2^2 + 6.8730z_3^2 = 7.$$

Diese Gleichung beschreibt einen hyperbolischen Zylinder.

Eigenvektoren einer symmetrischen reellen Matrix

Wie schon im obigen Beispiel besprochen wurde, hat eine reelle symmetrische Matrix **A** reelle Eigenwerte $\lambda_1, \lambda_2, \dots$ und dazugehörige orthogonale Eigenvektoren $\mathbf{q}_1, \mathbf{q}_2, \dots$.

Das folgende Programm verwendet eine inverse Iteration, um einen zu einem Eigenwert λ_k gehörigen Eigenvektor $\mathbf{q}_k$ mit der Bedingung $\|\mathbf{q}_k\|_R = 1$ zu berechnen. Das Verfahren geht von einem Startvektor $\mathbf{z}^{(0)}$ aus und berechnet dann sukzessive Vektoren $\mathbf{w}^{(n)}$ und $\mathbf{z}^{(n)}$ aus den Gleichungen

$$(\mathbf{A} - \lambda \mathbf{I})\mathbf{w}^{(n+1)} = \mathbf{z}^{(n)}$$

$$\mathbf{z}^{(n+1)} = s \mathbf{w}^{(n+1)} / \|\mathbf{w}^{(n+1)}\|_R$$

wo s für das Vorzeichen der ersten Komponente von $\mathbf{w}^{(n+1)}$ mit dem größten Absolutbetrag steht. Die Iterationen werden solange fortgesetzt, bis die Folge der $\mathbf{z}^{(n)}$ konvergiert. Dieser Vektor ist der zum Eigenwert λ_k gehörige Eigenvektor $\mathbf{q}_k$.

Der für λ_k verwendete Wert braucht nicht exakt zu sein; selbst bei kleinen Ungenauigkeiten in λ_k wird der zugehörige Eigenvektor genau berechnet. Auch brauchen Sie sich keine Gedanken darüber zu machen, vielleicht einen zu genauen Wert für λ_k zu erhalten (so daß $\mathbf{A} - \lambda_k \mathbf{I}$ singular werden könnte), denn der HP-15C kann selbst dann den Eigenvektor berechnen, wenn $\mathbf{A} - \lambda_k \mathbf{I}$ sehr schlecht konditioniert ist.

Bei diesem Verfahren muß eine Komponente des Vektors $\mathbf{z}^{(0)}$ in der Richtung des unbekannten Eigenvektors $\mathbf{q}_k$ ungleich null sein. Da $\mathbf{z}^{(0)}$ weiter keine Bedingungen erfüllen braucht, verwendet das Programm Zufallszahlen als Komponenten von $\mathbf{z}^{(0)}$. Nach jeder Iteration zeigt das Programm $\|\mathbf{z}^{(n+1)} - \mathbf{z}^{(n)}\|_R$ an, um die Konvergenz zu zeigen.

Dieses Programm kann auch für eine Matrix $\mathbf{A}$ verwendet werden, die nicht symmetrisch ist, aber eine diagonale kanonische Jordanform besitzt – d.h. es existiert eine nichtsinguläre Matrix $\mathbf{P}$ mit $\mathbf{P}^{-1}\mathbf{A}\mathbf{P} = \text{diag}(\lambda_1, \lambda_2, \dots)$.

Tastenfolge

Anzeige

[g] P/R

Programm-Modus.

[f] CLEAR [PRGM]

000–

[f] [LBL] [C]

001–42,21,13

[STO] 2

002– 44 2

Speichert Eigenwert in R₂.

[RCL] [MATRIX] [A]

003–45,16,11

[STO] [MATRIX] [B]

004–44,16,12

Speichert A in B.

[RCL] [DIM] [A]

005–45,23,11

[STO] 0

006– 44 0

[f] [LBL] 4

007–42,21, 4

[RCL] 0

008– 45 0

[STO] 1

009– 44 1

[RCL] [B]

010– 45 12

Tastenfolge

RCL **[-]** 2
STO **B**

f **DSE** 0
GTO 4
RCL **DIM** **A**
1
f **DIM** **C**
f **MATRIX** 1
f **LBL** 5
f **RAN#**
f **USER** **STO** **C**
f **USER**
GTO 5
f **LBL** 6

RCL **MATRIX** **C**
STO **MATRIX** **D**
STO **RESULT**
RCL **MATRIX** **B**
÷
ENTER
f **MATRIX** 7
÷
f **MATRIX** 1
f **LBL** 7

f **USER** **RCL** **C**
f **USER**
ENTER

g **ABS**
1
g **TEST** 6
GTO 7
RCL **MATRIX** **C**
g **LSTx**
÷

Anzeige

011–45,30, 2
012– 44 12

013–42, 5, 0
014– 22 4
015–45,23,11
016– 1
017–42,23,13
018–42,16, 1
019–42,21, 5
020– 42 36
021u 44 13

022– 22 5
023–42,21, 6

024–45,16,13
025–44,16,14
026– 44 26
027–45,16,12
028– 10
029– 36
030–42,16, 7
031– 10
032–42,16, 1
033–42,21, 7

034u 45 13

035– 36

036– 43 16
037– 1
038–43,30, 6
039– 22 7
040–45,16,13
041– 43 36
042– 10

Modifiziert Diagonalelemente in **B**.

Dimensioniert **C** $n \times 1$.

Speichert Zufallskomponenten in **C**.

Programm zur iterativen Berechnung von $\mathbf{z}^{(n)}$ und $\mathbf{w}^{(n)}$.

Speichert $\mathbf{z}^{(n)}$ in **D**.

Berechnet $\mathbf{w}^{(n+1)}$ in **C**.

Berechnet $\pm \mathbf{z}^{(n+1)}$ in **C**.

Programm zum Auffinden des Vorzeichens des größten Elements.

(Diese Zeile wird beim letzten Element übersprungen.)

Prüft, ob $|a_j| \neq 1$.

Ruft größtes a_j zurück.
Berechnet $\mathbf{z}^{(n+1)}$ in **C**.

Tastenfolge

[RCL] [MATRIX] [D]

[STO] [RESULT]

[-]

[f] [MATRIX] 7

[f] [MATRIX] 1

[R/S]

[GTO] 6

Anzeige

043-45,16,14

044- 44 26

045- 30

046-42,16, 7

047-42,16, 1

048- 31

049- 22 6

Berechnet $\mathbf{z}^{(n+1)} - \mathbf{z}^{(n)}$ in **D**.Berechnet $\|\mathbf{z}^{(n+1)} - \mathbf{z}^{(n)}\|_R$.Setzt $R_0 = R_1 = 1$, um **C** zu betrachten.

Zeigt Konvergenzparameter an.

Verwendete Labels: C, 4, 5, 6 und 7.

Verwendete Speicher: R_0 , R_1 und R_2 (Eigenwert).Verwendete Matrizen: **A** (ursprüngliche Matrix), **B** ($\mathbf{A} - \lambda \mathbf{I}$), $(\mathbf{z}^{(n+1)})$ und **D** $(\mathbf{z}^{(n+1)} - \mathbf{z}^{(n)})$.

Um dieses Programm zu benutzen:

1. Drücken Sie 2 [f] [DIM] [(i)], um die Speicher R_0 , R_1 und R_2 zu reservieren.
2. Drücken Sie [f] [USER], um den User-Modus zu aktivieren.
3. Dimensionieren Sie Matrix **A** mit [f] [DIM] [A] und geben Sie mit [STO] [A] die Elemente ein.
4. Geben Sie den Eigenwert ein und drücken Sie [C]. Es wird der Korrekturparameter $\|\mathbf{z}^{(1)} - \mathbf{z}^{(0)}\|_R$ angezeigt.
5. Drücken Sie mehrmals [R/S], bis der Korrekturparameter vernachlässigbar klein wird.
6. Drücken Sie mehrmals [RCL] [C], um die Komponenten des Eigenvektors $\mathbf{q}_k$ zu betrachten.

Beispiel: Berechnen Sie die Eigenvektoren $\mathbf{q}_1$, $\mathbf{q}_2$ und $\mathbf{q}_3$ der Matrix **A** des vorigen Beispiels.

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 2 \\ 1 & 2 & 3 \\ 2 & 3 & 4 \end{bmatrix}$$

Tastenfolge	Anzeige	
[g] [P/R]		Run-Modus.
2 [f] [DIM] [(i)]	2.0000	Reserviert Speicher R ₀ bis R ₂ .
[f] [USER]	2.0000	Aktiviert User-Modus.
3 [ENTER] [f] [DIM] [A]	3.0000	Dimensioniert Matrix A 3 × 3.
0 [STO] [A]	0.0000	Gibt Elemente von A ein.
1 [STO] [A]	1.0000	
⋮		
4 [STO] [A]	4.0000	
.8730 [CHS]	-0.8730	Gibt λ ₁ = -0.8730 (als Näherung).
[C]	0.8982	$\ z^{(1)} - z^{(0)}\ \cdot *$
[R/S]	0.0001	$\ z^{(2)} - z^{(1)}\ \cdot *$
[R/S]	2.4000	-09 $\ z^{(3)} - z^{(2)}\ \cdot *$
[R/S]	1.0000	-10 $\ z^{(4)} - z^{(3)}\ \cdot *$
[R/S]	0.0000	$\ z^{(5)} - z^{(4)}\ \cdot *$
[RCL] [C]	1.0000	} Eigenvektor zu λ ₁ .
[RCL] [C]	0.2254	
[RCL] [C]	-0.5492	
0 [C]	0.8485	Verwendet (als Näherung) λ ₂ = 0.
[R/S]	0.0000	} Eigenvektor zu λ ₂ .
[RCL] [C]	-0.5000	
[RCL] [C]	1.0000	
[RCL] [C]	-0.5000	
6.8730 [C]	0.7371	Verwendet (als Näherung) λ ₃ = 6.8730.
[R/S]	1.9372	-06
[R/S]	1.0000	-10
[R/S]	0.0000	

* Die Normen der Korrekturen sind je nach den verwendeten Zufallszahlen verschieden.

Tastenfolge**Anzeige**

RCL C	0.3923	}	Eigenvektor zu λ_3 .
RCL C	0.6961		
RCL C	1.0000		
f USER	1.0000		Desaktiviert User-Modus.

Wenn Matrix **A** 3×3 oder kleiner ist, kann dieses Programm in das vorherige Eigenwertprogramm miteingeschlossen werden. Da das Eigenwertprogramm Matrix **A** verändert, müssen die ursprünglichen Eigenwerte gespeichert werden und die ursprüngliche Matrix wieder in Matrix **A** eingegeben werden, bevor das Eigenvektorprogramm beginnen kann. Das folgende Programm kann angefügt werden, um die berechneten Eigenwerte in Matrix **E** zu speichern.

Tastenfolge**Anzeige**

f LBL E	127-42,21,15		
RCL DIM A	128-45,23,11		
STO 0	129- 44 0		
1	130- 1		Dimensioniert E $n \times 1$.
f DIM E	131-42,23,15		
f LBL 8	132-42,21, 8		
RCL 0	133- 45 0		Ruft Diagonalelement zurück.
ENTER	134- 36		
RCL g A	135-45,43,11		
RCL 0	136- 45 0		Speichert a_{ij} in e_j .
1	137- 1		
STO g E	138-44,43,15		
f DSE 0	139-42, 5, 0		Setzt wieder $R = R_1 = 1$.
GTO 8	140- 22 8		
f MATRIX 1	141-42,16, 1		
g RTN	142- 43 32		Run-Modus.
g P/R			

Verwendete Labels: E und 8.

Verwendete Speicher: Keine zusätzlichen Speicher.

Verwendete Matrizen: **A** (aus dem vorigen Programm) und **E** (Eigenwerte).

Um die kombinierten Eigenwert-, Eigenwertspeicher- und Eigenvektor-Programme für eine bis zu 3×3 Matrix **A** zu benutzen:

1. Lassen Sie das Eigenwertprogramm wie oben beschrieben laufen.

2. Drücken Sie $\boxed{\mathbf{E}}$, um die Eigenwerte zu speichern.
3. Geben Sie nochmals die Elemente der ursprünglichen Matrix in $\mathbf{A}$ ein.
4. Rufen Sie mit $\boxed{\mathbf{RCL}} \boxed{\mathbf{E}}$ den gewünschten Eigenwert aus Matrix $\mathbf{E}$ zurück.
5. Lassen Sie das Eigenvektorprogramm wie oben beschrieben laufen.
6. Wiederholen Sie die Schritte 4 und 5 für jeden Eigenwert.

Optimierung

Als *Optimierung* bezeichnet man die Aufgabenstellung, das Minimum oder Maximum einer gegebenen Funktion zu finden. Oft ist das Verhalten der Funktion in einem bestimmten Gebiet von besonderem Interesse.

Das folgende Programm verwendet das Verfahren des stärksten Abstiegs, um lokale Minima oder Maxima einer reellwertigen Funktion einer oder mehrerer Variabler zu finden. Es ist ein Iterationsverfahren, das den Gradienten der Funktion zur Bestimmung von Stützstellen verwendet. Vier Eingabeparameter steuern die Auswahl der Stützstellen.

Für eine Funktion

$$f(\mathbf{x}) = f(x_1, x_2, \dots, x_n)$$

ist ihr Gradient ∇f als

$$\nabla f(\mathbf{x}) = \begin{bmatrix} \partial f / \partial x_1 \\ \partial f / \partial x_2 \\ \vdots \\ \partial f / \partial x_n \end{bmatrix}.$$

definiert. Die Lösungen von $\nabla f(\mathbf{x}) = \mathbf{0}$ nennt man kritische Punkte von $f(\mathbf{x})$. An einem kritischen Punkt kann die Funktion ein lokales Minimum, ein lokales Maximum oder keins von beiden besitzen.

Für jeden Punkt $\mathbf{x}$ gibt der Gradient einer Funktion $f(\mathbf{x})$ die Richtung ihres steilsten Anstiegs an – d.h. die Richtung von $\mathbf{x}$, in der man den größten Zuwachs von $f(\mathbf{x})$ erhält. Der negative Gradient gibt die Richtung steilsten Abstiegs an. Für den Richtungsvektor gilt:

$$\mathbf{s} = \begin{cases} -\nabla f(\mathbf{x}) & \text{um ein Minimum zu finden} \\ \nabla f(\mathbf{x}) & \text{um ein Maximum zu finden.} \end{cases}$$

Wenn das Programm die Richtung aus dem Gradienten bestimmt hat, sucht es die optimale Entfernung, die $\mathbf{x}_j$ in der Richtung von $\mathbf{s}_j$ verändert werden soll – die Richtung, die die stärkste Annäherung von $f(\mathbf{x})$ an ein Minimum oder Maximum ergibt.

Hierzu sucht das Programm den optimalen Wert t_j , indem es die Steigung der Funktion

$$g_j(t) = f(\mathbf{x}_j + t\mathbf{s}_j)$$

mit wachsenden Werten von t berechnet; solange, bis die Steigung das Vorzeichen wechselt. Dieses Verfahren wird als «beschränkte Suche» bezeichnet, da das Programm versucht, den gesuchten Wert t_j auf ein Intervall zu beschränken. Wenn das Programm einen Vorzeichenwechsel findet, verkleinert es das Intervall, indem es $j+1$ mal halbiert, um den besten Wert für t in der Nähe von $t=0$ zu finden. Dieser Vorgang wird als «Intervallverkleinerung» bezeichnet – je näher $\mathbf{x}_j$ an die gesuchte Lösung konvergiert, desto bessere Werte für t_j liefert es. (Diese beiden Verfahren zusammen werden als «Linien suche» bezeichnet.) Der neue Wert für $\mathbf{x}$ ist dann

$$\mathbf{x}_{j+1} = \mathbf{x}_j + t_j \mathbf{s}_j.$$

Das Programm verwendet vier Parameter, die bestimmen, wie es die gewünschte Lösung angeht. Obwohl Ihnen keine Methode der Linien suche garantieren kann, den optimalen Wert von t zu finden, geben Ihnen die ersten beiden Parameter beträchtliche Freiheit festzulegen, wie das Programm t untersuchen soll.

- d Bestimmung des ersten Schritts u_1 der beschränkten Suche. Als erster Wert für t dient

$$u_1 = \frac{d}{(j+1)\|\mathbf{s}_j\|_F}.$$

Das entspricht einer Schrittweite von

$$\|(\mathbf{x}_j + u_1 \mathbf{s}_j) - \mathbf{x}_j\|_F = \frac{d}{j+1},$$

Man sieht, daß d und die Iterationsnummer bestimmen, wie nahe am letzten Wert $\mathbf{x}$ das Programm die beschränkte Suche beginnt.

- a Bestimmt die Werte $u_2, u_3, \dots$ der folgenden Schritte der beschränkten Suche. Diese Werte von t sind durch

$$u_{i+1} = au_i$$

definiert. α ist im wesentlichen ein Expansionsfaktor, der normalerweise größer als 1 ist und die Folge der t Werte anwachsen läßt.

- e Bestimmt die zulässige Toleranz für den Wert des Gradienten. Die Iteration wird unterbrochen, wenn

$$\|\nabla f(\mathbf{x}_j)\|_F \leq e.$$

- N Bestimmt die maximale Anzahl von Iterationen, die das Programm in der beschränkten Suche und dem allgemeinen Optimierungsverfahren durchführt. D.h. das Programm bleibt stehen, wenn die beschränkte Suche nach N Iterationen keinen Vorzeichenwechsel findet. Es bleibt ebenfalls stehen, wenn bei $\mathbf{x}_N$ der Gradient noch immer zu groß ist. In beiden Fällen wird **Error 1** angezeigt. Durch Drücken von $\blacksquare$ können die beiden unterschieden werden. Sie können das Programm weiterlaufen lassen.

Sie müssen für dieses Programm ein Unterprogramm eingeben, das $f(\mathbf{x})$ und $\nabla f(\mathbf{x})$ berechnet. Dieses Unterprogramm muß das Label «E» haben, die in Matrix **A** abgelegten Werte als Vektor $\mathbf{x}$ verwenden, den Gradienten in Matrix **E** ausgeben und $f(\mathbf{x})$ im X-Register ablegen.

Zusätzlich benötigt das Programm einen Anfangswert $\mathbf{x}_0$ für den gesuchten kritischen Punkt. Dieses Vektor muß in Matrix **A** gespeichert sein.

Das Programm hat folgende Eigenschaften:

- Es sucht irgendeinen Punkt $\mathbf{x}$ mit $\nabla f(\mathbf{x}) = \mathbf{0}$. Es könnte also auch beispielsweise zu einem Sattelpunkt konvergieren. Im allgemeinen müssen Sie auf andere Weise bestimmen, was für ein kritischer Punkt gefunden wurde. (Dieses Programm untersucht ebenfalls nicht näher, ob ein Extremwert am Rand des Abbildungsbereiches von $f(\mathbf{x})$ gefunden wurde.)
- Sie können auch nach dem Beginn des Programmes noch die Konvergenzparameter verändern. Häufig kann dies die Zeit bis zum Eintritt von Konvergenz beträchtlich verkürzen. Es folgen einige Tips:
 - Wenn das Programm mehrfach nach Untersuchen nur eines Punktes u_1 die Intervallverkleinerung beginnt, ist vielleicht die anfängliche Schrittweise zu groß. Verkleinern Sie d , um eine effektive Suche zu erreichen.
 - Wenn die Ergebnisse der beschränkten Suche zunächst vielversprechend aussehen (d.h. der Betrag der Steigung nimmt ab), dann aber ihr Betrag wieder zunimmt, hat die Suche vielleicht einen kritischen Punkt übersprungen. Verkleinern Sie α , um dichtere Stützstellenanzahl zu erhalten; unter Umständen müssen Sie auch N vergrößern.

- In Zeile 102 können Sie **[R/S]** durch **[PSE]** ersetzen oder es ganz weglassen, wenn Sie sich nicht für die Zwischenergebnisse interessieren.
- Für eine Funktion von n Variablen benötigt das Programm $4n+1$ Matrixspeicherplätze.

Tastenfolge**Anzeige****[g]** **[P/R]****[f]** **[CLEAR]** **[PRGM]****[f]** **[LBL]** 8**[RCL]** **[MATRIX]** **[C]****[STO]** **[MATRIX]** **[E]****[RCL]** **[MATRIX]** **[A]****[STO]** **[MATRIX]** **[C]****[RCL]** **[MATRIX]** **[E]****[STO]** **[MATRIX]** **[A]****[g]** **[RTN]****[f]** **[LBL]** 7**[RCL]** 4**[RCL]** **[÷]** 6**[STO]** 8**[GSB]** **[E]****[RCL]** **[MATRIX]** **[E]****[STO]** **[MATRIX]** **[D]****[RCL]** **[MATRIX]** **[D]****[g]** **[F?]** 0**[CHS]****[f]** **[MATRIX]** 8**[g]** **[x=0]****[g]** **[RTN]****[1/x]****[RCL]** **[x]** 8**[STO]** .1

0

[STO] .0**[RCL]** 5**[STO]** 7**[f]** **[LBL]** 6

000–

001–42,21, 8

002–45,16,13

003–44,16,15

004–45,16,11

005–44,16,13

006–45,16,15

007–44,16,11

008– 43 32

009–42,21, 7

010– 45 4

011–45,10, 6

012– 44 8

013– 32 15

014–45,16,15

015–44,16,14

016–45,16,14

017–43, 6, 0

018– 16

019–42,16, 8

020– 43 20

021– 43 32

022– 15

023–45,20, 8

024– 44 .1

025– 0

026– 44 .0

027– 45 5

028– 44 7

029–42,21, 6

Programm-Modus.

Programm zum Vertauschen von **A** und **C** mittels **E**.

Programm zur Liniensuche.

Speichert $d/(j+1)$ in R_8 .

Wechselt für ein Minimum das Vorzeichen des Gradienten.

Berechnet $\|\nabla f(\mathbf{x})\|$.Beendet, falls $\|\nabla f(\mathbf{x})\| = 0$.Berechnet u_1 .Speichert u_1 in $R_{.1}$.Speichert Zähler in R_7 .

Beginn der beschränkten Suche.

Tastenfolge

RCL .1

GSB 3

f PSE

g F? 0

CHS

g TEST 4

GTO 5

GSB 8

RCL .1

STO .0

RCL 2

STO $\times$.1

f DSE 7

GTO 6

RCL MATRIX A

g ABS

GTO 6

f LBL 5

RCL 6

STO 7

f LBL 4

GSB 8

RCL .0

RCL + .1

2

 $\div$

STO 8

GSB 3

g F? 0

CHS

Anzeige

030- 45 .1

031- 32 3

032- 42 31

033-43, 6, 0

034- 16

035-43,30, 4

036- 22 5

037- 32 8

038- 45 .1

039- 44 .0

040- 45 2

041-44,20, .1

042-42, 5, 7

043- 22 6

044-45,16,11

045- 43 16

046- 22 6

047-42,21, 5

048- 45 6

049- 44 7

050-42,21, 4

051- 32 8

052- 45 .0

053-45,40, .1

054-4 , .2

055- 10

056- 44 8

057- 32 3

058-43, 6, 0

059- 16

Zeigt Steigung an.

Untersucht, ob sich die Steigung ändert.

Verzweigung zur Intervallverkleinerung.

Speichert ursprüngliche Matrix wieder in A.

Speichert u_i in $R_{i,0}$.Speichert u_{i+1} in $R_{i,1}$.

Dekrementiert den Zähler.

Verzweigt zur Programmfortsetzung.

Zeigt **Error 1** mit A im X-Register an.

Verzweigt zur Programmfortsetzung.

Programm zur Intervallverkleinerung.

Speichert $j+1$ in R_7 .

Speichert ursprüngliche Matrix wieder in A.

Berechnet die Mitte des Intervalls.

Berechnet die Steigung.

Wechselt das Vorzeichen für Minimum.

Tastenfolge

1
 1
 [STO] [I]

 [R#]
 [g] [TEST] 1
 [f] [DSE] [I]
 [RCL] 8
 [STO] [(i)]

 [f] [DSE] 7
 [GTO] 4
 [g] [RTN]

 [f] [LBL] 3

 [RCL] [MATRIX] [D]
 [f] [RESULT] [C]
 [x]
 [RCL] [MATRIX] [A]
 [+]
 [GSB] 8

 [GSB] [E]
 [STO] 9
 [RCL] [MATRIX] [E]
 [RCL] [MATRIX] [D]
 [f] [RESULT] [B]
 [f] [MATRIX] 5
 1
 [ENTER]
 [RCL] [g] [B]
 [g] [RTN]

 [f] [LBL] [A]
 0
 [STO] 6
 [f] [LBL] 2
 1

Anzeige

060— 1
 061— 1
 062— 44 25

 063— 33
 064—43,30, 1
 065—42, 5,25
 066— 45 8
 067— 44 24

 068—42, 5, 7
 069— 22 4
 070— 43 32

 071—42,21, 3

 072—45,16,14
 073—42,26,13
 074— 20
 075—45,16,11
 076— 40
 077— 32 8

 078— 32 15
 079— 44 9
 080—45,16,15
 081—45,16,14
 082—42,26,12
 083—42,16, 5
 084— 1
 085— 36
 086—45,43,12
 087— 43 32

 088—42,21,11
 089— 0
 090— 44 6
 091—42,21, 2
 092— 1

Speichert Registernummer des Intervalls.

Speichert Intervallmitte in $R_{0,0}$ oder $R_{1,1}$.

Dekrementiert den Zähler.

Beendet Programmteil, wenn Zähler gleich null.

Programm zur Berechnung der Steigung.

Berechnet Punkt $x_j + ts_j$.
 Vertauscht ursprüngliche Matrix und neuen Punkt.
 Berechnet $\nabla f(\mathbf{x})$ in E.
 Speichert $f(\mathbf{x})$ in R_0 .

Berechnet Steigung $(\nabla f)^T s$.

Beendet Programmteil mit Steigung im X-Register.
 Hauptprogramm.

Tastenfolge

Anzeige

[STO] + 6	093–44,40, 6	Speichert $j+1$ in R_6 .
f [SCI] 3	094–42, 8, 3	
[GSB] 7	095– 32 7	Verzweigt zur Liniensuche.
[RCL] 6	096– 45 6	
f [FIX] 0	097–42, 7, 0	
f [PSE]	098– 42 31	Hält an, zeigt $j+1$ an.
f [MATRIX] 1	099–42,16, 1	Setzt zur Anzeige $R_0 = R_1 = 1$.
f [SCI] 3	100–42, 8, 3	
[RCL] 9	101– 45 9	Ruft $f(\mathbf{x})$ zurück.
[R/S]	102– 31	Stoppt Programm.
[RCL] 3	103– 45 3	Ruft e zurück.
[RCL] [MATRIX] [E]	104–45,16,15	
f [MATRIX] 8	105–42,16, 8	Berechnet $\ \nabla f(\mathbf{x})\ $.
g [x≤y]	106– 43 10	Prüft, ob $\ \nabla f(\mathbf{x})\ \leq e$.
[GTO] [B]	107– 22 12	Verzweigung, um die Lösung anzuzeigen.
f [PSE]	108– 42 31	Zeigt $\ \nabla f(\mathbf{x})\ $ an.
[RCL] 5	109– 45 5	
[RCL] 6	110– 45 6	
g [TEST] 8	111–43,30, 8	Prüft, ob $(j+1) < N$.
[GTO] 2	112– 22 2	Sprung zum Schleifenanfang.
[RCL] [MATRIX] [C]	113–45,16,13	
g [ABS]	114– 43 16	Zeigt Error 1 mit C im X-Register an.
[GTO] 2	115– 22 2	Verzweigt zur Programmfortsetzung.
f [LBL] [B]	116–42,21,12	Programm zur Anzeige der Lösung.
g [SF] 9	117–43, 4, 9	Setzt Blinkflag.
[R/S]	118– 31	Beendet Programm und zeigt $\ \nabla f(\mathbf{x}_{j+1})\ $ an.
[GTO] [B]	119– 22 12	Neuer Programmstart.

Verwendete Labels: A, B und 2 bis 8.

Verwendete Register: R_2 bis R_9 , $R_{0,}$ $R_{1,}$ und Indexregister.

Verwendete Matrizen: **A**, **B**, **C**, **D** und **E**.

Ihr Unterprogramm mit dem Label «E» kann alle Label und Register verwenden, die nicht schon oben aufgeführt sind, sowie das Indexregister, Matrix **B** und Matrix **E** (die Ihren berechneten Gradienten enthalten sollte).

Um das Programm zu benutzen:

1. Geben Sie Ihr Unterprogramm in den Programmspeicher ein.
2. Drücken Sie 11 $\boxed{f}$ $\boxed{DIM}$ $\boxed{(i)}$, um die Register R_0 bis R_{11} zu reservieren. (Ihr Unterprogramm benötigt vielleicht weitere Register.)
3. Setzen Sie Flag 0, wenn Sie ein lokales Minimum suchen; löschen Sie Flag 0, wenn Sie ein lokales Maximum suchen.
4. Dimensionieren Sie Matrix **A** auf $n \times 1$, wo n die Anzahl der Variablen ist.
5. Speichern Sie die nötigen Daten:
 - Speichern Sie die Anfangsnäherung x_0 in Matrix **A**.
 - Speichern Sie a in R_2 .
 - Speichern Sie e in R_3 .
 - Speichern Sie d in R_4 .
 - Speichern Sie N in R_5 .
6. Drücken Sie $\boxed{GSB}$ $\boxed{A}$, um während des Iterationsprozesses die Steigungen kontrollieren zu können.
 - Betrachten Sie die Iterationsnummer und den Wert von $f(x)$.
 - Wenn **Error 1** erscheint, drücken Sie $\boxed{\leftarrow}$, um die Anzeige zu löschen. Gehen Sie dann entweder zu Schritt 5 zurück und ändern nötigenfalls die Parameter ab, oder drücken Sie $\boxed{\leftarrow}$ $\boxed{R/S}$, um eine weitere Iteration der beschränkten Suche oder der Optimierung durchzuführen. (Wenn beim Auftreten des Fehlers der Deskriptor der Matrix **A** angezeigt wurde, hatte die Anzahl der Iterationen der beschränkten Suche N überschritten; wenn der Deskriptor der Matrix **C** angezeigt wurde, hatte die Anzahl der Iterationen in der Optimierung N überschritten.)
7. Drücken Sie $\boxed{R/S}$, um die Norm des Gradienten anzeigen zu lassen und um die nächste Iteration zu beginnen.
 - Wenn in der Anzeige die Norm des Gradienten blinkt, drücken Sie $\boxed{\leftarrow}$, und rufen Sie dann die Werte von x in Matrix **A** zurück.

- Wenn die Nummer der Iteration und der Wert von $f(\mathbf{x})$ angezeigt werden, wiederholen Sie diesen Schritt so oft wie nötig oder gehen Sie zurück zu Schritt 5 und ändern Sie die Parameter entsprechend ab.

Beispiel: Finden Sie mit dem Optimierungsprogramm die Maße derjenigen Kiste, die bei einer Summe von Länge und Umfang des Querschnitts von 100 cm maximales Volumen besitzt.

In dieser Aufgabe ist also

$$l + (2h + 2b) = 100$$

$$v = bhl$$

$$v(b, h) = bh(100 - 2h - 2b)$$

$$= 100bh - 2bh^2 - 2hb^2$$

$$\nabla v(b, h) = \begin{bmatrix} 2h(50 - h - 2b) \\ 2b(50 - b - 2h) \end{bmatrix}.$$

Für die Lösung soll $b + h < 50$, $b > 0$ und $h > 0$ gelten.

Geben Sie zunächst ein Unterprogramm ein, das den Gradienten und das Volumen berechnet.

Tastenfolge

```
f LBL E
RCL DIM A
f DIM E
f MATRIX 1
f USER RCL A
f USER
STO .2
STO E
RCL A
STO .3
f MATRIX 1
STO E
+
5
0
-
```

Anzeige

```
120-42,21,15
121-45,23,11
122-42,23,15
123-42,16, 1
124u 45 11
125- 44 .2
126- 44 15
127- 45 11
128- 44 .3
129-42,16, 1
130- 44 15
131- 40
132- 5
133- 0
134- 30
```

Unterprogramm für die Funktion.

Speichert b in R_2 .

Speichert b in e_2 .

Speichert h in R_3 .

Speichert h in e_1 .

Tastenfolge

Anzeige

CHS	135-	16
2	136-	2
x	137-	20
f x$\approx$.2	138-42, 4, .2	
STO x .3	139-44,20, .3	
RCL .2	140-	45 .2
RCL MATRIX E	141-45,16,15	
f RESULT E	142-42,26,15	
x	143-	20
RCL .3	144-	45 .3
RCL + .3	145-45,40, .3	
-	146-	30
RCL .2	147-	45 .2
RCL x .3	148-45,20, .3	
g RTN	149-	43 32

Berechnet $l = 2(50 - h - b)$.Speichert l in R_2 .Speichert bh in R_3 .Ersetzt e_i durch $le_i - 2bh$, die Elemente des Gradienten.Berechnet lbh .

Geben Sie nun die nötigen Daten ein und lassen das Programm laufen.

Tastenfolge

Anzeige

g P/R	
13 f DIM (i)	13.0000
g CF 0	13.0000
f USER	13.0000
f MATRIX 1	13.0000
2 ENTER 1	1
f DIM A	1.0000
15 STO A	15.0000
STO A	15.0000
3 STO 2	3.0000
0.1 STO 3	0.1000
0.05 STO 4	0.0500

Run-Modus.

Reserviert R_0 bis R_3 .

Sucht lokales Maximum.

Aktiviert User-Modus.

Gibt die Dimension in der Matrix **A** ein.Dimensioniert Matrix **A** 2×1 .Speichert Startwert: $l = b = 15$.Speichert $a = 3$.Speichert $e = 0.1$.Speichert $d = 0.05$.

Tastenfolge4 **[STO]** 5**[A]****Anzeige**

4.0000

4.415 04

4.243 04

3.718 04

2.045 04

Error 1**[↔]**

A 2 1 Beschränkte Suche mißlungen!

Speichert $N = 4$.Steigung an u_1 .Steigung an u_2 .Steigung an u_3 .Steigung an u_4 .

Da die Ergebnisse bisher gut aussehen (die Werte der Ableitungen nehmen ab), erlauben Sie bei der beschränkten Suche fünf weitere Stützstellen und setzen Sie für alle weiteren Iterationen $N = 8$.

Tastenfolge5 **[STO]** 78 **[STO]** 5**[R/S]****Anzeige**

5.000 00

8.000 00

-3.849 04

1. $j + 1$.

9.253 03 Volumen nach dieser Iteration.

3.480 01 Gradient.

1.121 03 Steigung an u_1 .9.431 02 Steigung an u_2 .4.126 02 Steigung an u_3 .-1.139 03 Steigung an u_4 (Vorzeichenwechsel).2. $j + 1$.

9.259 03 Volumen nach dieser Iteration.

5.479 -01 Gradient.

-6.127 -01 Steigung an u_1 (Vorzeichenwechsel).3. $j + 1$.

9.259 03 Volumen nach dieser Iteration.

7.726 -02 Gradient ist kleiner als ϵ .

7.726 -02 Hört zu blinken auf.

[R/S]**[↔]****[f]** **[FIX]** 4**[RCL]** **[A]****[RCL]** **[A]**

0.0773

16.6661

16.6661

Ruft h aus a_1 zurück.Ruft b aus a_2 zurück.

Tastenfolge

f **USER**
f **MATRIX** 0

Anzeige

16.6661
16.6661

Gibt Matrixspeicher wieder
frei.

Die gesuchten Maße sind $16.6661 \times 16.6661 \times 33.3355$ Zentimeter. (Eine alternative Methode zur Lösung dieser Aufgabe besteht darin, das lineare Gleichungssystem $\nabla v(h, h) = \mathbf{0}$ zu lösen.)

Genauigkeit numerischer Rechnungen

Mißverständnisse im Zusammenhang mit Fehlern

Das Auftreten von Fehlern ist keine Katastrophe und sollte Sie nicht von numerischen Rechnungen abhalten. Numerische Fehler sind lediglich der Unterschied zwischen dem Ergebnis, das Sie berechnen wollen und dem Ergebnis, das Sie erhalten. Der Unterschied zwischen den beiden spielt nur dann eine Rolle, wenn er zu groß wird. Normalerweise ist er vernachlässigbar klein; es gibt jedoch Fälle, in denen der Fehler beunruhigend groß, schwer zu erklären und noch schwerer zu korrigieren ist. Dieser Anhang behandelt vor allem solche Fehler, die groß werden können – aber selten vorkommen. Zunächst einige Beispiele:

Beispiel 1: Der Rechner funktioniert nicht. Da für $x \geq 0$ immer $(\sqrt{x})^2 = x$ gilt, erwarten wir, daß auch

$$f(x) = (((\underbrace{\sqrt{\sqrt{\dots \sqrt{\sqrt{x}}}}_{50 \text{ Wurzeln}}})^2 \underbrace{\dots}_{50 \text{ Quadrate}}))^2$$

gleich x sein sollte.

Der Ausdruck $f(x)$ kann für beliebige positive x mit einem Programm in 100 Schritten berechnet werden. Falls $x = 10$ ist, erhält der HP-15C jedoch stattdessen 1. Der Fehler $10 - 1 = 9$ erscheint enorm, wo doch nur 100 arithmetische Operationen ausgeführt wurden, von denen jede auf 10 Stellen genau sein sollte. Es stellt sich jedoch heraus, daß das Programm statt $f(x) = x$

$$f(x) = \begin{cases} 1 & \text{für } x \geq 1 \\ 0 & \text{für } 0 \leq x < 1, \end{cases}$$

berechnet, was sehr falsch aussieht. Muß dieser Rechner repariert werden?

Beispiel 2: Viele Pfennige. Karin H. Klüger ist bei einer Firma als wissenschaftlicher Berater bei einer Bezahlung von einem Pfennig pro Sekunde Denken angestellt, wobei ihr ein Jahr lang jeden Tag jede Sekunde ausbezahlt wird. Um sie nicht durch den Lärm der auf ihren Schreibtisch fallenden Pfennige abzulenken, schlägt ihr die Firma vor, die Pfennige für sie auf ein Bankkonto bei einem jährlichen Zinssatz von $11\frac{1}{4}$ Prozent, der jede Sekunde abgerechnet wird, einzuzahlen. Nach einem Jahr werden diese Pfennige zu einer Summe von

$$\text{Summe} = (\text{Bezahlung}) \times \frac{(1 + i/n)^n - 1}{i/n}$$

anwachsen, wo

Bezahlung = 0.01 DM = ein Pfennig pro Sekunde,

$i = 0.1125 = 11.25$ Prozent jährlicher Zinssatz und

$n = 60 \times 60 \times 24 \times 365 =$ Anzahl der Sekunden pro Jahr ist.

Mit ihrem HP-15C berechnet Karin, daß die Summe 376,877.67 DM sein wird. Der Kontostand ist aber nach einem Jahr nur 333,783.35 DM. Steht Karin die Differenz von 43,094.32 DM zu?

In beiden Beispielen sind vermeidbare Rundungsfehler die Ursachen der Diskrepanzen. Wie Sie sie vermeiden können, wird in diesem Anhang erklärt.

Der Kampf gegen die Fehler beginnt mit einer Absage an das Wunschdenken, in dem wir das, was wir erhalten wollen, mit dem, was wir tatsächlich erhalten, verwechseln. Um Verwechslungen zu vermeiden, müssen wir den wahren und den berechneten Ergebnissen verschiedene Namen geben, selbst wenn ihre Differenz so klein ist, daß diese Unterscheidung pedantisch erscheinen mag.

Beispiel 3: Pi. Die Konstante $\pi = 3.1415926535897932384626433\dots$ Wenn Sie die $\boxed{\pi}$ Taste Ihres HP-15C drücken, erhalten Sie einen anderen Wert

$$\boxed{\pi} = 3.141592654,$$

der mit π auf 10 Stellen übereinstimmt. Aber $\boxed{\pi} \neq \pi$, und deshalb sollte es uns nicht überraschen, daß der Rechner im Radiant-Modus nicht $\sin \boxed{\pi} = 0$ erhält.

Es sei unterstellt, daß eine Größe x zu ermitteln ist, aber stattdessen ein Wert X berechnet wird. (Diese Bezeichnungsweise wird im ganzen Anhang verwendet.) Der Fehler ist $x - X$. Der *absolute Fehler* ist $|x - X|$. Der *relative Fehler* wird gewöhnlich für $x \neq 0$ als $(x - X)/x$ definiert.

Beispiel 4: Zu kurze Brücke. Eine Auslegerbrücke besteht aus drei Teilen, deren Längen

$$x = 333.76 \quad y = 195.07 \quad z = 333.76$$

sein sollen. Es stellt sich heraus, daß die gemessenen Längen aber

$$X = 333.69 \quad Y = 195.00 \quad Z = 333.72$$

betragen. Die Abweichung der Gesamtlängen beträgt

$$d = (x + y + z) - (X + Y + Z) = 862.59 - 862.41 = 0.18.$$

Der Ingenieur Frank B. Findig vergleicht die Abweichung d mit der Gesamtlänge $(x + y + z)$ und findet die relative Abweichung

$$d/(x + y + z) = 0.0002 = 0.02\%$$

vernachlässigbar klein. Doch die Nieterin Christine A. Klopff findet die absolute Abweichung $|d| = 0.18$ m viel zu groß; die Brückenteile werden gewaltig gedehnt werden müssen, bevor sie sie zusammennieten kann. Beide sehen dieselbe Abweichung d , doch was für den einen vernachlässigbar klein scheint, kann für den anderen sehr groß sein.

Ob die Fehler groß oder klein sind, sie haben Ursachen, und wenn wir diese verstehen, können wir die meisten Fehler ausgleichen oder sie völlig umgehen. Um die Verzerrungen in den Brückenteilen zu verstehen, müßten wir uns mit Elastizitätstheorie und Bauingenieurwesen beschäftigen. Um die durch die Berechnung selbst hervorgerufenen Fehler zu verstehen, müßten wir uns mit der Funktionsweise und den Grenzen unserer Rechner beschäftigen. Alle diese Einzelheiten wollen wir meistens gar nicht wissen, insbesondere, da die Rundungsfehler eines guten Rechners fast immer minimal sind und daher unbedeutend zu sein scheinen. Wenn sie jedoch gelegentlich zusammenwirken und dann eine Berechnung verfälschen, müssen sie nachträglich doch als «signifikant» eingestuft werden.

Erklärung von Beispiel 1. Sei nun $f(x) = s(r(x))$ mit

$$r(x) = \underbrace{\sqrt{\sqrt{\dots \sqrt{\sqrt{x}}}}}_{50 \text{ Wurzeln}} = x^{(1/2^{50})}$$

und

$$s(r) = ((\dots ((r)^2 \dots)^2)^2 = r^{(2^{50})}.$$

50
Quadrate

Die Werte der Potenzen sind $1/2^{50} = 8.8818 \times 10^{-16}$ und $2^{50} = 1.1259 \times 10^{15}$. x muß aber zwischen 10^{-99} und $9.999\dots \times 10^{99}$ liegen, da keine positiven Zahlen außerhalb dieses Intervalls in den Rechner eingegeben werden können. Da r eine monoton steigende Funktion ist, muß $r(x)$ zwischen

$$r(10^{-99}) = 0.999999999999997975\dots$$

und

$$r(10^{100}) = 1.0000000000002045\dots$$

liegen. Daher sollte der für $r(x)$ berechnete Wert $R(x)$ für alle erlaubten Eingaben in den Rechner gleich 1 sein. Tatsächlich erhalten wir wegen der Rundung aber

$$R(x) = \begin{cases} 0.9999999999 & \text{für } 0 < x < 1 \\ 1.0000000000 & \text{für } 1 \leq x \leq 9.999999999 \times 10^{99} \end{cases}$$

In einem Rechner mit 10stelliger Rechengenauigkeit gilt für $0 < x < 1$ immer $x \leq 0.9999999999$. Daher erwarten wir zu Recht, daß $\sqrt{x} \leq \sqrt{0.9999999999}$ ist, das gleich $0.99999999994999999998\dots$ ist und wiederum auf 0.9999999999 gerundet wird. Daher kann nach beliebig häufigem Drücken der Taste $\sqrt{x}$ für einen Anfangswert $x < 1$ das Ergebnis den Wert 0.9999999999 nicht überschreiten. Damit haben wir eine Erklärung dafür, daß wir oben $R(x) = 0.9999999999$ für $0 < x < 1$ erhalten haben. Wenn $R(x)$ 50 mal quadriert wird, um $F(x) = S(R(x))$ zu erzeugen, dann ist das Ergebnis für $x \geq 1$ tatsächlich gleich 1, doch warum ist für $0 \leq x < 1$ der Wert $F(x) = 0$? Wenn $x < 1$ ist, dann ist

$$s(R(x)) \leq s(0.9999999999) = (1 - 10^{-10})^{2^{50}} \approx 6.14 \times 10^{-48898}.$$

Dieser Wert ist so klein, daß der berechnete Wert $F(x) = S(R(x))$ wegen Underflow gleich 0 wird. Also funktioniert der HP-15C noch; er rechnet so gut, wie man mit 10 Stellen Rechengenauigkeit und zweistelligen Exponenten überhaupt nur rechnen kann.

Wir haben in der Erklärung von Beispiel 1 keine weiteren Informationen über den HP-15C verwendet, außer daß er jede arithmetische Operation $\sqrt{x}$ und x^2 so genau durchführt, wie das mit 10stelliger Rechengenauigkeit und 2stelligen Exponenten nur möglich ist. Alle weiteren Informationen, die wir verwendet haben, waren mathematisches Wissen über die Funktionen f , r , und s . Beispielsweise wurde oben der Wert $r(10^{100})$ als

$$\begin{aligned} r(10^{100}) &= (10^{100})^{(1/2)^{50}} \\ &= \exp(\ln(10^{100})/2^{50}) \\ &= \exp(100(\ln 10)/2^{50}) \\ &= \exp(2.045 \times 10^{-13}) \\ &= 1 + (2.045 \times 10^{-13}) + \frac{1}{2}(2.045 \times 10^{-13})^2 + \dots \end{aligned}$$

berechnet, indem die Reihenentwicklung $\exp(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \dots$ verwendet wurde. Außerdem wurde zur Berechnung von

$$\begin{aligned} \sqrt{0.9999999999} &= (1 - 10^{-10})^{1/2} \\ &= 1 - \frac{1}{2}(10^{-10}) - \frac{1}{8}(10^{-10})^2 - \dots \end{aligned}$$

der binomische Lehrsatz verwendet. Diese mathematischen Überlegungen gehen erheblich weiter, als man vielleicht zur Untersuchung einer Rechnung mit nur ein paar Multiplikationen und Quadratwurzeln erwartet hätte. Damit verdeutlicht Beispiel 1 eine unangenehme Wahrheit: Die Untersuchung einer Berechnung wird durch Fehler erheblich erschwert. Daher erzeugt ein gut konstruierter Rechner wie der HP-15C so wenige Fehler, wie das bei einem vernünftigen Preis nur möglich ist. Viel mehr Fehler würden eine bereits schwere Aufgabe hoffnungslos machen.

Beispiel 1 sollte zwei häufige *Mißverständnisse* beseitigen.

- Eine Berechnung kann durch Rundungsfehler nur dann verfälscht werden, wenn die Rundungsfehler in großer Anzahl auftreten.
- Einige wenige Rundungsfehler können eine Berechnung nur dann verfälschen, wenn gleichzeitig Auslöschung auftritt.

Zum ersten Mißverständnis: Beispiel 1 würde das gleiche exorbitante Verhalten zeigen, wenn es nur einen Rundungsfehler enthielte, denjenigen, der $R(x) = 1$ oder 0.999999999 erzeugt, was ein Fehler von weniger als eins in der letzten (10ten) Stelle ist.

Zum zweiten Mißverständnis: Auslöschung tritt auf, wenn zwei fast gleiche Zahlen voneinander abgezogen werden. Beispielsweise muß man bei der Berechnung von

$$c(x) = (1 - \cos x)/x^2$$

im Radiant-Modus bei kleinen x -Werten vorsichtig sein, da Auslöschung auftritt. Mit $x = 1.2 \times 10^{-5}$ und 10stelliger Rechengenauigkeit erhält man

$$\cos x = 0.9999999999$$

und

$$1 - \cos x = 0.0000000001,$$

wo dann wegen der Auslöschung vielleicht nur eine signifikante Stelle im Nenner verbleibt. Weiterhin gilt

$$x^2 = 1.44 \times 10^{-10}.$$

Damit erhalten wir

$$C(x) = 0.6944.$$

Dieses berechnete Ergebnis ist falsch, da für alle $x \neq 0$ gilt $0 \leq c(x) < 1/2$. Um den numerischen Fehler der Auslöschung zu vermeiden, benutzen wir die trigonometrische Beziehung $\cos x = 1 - 2 \sin^2(x/2)$, um die 1 *genau* wegzukürzen, und erhalten die bessere Gleichung

$$c(x) = \frac{1}{2} \left(\frac{\sin(x/2)}{x/2} \right)^2.$$

Bei der Berechnung dieses Ausdrucks (im Radiant-Modus) an der Stelle $x = 1.2 \times 10^{-5}$ ist das berechnete Ergebnis $C(x) = 0.5$ auf 10 Stellen genau. Dieses Beispiel erklärt den Begriff «Auslöschung» und illustriert die Bedeutung dieses Phänomens, das später in diesem Abschnitt noch detaillierter behandelt wird. Halten Sie sich hier vor Augen, daß im Beispiel 1 keine Subtraktion vorkommt, und daher auch keine Auslöschung, und daß es dennoch durch den Rundungsfehler völlig verfälscht wurde. In dieser Hinsicht ist das Beispiel 1 etwas erschreckend. Wir können dort nirgends eine oder zwei arithmetische Operationen finden, die an dem Unglück schuld sind. Es gibt keine einfache Umformung, die alles rettet, so wie bei $c(x)$. Und leider ist das Beispiel 1 kein Einzelfall. Mit dem Anstieg der Leistungsfähigkeit von Computern und Taschenrechnern treten auch Gelegenheiten für das Auftreten verborgener Fehler und deren Anwachsen häufiger auf.

Dieser Anhang will Ihnen dabei helfen, auf Fehlerfortpflanzungen aufmerksam zu werden und mit ihnen fertig zu werden. Wir beginnen mit den einfachsten Arten von Fehlern und arbeiten uns dann allmählich zu den subtilen Fehlern vor, die die auf Ihrem HP-15C möglichen, komplizierten Berechnungen beeinflussen können.

Eine Fehlerhierarchie

Manche Fehler können einfacher erklärt und toleriert werden als andere. Daher sind die Funktionen, die der HP-15C auf einen einzigen Tastendruck ausführen kann, zwecks einer einfacheren Darstellung danach, wie schwer ihre Fehler abzuschätzen sind, in verschiedene Kategorien eingeteilt worden. Diese Abschätzungen sollten als Ziele, die bei der Entwicklung des Rechners gesetzt wurden, angesehen werden und nicht als technische Daten, die eine bestimmte Genauigkeit garantieren. Die Konstrukteure des Rechners glauben jedoch, mathematisch beweisen zu können, daß sie ihre Ziele erreicht haben, und bisher hat auch sehr umfangreiches Testen kein Anzeichen dafür gegeben, daß sie sich vielleicht geirrt hätten.

Kategorie 0: Gar kein Fehler

Funktionen, die kleine ganze Zahlen (kleiner als 10^{10}) auf ebensolche abbilden, tun dies erwartungsgemäß exakt und ohne Fehler.

Beispiele:

$$\sqrt{4} = 2 \qquad -2^3 = -8 \qquad 3^{20} = 3,486,784,401$$

$$\log(10^9) = 9 \qquad 6! = 720$$

$$\cos^{-1}(0) = 90 \text{ (im Altgrad-Modus)}$$

$$\text{ABS}(4,684,660 + 4,684,659i) = 6,625,109 \text{ (im Komplex-Modus)}$$

Ebenso sind für reelle Argumente **ABS**, **FRAC**, **INT**, **RND** und Vergleiche (wie **x≤y**) exakt. Die Matrixfunktionen **x**, **÷**, **1/x**, **MATRIX** 6 und **MATRIX** 9 (Determinante) bilden jedoch Ausnahmen (siehe Seite 192).

Kategorie ∞ : Overflow/Underflow

Ergebnisse, deren Betrag kleiner als 10^{-99} ist, werden ohne Meldung durch null ersetzt (Underflow). Jedes Ergebnis, das jenseits der Overflowgrenzen $\pm 9.999999999 \times 10^{99}$ liegt, wird durch die nächstgelegene Grenze ersetzt, Flag 9 wird gesetzt, und die Anzeige blinkt. (Drücken von **ON** **ON** oder **CF** 9 oder **↔** löscht den Flag und beendet das Blinken.) Bei den meisten Funktionen, deren Ergebnis mehr als eine Komponente hat, kann in einer Komponente Overflow/Underflow auftreten, ohne die anderen zu verfälschen; beispielsweise **↔R**, **↔P**, komplexe Arithmetik, und die meisten Matrixoperationen. Ausnahmen sind Matrixinversion (**1/x** und **÷**), **MATRIX** 9 (Determinante) und **L.R.**

Kategorie 1: Korrekt oder fast korrekt gerundet

Operationen mit «korrekt gerundeten» Ergebnissen, d.h. deren Fehler in der letzten (10ten) angezeigten Stelle kleiner als $\frac{1}{2}$ ist, sind unter anderem die reellen algebraischen Operationen **+**, **-**, **×**, **÷**, **x²**, **√x**, **1/x** und **%**, die komplexen und Matrixoperationen **+** und **-**, die Matrix- mit Skalaroperationen **×** und **÷** (außer Division durch eine Matrix), und **↔HMS**. Diese Ergebnisse sind das beste, was man mit 10stelliger Genauigkeit erhalten kann, wie z.B. die bekannten Konstanten **π**, **1 e^x**, **2 LN**, **10 LN**, **1 ↔RAD** und viele andere. Operationen, bei denen ein etwas größerer Fehler, aber immer noch erheblich kleiner als eine Einheit der letzten, zehnten Stelle des Ergebnisses, auftreten kann, sind u.a. **Δ%**, **↔H**, **↔RAD**, **↔DEG**, **P_{y,x}** und **C_{y,x}**, **LN**, **LOG**, **10^x** und **TANH** für reelle Argumente; **↔P**, **SIN⁻¹**, **COS⁻¹**, **TAN⁻¹**, **SINH⁻¹**, **COSH⁻¹** und **TANH⁻¹** für reelle und komplexe Argumente; **ABS**, **√x** und **1/x** für komplexe Argumente; die Matrixnormen **MATRIX** 7 und **MATRIX** 8; und schließlich **SIN**, **COS** und **TAN** für reelle Argumente im Altgrad- und Neugrad-Modus (aber nicht im Radiant-Modus – siehe dazu Kategorie 2, Seite 184).

Bei Funktionen, die exponentiell gegen 0 oder ∞ gehen, wenn ihr Argument gegen $\pm \infty$ geht, kann ein Fehler größer als eine Einheit der zehnten Stelle auftreten, jedoch nur, wenn der Funktionswert kleiner als 10^{-20} oder größer als 10^{20} wird. Obwohl der relative Fehler bei extremeren Werten größer wird, bleibt der Fehler stets kleiner als drei Einheiten in der letzten (zehnten) Stelle. Die Ursache für diesen Fehler erklären wir später. Hiervon betroffene Funktionen sind **e^x**, **y^x**, **|x|** (für nichtganzzahlige x), **SINH** und **COSH** für reelle Argumente. Der schlimmste bekannte Fall ist 3^{201} , wo das berechnete Ergebnis $7.968419664 \times 10^{95}$ ist. Die letzte Stelle 4 sollte jedoch stattdessen eine 6 sein, wie bei $7.29^{33.5}$, wo das Ergebnis $7.968419666 \times 10^{28}$ berechnet wird.

Die bisherigen Aussagen über Fehler können für alle Funktionen der Kategorie 1 auf folgende Weise, die sich später als günstig herausstellen wird, zusammengefaßt werden:

Bei der Berechnung einer Funktion f der Kategorie 1 erhält man einen Wert $F = (1 + \varepsilon)f$, dessen relativer Fehler ε zwar unbekannt, aber sehr klein ist:

$$|\varepsilon| < \begin{cases} 5 \times 10^{-10} & \text{wenn } F \text{ korrekt gerundet wird.} \\ 1 \times 10^{-9} & \text{für alle übrigen Funktionen } F \text{ der Kategorie 1.} \end{cases}$$

Diese einfache Charakterisierung aller Funktionen der Kategorie 1 geht nicht auf viele weitere wichtige Eigenschaften, die sie alle besitzen, ein, wie z.B.

- Genau ganzzahlige Funktionswerte: wie in Kategorie 0 erwähnt.
- Vorzeichensymmetrie: $\sinh(-x) = -\sinh(x)$, $\cosh(-x) = \cosh(x)$, $\ln(1/x) = -\ln(x)$ (falls $1/x$ exakt berechnet wurde).
- Monotonie: Für $f(x) \geq f(y)$ gilt $F(x) \geq F(y)$.

Diese weiteren Eigenschaften haben wichtige Folgen; beispielsweise gilt genau $\text{TAN}(20^\circ) = \text{TAN}(200^\circ) = \text{TAN}(2.000^\circ) = \dots = \text{TAN}(2 \times 10^{99}^\circ) = 0.3639702343$. Doch die obige einfache Charakterisierung enthält bereits die am häufigsten benötigten Informationen.

Erklärung von Beispiel 2. Karin wollte

$$\text{Summe} = \text{Auszahlung} \times \frac{(1 + i/n)^n - 1}{i/n}$$

berechnen, wo

Auszahlung = 0.01 DM,

$i = 0.1125$ und

$n = 60 \times 60 \times 24 \times 365 = 31,536,000$ galt.

Sie erhielt auf ihrem HP-15C eine Summe von 376,877.67 DM, doch die Bank erhielt 333,783.35 DM, und dieses Ergebnis stimmt mit dem überein, was gute, moderne Finanzrechner wie der HP-12C, HP-37E, HP-38E/38C und der HP-92 erhalten. Wo war der Fehler in Karins Rechnung? Keine Auslöschung, kein Zusammentreffen vieler Fehler; lediglich ein im Verborgenen wachsender Rundungsfehler war schuld:

$$i/n = 0.000000003567351598$$

$$1 + i/n = 1.000000004$$

bei 10stelliger Genauigkeit! Mit diesem Rundungsfehler erhält Karin bei der Berechnung von $(1 + i/n)^n$ stattdessen $(1.000000004)^{31.536.000} = 1.134445516$, was bereits in der zweiten Stelle falsch ist.

Wie erhält man den richtigen Wert? Die Lösung liegt darin, möglichst viele Stellen von i/n zu erhalten. Da

$$(1 + i/n)^n = e^{n \ln(1 + i/n)},$$

können wir versuchen, den Logarithmus auf eine andere Weise zu berechnen, bei der wir nicht so viele wertvolle Stellen verlieren. Wir können dies auf dem HP-15C einfach tun.

Um $\lambda(x) = \ln(1 + x)$ für alle $x > -1$ exakt zu berechnen, selbst wenn $|x|$ sehr klein ist:

1. Berechnen Sie $u = 1 + x$ gerundet.
2. Dann ist

$$\lambda(x) = \begin{cases} x & \text{für } u = 1 \\ \ln(u)x/(u-1) & \text{für } u \neq 1 \end{cases}$$

Das folgende Programm berechnet $\lambda(x) = \ln(1 + x)$.

Tastenfolge

Anzeige

g P/R		
f CLEAR PRGM	000–	
f LBL A	001–42,21,11	Erwartet x im X-Register.
ENTER	002– 36	
ENTER	003– 36	
EEX	004– 36	Speichert 1 in das X-Register.
+	005– 40	Berechnet $u = 1 + x$ gerundet.
g LN	006– 43 12	Berechnet $\ln(u)$ (null für $u = 1$).
x$\rightarrow$y	007– 34	Speichert x wieder in das X-Register.
g LSTx	008– 43 36	Ruft u zurück.

Tastenfolge

Anzeige

EEX	009– 26	Speichert 1 im X-Register.
g TEST 6	010–43,30, 6	Prüft, ob $u \neq 1$.
–	011– 30	Berechnet $u-1$ für $u \neq 1$.
÷	012– 10	Berechnet $x/(u-1)$ oder $1/1$.
×	013– 20	Berechnet $\lambda(x)$.
g RTN	014– 43 32	
g P/R		

Der vom HP-15C für u berechnete und korrekt gerundete Wert ist $u = (1 + \varepsilon)(1 + x)$, mit $|\varepsilon| < 5 \times 10^{-10}$. Falls $u = 1$ ist, gilt ebenfalls

$$|x| = |1/(1 + \varepsilon) - 1| \leq 5 \times 10^{-10}$$

und wir sehen in diesem Fall aus der Taylor-Entwicklung

$$\lambda(x) = x(1 - \frac{1}{2}x + \frac{1}{3}x^2 - \dots),$$

daß der korrekt gerundete Wert für $\lambda(x)$ gerade x sein muß. Andernfalls wollen wir anstelle von $\lambda(x)$ $x\lambda(u-1)/(u-1)$ genau berechnen. Doch $\lambda(x)/x = 1 - \frac{1}{2}x + \frac{1}{3}x^2 - \dots$ ändert sich nur sehr langsam; so langsam, daß der absolute Fehler $\lambda(x)/x - \lambda(u-1)/(u-1)$ nicht größer als der absolute Fehler $x - (u-1) = -\varepsilon(1+x)$ ist, und falls $x \leq 1$ ist, ist dieser Fehler relativ zu $\lambda(x)/x$ vernachlässigbar. Falls $x > 1$ ist, dann liegt $u-1$ so nahe an x , daß der Fehler wieder vernachlässigbar wird; $\lambda(x)$ ist auf neun Stellen genau.

Wie in Fehleranalysen üblich, ist die Erklärung viel länger als die der Rechnung selbst, und sie verschleiert eine wichtige Tatsache: Wir haben die Fehler in $\ln(u)$ und $u-1$ vernachlässigt, weil wir wußten, daß sie vernachlässigbar sein würden. Diese Tatsache, und daher das einfache Verfahren, *stimmen* auf einigen anderen Taschenrechnern und großen Computern *nicht* ! Es gibt Rechner, die $\ln(u)$ und/oder $1-u$ mit kleinem *absoluten* Fehler berechnen, aber großem *relativen* Fehler für u nahe 1; auf diesen Rechnern sind die obigen Rechnungen komplizierter oder falsch oder oft beides. (Mehr hierzu finden Sie in der Diskussion der Fehler der Kategorie 2.)

Zurück zu Karins Summe. Mit dem obigen einfachen Verfahren zur Berechnung von $\lambda(i/n) = \ln(1 + i/n) = 3.567351591 \times 10^{-9}$ erhält Sie den besseren Wert:

$$(1 + i/n)^n = e^n \lambda(i/n) = 1.119072257$$

aus dem die korrekte Summe folgt.

Um den Fehler in 3^{201} zu verstehen, beachten Sie, daß dieser Wert als $e^{201 \ln(3)} = e^{220.821\dots}$ berechnet wird. Um den gesamten relativen Fehler kleiner als eine Einheit der 10ten Stelle zu halten, müßte $201 \ln(3)$ mit einem erheblich kleineren absoluten Fehler als 10^{-10} berechnet werden, wozu man für dieses Zwischenergebnis mindestens 14 Stellen Genauigkeit bräuchte. Für bestimmte Zwischenrechnungen benutzt der Rechner intern 13 Stellen, doch eine 14te Stelle würde mehr kosten, als sie wert ist.

Kategorie 1C: Kategorie 1 im Komplexen

Die meisten komplexen arithmetischen Funktionen können zwar nicht 9 oder 10 korrekte Stellen sowohl im Real- als auch im Imaginärteil ihres Ergebnisses garantieren, doch gehorcht das Ergebnis den zusammengefassten Aussagen über Funktionen der Kategorie 1, vorausgesetzt, f , F und ε werden als komplexe Zahlen verstanden. Mit anderen Worten liefert jede komplexe Funktion f der Kategorie 1C einen berechneten komplexen Wert $F = (1 + \varepsilon)f$, für dessen kleinen komplexen relativen Fehler ε gilt $|\varepsilon| < 10^{-9}$. Komplexe Funktionen der Kategorie 1C sind $\boxed{\times}$, $\boxed{\div}$, $\boxed{x^2}$, $\boxed{\text{LN}}$, $\boxed{\text{LOG}}$, $\boxed{\text{SIN}^{-1}}$, $\boxed{\text{COS}^{-1}}$, $\boxed{\text{TAN}^{-1}}$, $\boxed{\text{SINH}^{-1}}$, $\boxed{\text{COSH}^{-1}}$ und $\boxed{\text{TANH}^{-1}}$. Daher kann eine Funktion wie $\lambda(z) = \ln(1 + z)$ für alle z mit genau demselben obigen Programm und derselben Erklärung exakt berechnet werden.

Um zu verstehen, warum der Real- und Imaginärteil eines komplexen Ergebnisses vielleicht nicht beide gleichzeitig 9 oder 10 korrekte Stellen besitzen, betrachten wir beispielsweise $\boxed{\times}$: Ideal sollte gelten $(a + ib) \times (c + id) = (ac - bd) + i(ad + bc)$. Versuchen wir dies mit $a = c = 9.999999998$, $b = 9.999999999$ und $d = 9.999999997$; der exakte Wert des Realteils des Produkts $(ac - bd)$ sollte dann

$$\begin{aligned} & (9.999999998)^2 - (9.999999999)(9.999999997) \\ &= 99.999999980000000004 - 99.999999980000000003 \\ &= 10^{-18} \end{aligned}$$

sein, was in der Zwischenrechnung mindestens 20 Stellen benötigt. Der HP-15C verwendet bei Zwischenrechnungen intern 13stellige Rechengenauigkeit und erhält daher für den Realteil 0 anstelle von 10^{-18} , doch ist dieser Fehler im Vergleich zum Imaginärteil 199.999999 vernachlässigbar.

Kategorie 2: Korrekte Rundung bei möglicherweise gestörten (ungenauen) Eingaben

Trigonometrische Funktionen von reellen Winkeln in Radiant

Erinnern wir uns an das Beispiel 3, das zeigte, daß die Taste $\boxed{\pi}$ des Rechners eine Näherung für π mit 10 korrekten Stellen gibt, die sich aber doch etwas von π unterscheidet, so daß $0 = \sin(\pi) \neq \sin(\boxed{\pi})$ ist, wofür der Rechner

$$\boxed{\text{SIN}}(\boxed{\pi}) = -4.100000000 \times 10^{-10}$$

erhält. Dieser berechnete Wert stimmt nicht genau mit dem wahren Wert

$$\sin(\boxed{\pi}) = -4.10206761537356... \times 10^{-10}$$

überein. Die Tatsache, ob die Diskrepanz klein (absoluter Fehler weniger als 2.1×10^{-13}) oder relativ groß (in der vierten Stelle bereits falsch) für einen Rechner mit 10stelliger Rechengenauigkeit aussieht, bedarf einer Untersuchung, denn sie zieht andere Fehler nach sich, die auf den ersten Blick viel schlimmer aussehen.

Betrachten wir

$$10^{14}\pi = 314159265358979.3238462643...$$

mit $\sin(10^{14}\pi) = 0$ und

$$10^{14} \times \boxed{\pi} = 314159265400000$$

mit $\boxed{\text{SIN}}(10^{14} \boxed{\pi}) = 0.7990550814$, obwohl der exakte Wert

$$\sin(10^{14} \boxed{\pi}) = -0.78387... \text{ ist.}$$

Der Vorzeichenfehler darf nicht einfach übergangen werden; er scheint anzuzeigen, daß der Rechner nicht richtig funktioniert.

Um den Fehler in trigonometrischen Funktionen zu verstehen, müssen wir die kleinen Unterschiede zwischen π und zwei Näherungen für π beachten:

Wahres $\pi = 3.1415926535897932384626433...$

Taste $\boxed{\pi} = 3.141592654$ (stimmt mit π auf 10 Stellen überein)

Internes $p = 3.141592653590$ (stimmt mit π auf 13 Stellen überein)

Damit erklärt die folgende Formel für den berechneten Wert alles: $\boxed{\text{SIN}}(x) = \sin(x\pi/p)$ bis auf ± 0.6 Einheiten der letzten (10ten) Stelle.

Allgemeiner, sei $\text{trig}(x)$ eine der Funktionen $\sin(x)$, $\cos(x)$ oder $\tan(x)$ im reellen Radiant-Modus, dann berechnet der HP-15C

$$\boxed{\text{TRIG}}(x) = \text{trig}(x\pi/p)$$

auf ± 0.6 Einheiten der 10ten Stelle.

Diese Formel hat wichtige praktische Konsequenzen:

- Da $\pi/p = 1 - 2.0676... \times 10^{-13}/p = 0.999999999999342...$, unterscheiden sich die Werte $\boxed{\text{TRIG}}(x)$ und $\text{trig}(x)$ höchstens um die Auswirkungen zweier Störungen: einer in der 10ten Stelle der Ausgabe von $\text{trig}(x)$ und einer in der 13ten Stelle der Eingabe x .

Wenn x mit 10stelliger Genauigkeit berechnet und dann gerundet wird, ist der Fehler in der 10ten Stelle sicher um Zehnerpotenzen größer als die zweite Störung von $\boxed{\text{TRIG}}$ in der 13ten Stelle von x , so daß diese zweite Störung vernachlässigt werden kann, es sei denn, wir setzen x als bekannt oder exakt berechnet voraus.

- Jede trigonometrische Identität, die nicht explizit π enthält, wird bis auf einen Rundungsfehler in der 10ten Stelle der in der Identität berechneten Werte erfüllt. Beispielsweise gilt

$$\sin^2(x) + \cos^2(x) = 1, \text{ daher } (\boxed{\text{SIN}}(x))^2 + (\boxed{\text{COS}}(x))^2 = 1$$

$$\sin(x)/\cos(x) = \tan(x), \text{ daher } \boxed{\text{SIN}}(x)/\boxed{\text{COS}}(x) = \boxed{\text{TAN}}(x)$$

wobei die berechneten Ergebnisse für alle x bis auf neun Stellen übereinstimmen. Beachten Sie, daß $\boxed{\text{COS}}(x)$ für keinen Wert x , der mit nur 10 Stellen genau dargestellt werden kann, gleich null wird. Und wenn aus gegebenem x der Wert $2x$ genau berechnet werden kann, stimmt

$$\sin(2x) = 2 \sin(x)\cos(x), \text{ so } \boxed{\text{SIN}}(2x) = 2 \boxed{\text{SIN}}(x)\boxed{\text{COS}}(x)$$

auf neun Stellen. Überprüfen Sie die letzte Beziehung mit $x = 52174$ Radiant auf Ihrem HP-15C:

$$\boxed{\text{SIN}}(2x) = -0.00001100815000,$$

$$2 \boxed{\text{SIN}}(x)\boxed{\text{COS}}(x) = -0.00001100815000.$$

Beachten Sie die gute Übereinstimmung, obwohl für dieses x $\sin(2x) = 2 \sin(x)\cos(x) = -0.0000110150176...$ sich von $\boxed{\text{SIN}}(2x)$ bereits in der vierten Stelle unterscheidet. Dieselben Beziehungen werden auch von $\boxed{\text{TRIG}}(x)$ Werten und von $\text{trig}(x)$ Werten erfüllt, obwohl sich $\boxed{\text{TRIG}}(x)$ und $\text{trig}(x)$ unterscheiden können.

- Trotz der zwei Fehlertypen in $\boxed{\text{TRIG}}$, erfüllen die berechneten Werte alle bekannten Beziehungen:

- Vorzeichensymmetrie:

$$\boxed{\text{COS}}(-x) = \boxed{\text{COS}}(x)$$

$$\boxed{\text{SIN}}(-x) = -\boxed{\text{SIN}}(x)$$

- Monotonie: für $\text{trig}(x) \geq \text{trig}(y)$,
gilt $\boxed{\text{TRIG}}(x) \geq \boxed{\text{TRIG}}(y)$
(solange $|x - y| < 3$)
- Begrenzende Ungleichheiten: $\boxed{\text{SIN}}(x)/x \leq 1$ für alle $x \neq 0$
 $\boxed{\text{TAN}}(x)/x \geq 1$ für $0 < |x| < \pi/2$
 $-1 \leq \boxed{\text{SIN}}(x)$ und $\boxed{\text{COS}}(x) \leq 1$
für alle x .

Welche Auswirkungen haben diese Eigenschaften auf technische Berechnungen? *Sie können diese vergessen!*

Technische Berechnungen werden im allgemeinen von dem Unterschied zwischen p und π nicht betroffen, da die Auswirkungen dieses Unterschieds in der Formel, die oben $\boxed{\text{TRIG}}(x)$ definiert, vom Unterschied zwischen $\boxed{\pi}$ und π sowie gewöhnlichem, unvermeidlichem Runden in x oder in $\text{trig}(x)$ überdeckt werden. Für den Ingenieur kann der Quotient $\pi/p = 0.9999999999999342\dots$ ohne merkliche Auswirkung auf das Verhalten von $\boxed{\text{TRIG}}$ durch 1 ersetzt werden.

Beispiel 5: Mondphasen. Wenn wir den genauen Abstand der Erde zum Mond kennen würden, könnten wir den Phasenunterschied zwischen Radarsignalen berechnen, wenn sie zum Mond gesendet und von ihm reflektiert werden. In dieser Rechnung macht die durch $p \neq \pi$ verursachte Änderung der Phase weniger aus, als die Änderung des Abstandes der Erde zum Mond um nur die Dicke dieser Seite. Weiterhin werden die Berechnungen der Intensität, Richtung und Änderung der Radiosignale in der Nähe des Mondes oder der reflektierten Signale in der Nähe der Erde, deren Gültigkeit darauf beruht, daß die trigonometrischen Beziehungen weiterhin gelten, von der Tatsache, daß $p \neq \pi$ ist, nicht betroffen. Sie beruhen vielmehr darauf, daß p eine Konstante ist (die in der Formel für $\boxed{\text{TRIG}}(x)$ nicht von x abhängt); und diese Konstante ist sehr nahe an π .

Die p verwendenden Tastenfunktionen des HP-15C sind die trigonometrischen Funktionen $\boxed{\text{SIN}}$, $\boxed{\text{COS}}$ und $\boxed{\text{TAN}}$ für reelle und komplexe Argumente; die hyperbolischen Funktionen $\boxed{\text{SINH}}$, $\boxed{\text{COSH}}$ und $\boxed{\text{TANH}}$ für komplexe Argumente; die komplexen Operationen $\boxed{e^x}$, $\boxed{10^x}$ und $\boxed{y^x}$; und reelles und komplexes $\boxed{\leftrightarrow R}$.

All dies sieht aus wie viel Lärm um nichts. Nach einer Überschwemmung mit Formeln und Beispielen kommen wir zu dem Schluß, daß für die Zwecke des Ingenieurs der durch $p \neq \pi$ hervorgerufene Fehler vernachlässigt werden kann, also hätten wir uns gar nicht um ihn zu kümmern brauchen. Mit diesem Problem muß der gewissenhafte Fehleranalysator leben; wenn er von vornherein davon ausginge, daß man kleine Fehler vernachlässigen kann, könnte er sich irren.

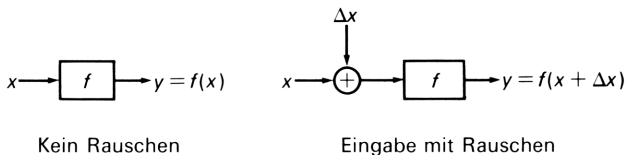
Rückwärts-Fehleranalyse

Bis zum Ende der fünfziger Jahre lebten die meisten Computerspezialisten in Angst vor dem Schaden, den Rundungsfehler in numerischen Berechnungen anrichten. Sie begründeten ihre Angst auf veröffentlichte Fehleranalysen wie die eines berühmten Wissenschaftlers, der den Schluß zog, 40×40 Matrizen seien schon viel zu groß, um numerisch invertiert werden zu können, da Rundungsfehler dies unmöglich machen würden. In der Mitte der sechziger Jahre wurden jedoch schon routinemäßig 100×100 Matrizen invertiert, und heutzutage werden auf der ganzen Welt in geodätischen Berechnungen Gleichungen mit Hunderttausenden von Variablen gelöst. Wie ist dieser Fortschritt möglich, wo doch die mathematische Analyse jenes berühmten Wissenschaftlers völlig korrekt war?

Heute wissen wir mehr als damals darüber Bescheid, warum verschiedene Formeln zur Berechnung desselben Resultats in ihrer Anfälligkeit für Rundungsfehler vollkommen verschieden sein können. Wir wissen beispielsweise, warum die zu gewissen Kleinsten Quadrate Aufgaben gehörigen Normalgleichungen nur mit Arithmetik extrem hoher Rechengenauigkeit gelöst werden können. (Das hatte der berühmte Wissenschaftler eigentlich bewiesen.) Wir kennen auch neue Verfahren, eines ist auf Seite 140 beschrieben, die dieselben Kleinsten Quadrate Aufgaben mit wenig mehr Rechengenauigkeit, als zur Darstellung der Daten selbst erforderlich ist, lösen können. Diese neuen und besseren numerischen Verfahren sind nicht trivial und wären vielleicht nie erfunden worden, wenn es nicht zuerst neue und bessere Techniken der Fehleranalyse gegeben hätte, mit denen wir gelernt haben, Formeln, die für Rundungsfehler überempfindlich sind, von anderen, die es nicht sind, zu unterscheiden. Eine der neuen Techniken (von 1957) wird heute als «Rückwärts-Fehleranalyse» bezeichnet, und Sie haben sie bereits zweimal in Aktion gesehen: Das erste Mal konnten wir damit erklären, warum das Verfahren zur Berechnung von $\lambda(x)$ genau genug ist, um den Fehler in Beispiel 2 zu beseitigen; das zweite Mal konnten wir damit erklären, warum die **TRIG** Funktionen des Rechners mit sehr hoher Genauigkeit selbst bei sehr großen Radianten x , bei denen **TRIG**(x) und $\text{trig}(x)$ sehr verschieden sein können, dieselben Beziehungen erfüllen wie die trigonometrischen Funktionen selber. Die folgenden Abschnitte erklären nun die Rückwärts-Fehleranalyse selbst in etwas allgemeinerer Form.

Betrachten wir ein allgemeines System F , das eine Eingabe x in eine Ausgabe $y = f(x)$ transformieren soll. F könnte beispielsweise ein Signalverstärker, ein Filter, ein Wandler, ein Kontrollsystem, eine Raffinerie, das Wirtschaftssystem eines Staats, ein Computerprogramm oder ein Taschenrechner sein. Die Ein- und Ausgabe brauchen keine Zahlen zu sein, sie könnten Mengen von Zahlen oder Matrizen oder irgendeine andere quantitative Größe sein. Wenn die

Eingabe x durch Rauschen Δx gestört ist, dann ist im allgemeinen auch die Ausgabe $y + \Delta y = f(x + \Delta x)$ durch Rauschen $\Delta y = f(x + \Delta x) - f(x)$ gestört.

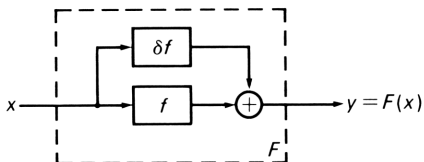


Manche Transformationen f sind gegenüber Rauschen in der Eingabe stabil; Δy bleibt relativ klein, solange Δx relativ klein ist. Andere Transformationen f können gegenüber Rauschen instabil sein, da schon gewisse relativ kleine Störungen Δx in der Eingabe relativ große Störungen Δy in der Ausgabe hervorrufen. Im allgemeinen wird das Eingaberauschen Δx auf irgendeine Art von der Transformation f auf dem Weg zum Ausgaberauschen Δy gefärbt werden, und eine Verkleinerung von Δy ist nur über ein Verkleinern von Δx oder ein Ändern von f möglich. Wenn wir f als ein Betriebsdatum oder als Entwurfsziel akzeptiert haben, müssen wir uns mit der Weise, in der f Eingaberauschen färbt, abfinden.

Das tatsächliche System F unterscheidet sich von dem gewünschten f durch Rauschen oder andere Störungen innerhalb von F . Bevor wir uns mit den Konsequenzen dieses internen Rauschens beschäftigen können, müssen wir eine Schreibweise oder Konvention dafür festsetzen. Die einfachste ist

$$F(x) = (f + \delta f)(x)$$

wobei die Störung δf das interne Rauschen in F bezeichnet.



Eine kleine Ausgabestörung (Kategorie 1)

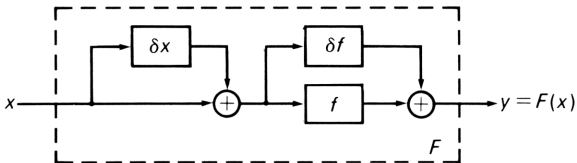
Wir hoffen, der Störungsterm δf ist gegen f vernachlässigbar. Wenn sich diese Hoffnung erfüllt, stuften wir f zu Darstellungszwecken in Kategorie I ein; das bedeutet, daß das Rauschen innerhalb von F als ein kleiner Zusatzterm δf zu der gewünschten Ausgabe f beschrieben werden kann.

Beispielsweise wird $F(x) = \boxed{\text{LN}}(x)$ in Kategorie 1 eingestuft, da sich die vielen kleinen Fehler, die der HP-15C bei seiner Berechnung von $F(x) = (f + \delta f)(x)$ macht, zu einer Gesamtstörung $\delta f(x)$, die kleiner als 0.6 in der letzten (10ten) Stelle der gewünschten Ausgabe $f(x) = \ln(x)$ ist, addieren. Doch für Radiant x ist $F(x) = \boxed{\text{SIN}}(x)$ nicht in Kategorie 1, da sich $F(x)$ zu sehr von $f(x) = \sin(x)$ unterscheiden kann; z.B. hat $F(10^{14} \pi) = 0.799\dots$ ein anderes Vorzeichen als $f(10^{14} \pi) = -0.784\dots$, so daß die Gleichung $F(x) = (f + \delta f)(x)$ nur dann erfüllt sein kann, wenn δf manchmal erheblich größer als f wird (was nicht gut aussieht).

Realistische Systeme ähneln meist mehr $\boxed{\text{SIN}}$ als $\boxed{\text{LN}}$. In den meisten realistischen Systemen kann sich das Rauschen gelegentlich so anhäufen, daß es die gewünschte Ausgabe zumindest für manche Eingaben völlig überdecken kann, und doch sind solche Systeme nicht notwendigerweise zu verdammen. Viele realistische Systeme F funktionieren zuverlässig, da ihr internes Rauschen, obwohl es manchmal groß ist, nie sehr viel mehr Schaden anrichtet, als eine vernachlässigbar kleine Störung des Eingabesignals x . Solche Systeme lassen sich als

$$F(x) = (f + \delta f)(x + \delta x)$$

darstellen, wo δf stets klein im Vergleich zu f und δx stets kleiner als oder vergleichbar zu dem Rauschen Δx ist, dessen Störung von x bekannt und berücksichtigt ist. Die zwei Störterme δf und δx sind hypothetische Störungen, die wir einführen, um die Vielfalt von realen, tatsächlich überall in F verteilten Störungen zu beschreiben. Ein Teil der Störungen scheint eine vernachlässigbar kleine Störung Δx der Eingabe zu sein – daher der Name «Rückwärts-Fehleranalyse». Ein solches System F , dessen Rauschen durch zwei vernachlässigbar kleine Störungen beschrieben werden kann, wird daher zu Darstellungszwecken in Kategorie 2 eingestuft.



Kleine Störungen der Ein- und Ausgabe (Kategorie 2)

Zunächst wird der an lineare Systeme und kleine Signalgrößen gewöhnte Leser keinen Unterschied zwischen Kategorie 1 und Kategorie 2 feststellen können, da die Störungen bei solchen Systemen gleichermaßen der Ein- oder der Ausgabe zugeordnet werden können. Andere, allgemeinere Systeme jedoch, die digital oder nichtlinear sein können, gestatten keine beliebige Änderung der Zuordnung von Eingaberauschen zu Ausgaberauschen oder umgekehrt.

Kann beispielsweise der gesamte Fehler in $\boxed{\cos}$ einfach einer im Vergleich zur Eingabe x kleinen Eingabestörung δx zugeordnet werden, indem man $\boxed{\cos}(x) = \cos(x + \delta x)$ schreibt? Nicht für sehr kleine x . Wenn x beispielsweise 10^{-5} Radiant annähert, dann ist $\cos(x)$ sehr nahe an 0.99999999995 und muß dann entweder auf $1 = \cos(0)$ oder auf $0.9999999999 = \cos(1.414... \times 10^{-5})$ gerundet werden. Daher stimmt $\boxed{\cos}(x) = \cos(x + \delta x)$ nur, wenn δx relativ groß sein darf, etwa in der Größenordnung von x , wenn x klein ist. Wenn wir den Fehler in $\boxed{\cos}$ mit nur relativ kleinen Störungen erklären wollen, benötigen wir mindestens zwei: einmal eine Störung $\delta x = (-6.58... \times 10^{-14})x$, die kleiner als der Rundungsfehler der Eingabe ist, und dann eine weitere in der Ausgabe in der Größenordnung des dortigen Rundungsfehlers; $\boxed{\cos}(x) = (\cos + \delta \cos)(x + \delta x)$ mit einem unbekannten $|\delta \cos| \leq (6 \times 10^{-10})|\cos|$.

Jedes System der Kategorie 2 wird wie $\boxed{\cos}$ durch nur zwei kleine Toleranzen – nennen wir sie ε und η – charakterisiert, die all das beschreiben, was Sie über die internen Störungen des Systems wissen müssen. Die Toleranz ε enthält eine hypothetische Ausgabestörung $|\delta f| \leq \varepsilon|f|$ und η enthält eine hypothetische Eingabestörung $|\delta x| \leq \eta|x|$, die in einer einfachen Formel wie

$$F(x) = (f + \delta f)(x + \delta x)$$

mit $|\delta f| \leq \varepsilon|f|$ und $|\delta x| \leq \eta|x|$

auftreten können. Ziel der Rückwärts-Fehleranalyse ist, sicherzustellen, daß das gesamte interne Rauschen von F tatsächlich von solch einer einfachen Formel mit befriedigend kleinen Toleranzen ε und η beschrieben werden kann. Im günstigsten Fall stellt die Rückwärts-Fehleranalyse sicher, daß *der tatsächliche Wert $F(x)$ sich kaum von dem idealen Wert $f(x + \delta x)$ unterscheidet, der von einer Eingabe $x + \delta x$ erzeugt worden wäre, die sich kaum von der eigentlichen Eingabe x unterscheidet*, und gibt dem Wort «kaum» eine quantitative Bedeutung (ε und η). Die Rückwärts-Fehleranalyse gelingt jedoch nur für Systeme F , die sehr sorgfältig entworfen worden sind, um sicherzustellen, daß jede interne Störung schlimmstenfalls zu einer erträglich kleinen Ein- oder Ausgabestörung äquivalent ist. Erste Entwürfe von Systemen, besonders von Programmen zur Ausführung von numerischen Berechnungen, leiden oft auf kompliziertere und unangenehmere Weise unter internen Störungen, wie das folgende Beispiel veranschaulicht.

Beispiel 6: Die kleinere Nullstelle einer quadratischen Gleichung. Die beiden Nullstellen x und y der quadratischen Gleichung $c - 2bz + az^2 = 0$ sind immer dann reell, wenn $d = b^2 - ac$ positiv ist. Dann kann man die Nullstelle y mit dem kleineren Betrag als eine Funktion $y = f(a, b, c)$ der Koeffizienten der quadratischen Gleichung auffassen:

$$f(a, b, c) = \begin{cases} (b - \sqrt{d} \operatorname{sgn}(b))/a & \text{für } a \neq 0 \\ (c/b)/2 & \text{sonst.} \end{cases}$$

Wenn man diese Gleichung direkt in ein Programm $F(a, b, c)$ umsetzen würde, das $f(a, b, c)$ berechnen soll, dann würde dieses Programm immer dann, wenn ac soviel kleiner als b^2 wird, daß der berechnete Wert d auf b^2 gerundet wird, $F = 0$ ergeben, obwohl $f \neq 0$. Ein so drastischer Fehler kann von der Rückwärts-Fehleranalyse nicht erklärt werden, da keine kleinen Störungen aller Koeffizienten a, b und c den Wert c nullsetzen können, was nötig wäre, um die kleinere Nullstelle null werden zu lassen. Andererseits läßt sich die algebraisch äquivalente Gleichung

$$f(a, b, c) = \begin{cases} c/(b + \sqrt{d} \operatorname{sgn}(b)) & \text{für Nenner ungleich null} \\ 0 & \text{sonst} \end{cases}$$

in ein viel besseres Programm F umsetzen, dessen Störungen nicht mehr Schaden anrichten als eine Störung in der letzten (10ten) Stelle von c . Ein solches Programm wird später (auf Seite 205) gelistet und muß in all den (in Anwendungen häufig vorkommenden) Fällen verwendet werden, in denen die kleinere Nullstelle y exakt berechnet werden soll, obwohl die andere, nicht benötigte Nullstelle der quadratischen Gleichung relativ groß ist.

Fast alle in den HP-15C eingebauten Funktionen sind so entworfen worden, daß eine Rückwärts-Fehleranalyse ihre Fehler befriedigend erklären kann. Ausnahmen sind $\boxed{\text{SOLVE}}$, $\boxed{\int}$ und die Statistikfunktionstasten $\boxed{\text{s}}$, $\boxed{\text{L.R.}}$ und $\boxed{\text{f.r.}}$, die in gewissen pathologischen Fällen falsche Resultate liefern können. Alle anderen Funktionen F zur Berechnung eines Funktionswerts $f(x)$ liefern stattdessen einen Wert $F(x)$, der sich nicht mehr von $f(x)$ unterscheidet, als wenn zunächst x mit $|\delta x| \leq \eta|x|$ zu $x + \delta x$ gestört worden wäre und dann $f(x + \delta x)$ mit $|\delta f| \leq \varepsilon|f|$ zu $(f + \delta f)(x + \delta x)$ gestört würde. Die Toleranzen η und ε haben von Funktion zu Funktion etwas verschiedene Werte; es gilt grob

$$\eta = 0 \text{ und } \varepsilon < 10^{-9} \quad \text{für alle Funktionen der Kategorie 1,}$$

$$\eta < 10^{-12} \text{ und } \varepsilon < 6 \times 10^{-10} \quad \text{für andere reelle und komplexe Funktionen.}$$

Bei Matrixoperationen müssen die Beträge $|\delta x|$, $|x|$, $|\delta f|$ und $|f|$ durch die Matrixnormen $\|\delta \mathbf{x}\|$, $\|\mathbf{x}\|$, $\|\delta \mathbf{f}\|$ und $\|\mathbf{f}\|$ ersetzt werden, die im Abschnitt 4 erklärt und mit **MATRIX** 7 oder **MATRIX** 8 berechnet werden. Dann fallen alle Matrixfunktionen, die sich nicht in Kategorie 1 befinden, in Kategorie 2, wobei grob gilt

$\eta \leq 10^{-12}n$ und $\varepsilon < 10^{-9}$ für Matrixoperationen (außer Determinante **MATRIX** 9, $\oplus$ und $\oplus/x$)

$\eta \leq 10^{-9}n$ und $\varepsilon < 10^{-9}$ für Determinante **MATRIX** 9, $\oplus/x$ und $\oplus$ mit einem Matrixdivisor

wo n die größte Dimension aller in der Operation vorkommender Matrizen ist.

Die Schlußfolgerungen aus einer erfolgreichen Rückwärts-Fehleranalyse sehen nur dann einfach aus, wenn die Eingabedaten x durch unvermeidliche und unzusammenhängende Störungen Δx gestört sind, was häufig der Fall ist. Wenn wir dann $f(x)$ berechnen wollen, ist das beste Ergebnis, das wir erwarten dürfen $f(x + \Delta x)$, doch tatsächlich erhalten wir $F(x + \Delta x) = (f + \delta f)(x + \Delta x + \delta x)$, wo $|\delta f| \leq \varepsilon/|f|$ und $|\delta x| \leq \eta|x|$ gilt.

Das Ergebnis, das wir erhalten, ist selten schlechter als das Beste, das wir erwarten durften, vorausgesetzt, daß die Toleranzen ε und η klein genug sind, besonders falls $|\Delta x|$ wenigstens ungefähr so groß wie $\eta|x|$ ist. Natürlich kann auch das Beste, das wir erwarten dürfen, sehr schlecht sein, besonders dann, wenn f eine Singularität besitzt, die näher an x liegt, als die Toleranzen für die Störungen Δx und δx von x sind.

Rückwärts-Fehleranalyse und Singularitäten

Das Wort «Singularität» bezeichnet sowohl einen speziellen Wert des Arguments x als auch die Weise, in der sich $f(x)$ bei Annäherung von x an diesen besonderen Wert benimmt. Meistens werden $f(x)$ oder die erste Ableitung $f'(x)$ unendlich groß, oder sie oszillieren wild, wenn x sich der Singularität nähert. Manchmal bezeichnet man auch die Singularitäten von $\ln|f|$ als Singularitäten von f , wobei die Nullstellen von f zu den eigentlichen Singularitäten hinzuge-setzt werden; was, wie wir sehen werden, sinnvoll ist, wenn es um die relative Genauigkeit einer Berechnung von f geht. Für unsere Zwecke kann der Begriff «Singularität» etwas vage gelassen werden.

Normalerweise wird man versuchen, Singularitäten zu vermeiden oder zu neutralisieren. Beispielsweise hat die Funktion

$$c(x) = \begin{cases} (1 - \cos x)/x^2 & \text{für } x \neq 0 \\ 1/2 & \text{sonst} \end{cases}$$

an der Stelle $x = 0$ keine Singularität, obwohl sich ihre Bestandteile $1 - \cos x$ und x^2 (eigentlich deren Logarithmen) singular verhalten, wenn x gegen 0 geht. Die Singularitäten der Bestandteile machen dem Programm, das $c(x)$ berechnet, Schwierigkeiten. Die meisten werden durch die Wahl einer besseren Gleichung gelöst:

$$c(x) = \begin{cases} \frac{1}{2} \left(\frac{\sin(x/2)}{x/2} \right)^2 & \text{für } x/2 \neq 0 \\ 1/2 & \text{sonst.} \end{cases}$$

Nun könnte die Singularität völlig umgangen werden, wenn das Programm, das $c(x)$ berechnet, prüft, ob $x/2 = 0$ ist.

Die Rückwärts-Fehleranalyse kompliziert Singularitäten auf eine Weise, die am besten am Beispiel der Funktion $\lambda(x) = \ln(1+x)$ illustriert werden kann, mit der wir die Zinsaufgabe im Beispiel 2 gelöst hatten. Das dort verwendete Verfahren berechnete $u = 1+x$ (gerundet) $= 1+x + \Delta x$. Dann gilt

$$\lambda(x) = \begin{cases} x & \text{für } u = 1 \\ \ln(u) x / (u - 1) & \text{sonst.} \end{cases}$$

Dieses Verfahren benutzt die Tatsache, daß $\lambda(x)/x$ an der Stelle $x=0$ eine hebbare Singularität besitzt, was bedeutet, daß $\lambda(x)/x$ sich kontinuierlich ändert und für x gegen 0 gegen 1 geht. Daher wird für $|\Delta x| < 10^{-9} \lambda(x)/x$ relativ gut durch $\lambda(x + \Delta x)/(x + \Delta x)$ angenähert, und daher wird

$$\lambda(x) = x(\lambda(x)/x) \approx x(\lambda(x + \Delta x)/(x + \Delta x)) = x(\ln(u)/(u-1))$$

genau berechnet, da $\boxed{\text{LN}}$ eine Funktion der Kategorie 1 ist. Was würde passieren, wenn $\boxed{\text{LN}}$ stattdessen in Kategorie 2 wäre?

Wenn $\boxed{\text{LN}}$ in Kategorie 2 wäre, dann würde eine «erfolgreiche» Rückwärts-Fehleranalyse für Argumente u nahe 1 zeigen, daß $\boxed{\text{LN}}(u) = \ln(u + \delta u)$ mit $|\delta u| < 10^{-9}$ wäre. Dann würde das obige Verfahren nicht $x(\ln(u)/(u-1))$ sondern

$$\begin{aligned} x(\ln(u + \delta u)/(u - 1)) &= x\lambda(x + \Delta x + \delta u)/(x + \Delta x) \\ &= x(\lambda(x + \Delta x + \delta u)/(x + \Delta x + \delta u)) \frac{x + \Delta x + \delta u}{x + \Delta x} \\ &\approx x(\lambda(x)/x)(1 + \delta u/(x + \Delta x)) \\ &= \lambda(x)(1 + \delta u/(x + \Delta x)) \end{aligned}$$

ergeben. Wenn $|x + \Delta x|$ nicht viel größer als 10^{-9} ist, kann der letzte Ausdruck völlig verschieden von $\lambda(x)$ sein. Daher würde das Verfahren, mit dem wir Beispiel 2 gelöst haben, auf Rechnern, deren $\boxed{\text{LN}}$ nicht in Kategorie 1 ist, nicht funktionieren. Es *gibt* solche Rechner, und auf ihnen bricht dieses Verfahren für gewisse, ansonsten harmlos aussehende Eingaben, zusammen. Ähnliche Probleme treten auch auf Rechnern auf, die $(u + \delta'u) - 1$ statt $u - 1$ berechnen, da ihr $\boxed{-}$ in Kategorie 2 statt in Kategorie 1 liegt. Und diejenigen Rechner, die $\ln(u + \delta u)/(u + \delta'u - 1)$ statt $\ln(u)/(u - 1)$ berechnen, da sowohl ihr $\boxed{\text{LN}}$ als auch ihr $\boxed{-}$ in Kategorie 2 liegen, wären doppelt anfällig, gäbe es nicht einen bisher schlecht verstandenen Zufall, der im allgemeinen die beiden Fehler δu und $\delta'u$ so korreliert, daß nur die Hälfte aller signifikanten Stellen des berechneten λ (anstelle aller Stellen) verfälscht ist.

Erste Zusammenfassung der Rückwärtsanalyse

Nachdem wir die Komplexität der Rückwärts-Fehleranalyse bei Singularitäten demonstriert haben, ist es jetzt an der Zeit, das bis hier Besprochene zusammenzufassen, zu vereinfachen und zu konsolidieren.

- Die Ergebnisse vieler numerischer Algorithmen sind zu falsch, um durch irgendeine befriedigende Fehleranalyse erklärt zu werden.
- Die Ergebnisse gewisser numerischer Verfahren sind nur wenig schlechter, als die, die man beim exakten Lösen einer Aufgabe erhalten hätte, die sich nur geringfügig von der ursprünglichen unterscheidet. Solche Verfahren, die wir in Kategorie 2 eingestuft haben, werden vom Standpunkt der Rückwärts-Fehleranalyse weithin als befriedigend akzeptiert.
- Verfahren der Kategorie 2 können Ergebnisse produzieren, die relativ weit von dem entfernt sind, was man erhalten hätte, wenn gar keine Fehler begangen worden wären, doch können große Fehler nur für Werte relativ nahe an einer Singularität der zu berechnenden Funktion auftreten.
- Verfahren der Kategorie 1 produzieren relativ genaue Ergebnisse, auch bei Annäherung an eine Singularität. Solche Verfahren sind selten, doch vorzuziehen, allein schon deshalb, da ihre Ergebnisse einfacher zu interpretieren sind, besonders wenn sie von mehreren Variablen abhängen.

Ein einfaches Beispiel illustriert alle vier Gesichtspunkte.

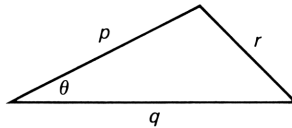
Beispiel 7: Der Winkel in einem Dreieck. Der Kosinussatz besagt für Dreiecke, daß

$$r^2 = p^2 + q^2 - 2pq \cos \theta$$

(siehe Abbildung) gilt. Berechnungen in Wissenschaft und Technik erfordern oft, daß der Winkel θ aus den gegebenen Längen p , q und r der Dreiecksseiten berechnet werden soll. Das ist möglich, wenn $0 < p \leq q + r$, $0 < q \leq p + r$ und $0 < r \leq p + q$ gilt; dann gilt

$$0 \leq \theta = \cos^{-1} ((p^2 + q^2) - r^2)/(2pq) \leq 180^\circ;$$

ansonsten gibt es kein Dreieck mit diesen Seitenlängen, oder $\theta = 0/0$ ist nicht definiert.



Die obige Gleichung für θ definiert eine Funktion $\theta = f(p, q, r)$ und auch auf natürliche Weise ein Programm $F(p, q, r)$, das diese Funktion berechnet. Dieses Programm ist im folgenden unter dem Label «A» gelistet, und die Ergebnisse $F_A(p, q, r)$ sind für einige Eingaben p , q und r tabelliert. Diese Eingaben entsprechen sehr spitzwinkligen Dreiecken, für die das Programm sehr unter Rundungsfehlern leidet. Die numerische Unzuverlässigkeit dieser Gleichung ist wohlbekannt, ebenso wie die der algebraisch äquivalenten, doch zuverlässigeren Gleichung $\theta = f(p, q, r) = 2 \tan^{-1} \sqrt{ab/(cs)}$, wo $s = (p + q + r)/2$, $a = s - p$, $b = s - q$ und $c = s - r$ gilt. Ein weiteres Programm $F(p, q, r)$, das auf dieser besseren Formel basiert, ist im folgenden unter dem Label «B» gelistet; die Ergebnisse für ausgewählte Eingaben erscheinen unter $F_B(p, q, r)$. Offensichtlich ist F_B nicht viel zuverlässiger als F_A . Die meisten dieser schlechten Ergebnisse können mit einer Rückwärts-Fehleranalyse erklärt werden, wenn wir voraussetzen, daß die Berechnungen $F(p, q, r) = f(p + \delta p, q + \delta q, r + \delta r)$ für unbekannte, aber kleine Störungen $|\delta p| < 10^{-9}|p|$ usw. erfüllen. Selbst wenn diese Erklärung stimmen würde, hätte sie erstaunliche und unangenehme Konsequenzen, da die Winkel in spitzwinkligen Dreiecken sich relativ stark verändern, wenn die Seitenlängen nur relativ wenig geändert werden; für Grenzeingaben ist $f(p, q, r)$ relativ instabil.

Tatsächlich ist diese Erklärung falsch. Eine Rückwärts-Fehleranalyse könnte die Ergebnisse F_A und F_B in Fall 1 nur dann erklären, wenn Störungen δp , δq , δr die Eingabe bereits in der fünften Stelle verfälschen dürften, was 1 zu 1.0001 oder 0.9999 machen würde. Für 10 Stellen Rechengenauigkeit sind diese Störungen untolerierbar groß. Ein weit besseres Programm ist F_C mit dem Label «C», das wir etwas später erklären.

In den drei unteren Zeilen der folgenden Tabelle sind die Ergebnisse der drei Programme «A», «B» und «C» angegeben, die mit drei verschiedenen Formeln $F(p,q,r)$ berechnet wurden, die alle algebraisch-äquivalent sind zu

$$\theta = f(p,q,r) = \cos^{-1}((p^2 + q^2 - r^2)/(2pq)).$$

Die unterschiedlichen Ergebnisse der drei Programme F_A , F_B , F_C

Fall 1		Fall 2	Fall 3
p	1.	9.999999996	10.
q	1.	9.999999994	5.000000001
r	1.00005×10^{-5}	3×10^{-9}	15.
F_A	0.	0.	180.
F_B	5.73072×10^{-4}	Error 0	180.
F_C	5.72986×10^{-4}	1.28117×10^{-8}	179.9985965
Fall 4		Fall 5	Fall 6
p	0.527864055	9.999999996	9.999999999
q	9.472135941	3×10^{-9}	9.999999999
r	9.999999996	9.999999994	20.
F_A	Error 0	48.18968509	180.
F_B	Error 0	Error 0	180.
F_C	180.	48.18968510	Error 0
Fall 7		Fall 8	Fall 9
p	1.00002	3.162277662	3.162277662
q	1.00002	2.3×10^{-9}	1.5555×10^{-6}
r	2.00004	3.162277661	3.162277661
F_A	Error 0	90.	90.
F_B	180.	70.52877936	89.96318706
F_C	180.	64.22853822	89.96315156

Um ein Programm zu benutzen, geben Sie p [ENTER] q [ENTER] r ein, lassen Programm «A», «B» oder «C» laufen und schauen sich dann die Näherung F an, die das Programm für $\theta = f$ erhält. Nur das Programm «C» liefert verlässliche Ergebnisse.

Tastenfolge

Anzeige

g DEG	000–
g P/R	001–42,21,11
f CLEAR PRGM	002– 43 11
f LBL A	003– 34
g x²	004– 43 11
x$\hat{=}$y	005– 43 36
g x²	006– 43 33
g LSTx	007– 20
g R∇	008– 34
x	009– 43 36
x$\hat{=}$y	010– 43 11
g LSTx	011– 40
g x²	012– 43 33
+	013– 30
g R∇	014– 34
–	015– 36
x$\hat{=}$y	016– 40
ENTER	017– 10
+	018– 43 24
÷	019– 43 32
g COS⁻¹	020–42,21,12
g RTN	021– 44 1
f LBL B	022– 36
STO 1	023– 43 33
ENTER	024–44,40, 1
g R∇	025– 43 33
STO + 1	026–44,40, 1
g R∇	027– 2
STO + 1	028–44,10, 1
2	029– 33
STO ÷ 1	030–45,30, 1
R∇	031– 34
RCL – 1	032–45,30, 1
x$\hat{=}$y	033– 20
RCL – 1	034– 11
x	035– 34
√x	036–45,30, 1
x$\hat{=}$y	037–45,20, 1
RCL – 1	
RCL × 1	

Tastenfolge

Anzeige

CHS	038– 16
$\sqrt{x}$	039– 11
g $\Rightarrow P$	040– 43 1
R∇	041– 33
x	042– 20
g RTN	043– 43 32
f LBL C	044–42,21,13
STO 0	045– 44 0
R∇	046– 33
g $x \leq y$	047– 43 10
$x \geq y$	048– 34
STO 1	049– 44 1
STO + 0	050–44,40, 0
$x \geq y$	051– 34
STO + 0	052–44,40, 0
-	053– 30
g R∇	054– 43 33
STO - 1	055–44,30, 1
g LSTx	056– 43 36
ENTER	057– 36
RCL + 1	058–45,40, 1
$\sqrt{x}$	059– 11
f $x \geq 0$	060–42, 4, 0
$\sqrt{x}$	061– 11
STO x 0	062–44,20, 0
g CLx	063– 43 35
+	064– 40
R∇	065– 33
+	066– 40
f $x \geq 1$	067–42, 4, 1
g R∇	068– 43 33
g LSTx	069– 43 36
g $x \leq y$	070– 43 10
GTO .9	071– 22 .9
R∇	072– 33
g TEST 2	073–43,30, 2
$\sqrt{x}$	074– 11
$x \geq y$	075– 34
GTO .8	076– 22 .8
f LBL .9	077–42,21, .9

Tastenfolge

Anzeige

g TEST 2	078- 43,30, 2
√x	079- 11
g R#	080- 43 33
f LBL .8	081-42,21, .8
-	082- 30
√x	083- 11
RCL 1	084- 45 1
√x	085- 11
x	086- 20
RCL 0	087- 45 0
g →P	088- 43 1
g x=0	089- 43 20
÷	090- 10
x^zy	091- 34
ENTER	092- 36
+	093- 40
g RTN	094- 43 32
g P/R	

Die Ergebnisse $F_C(p, q, r)$ sind auf mindestens neun Stellen genau. Sie werden vom einem Programm «C» berechnet, das vollkommen verlässlich, jedoch länger als die nicht verlässlichen Programme «A» und «B» ist. Das Prinzip des Programmes «C» ist:

1. Falls $p < q$, vertausche sie, damit $p \geq q$.
2. Berechne $b = (p - q) + r$, $c = (p - r) + q$ und $s = (p + r) + q$.
3. Berechne

$$a = \begin{cases} r - (p - q) & \text{falls } q \geq r \geq 0 \\ q - (p - r) & \text{falls } r > q \geq 0 \\ \text{Error 0} & \text{sonst (es gibt kein solches Dreieck).} \end{cases}$$

4. Berechne $F_C(p, q, r) = 2 \tan^{-1} (\sqrt{ab}/\sqrt{cs})$.

Dieses Verfahren ergibt $F_C(p, q, r) = \theta$ mit fast neun korrekten Stellen, sicher ein besseres Ergebnis als die der anderen, besser bekannten Formeln. Doch die interne Funktionsweise dieses Verfahrens ist schwer zu erklären; auf manchen Taschenrechnern und Computern wird dieses Verfahren sogar überhaupt nicht funktionieren. Einwandfrei arbeitet dieses Verfahren nur auf Rechnern wie dem HP-15C, bei denen die Subtraktionsoperation keine vermeidbaren Fehler hat

und daher die folgende Eigenschaft besitzt: Immer wenn y zwischen $x/2$ und x liegt, ergibt die Subtraktionsoperation keinen zusätzlichen Rundungsfehler in dem berechneten Ergebnis $x-y$. Daher hat die wesentliche Differenz $(p-q)$ oder $(p-r)$ immer dann, wenn a , b oder c relativ stark durch Auslöschung verfälscht werden könnten, keinen Fehler, und dann stellt sich die Auslöschung als vorteilhaft heraus!

Auslöschung bleibt jedoch auf solchen anderen Rechnern, die statt $x-y$ $(x+\delta x)-(y+\delta y)$ berechnen, ein Problem, obwohl weder δx noch δy auch nur eine Einheit der letzten Stelle von x oder y ausmacht. Solche Rechner erhalten $F_C(p,q,r) = f(p+\delta p, q+\delta q, r+\delta r)$ mit Störungen δp , δq und δr in der letzten Stelle, die zwar unter dem Gesichtspunkt der Rückwärts-Fehleranalyse vernachlässigbar zu sein scheinen, doch beunruhigende Konsequenzen besitzen können. Beispielsweise könnte nur eines der Triple (p,q,r) oder $(p+\delta p, q+\delta q, r+\delta r)$, aber nicht beide, die Seitenlängen eines möglichen Dreieckes bilden, so daß F_C auf solchen Rechnern vielleicht eine Fehlermeldung liefert, wenn es gar keine geben sollte, oder umgekehrt.

Rückwärts-Fehleranalyse bei der Matrixinversion

Ein Maß für den Betrag einer Matrix $\mathbf{X}$ ist eine Norm $\|\mathbf{X}\|$, wie sie z.B. von **MATRIX** 7 oder **MATRIX** 8 berechnet wird; wir verwenden im folgenden die Zeilensummennorm

$$\|\mathbf{X}\| = \max_i \sum_j |x_{ij}|$$

Diese Norm besitzt ähnliche Eigenschaften wie die Länge eines Vektors und ist insbesondere multiplikativ:

$$\|\mathbf{XY}\| \leq \|\mathbf{X}\| \|\mathbf{Y}\|.$$

Wenn für eine vorgegebene $n \times n$ Matrix $\mathbf{A}$ und einen Spaltenvektor $\mathbf{b}$ das Gleichungssystem $\mathbf{Ax} = \mathbf{b}$ numerisch gelöst wird, ist die berechnete Lösung ein Spaltenvektor $\mathbf{c}$, der fast dasselbe Gleichungssystem wie $\mathbf{x}$ erfüllt, nämlich

$$(\mathbf{A} + \delta\mathbf{A})\mathbf{c} = \mathbf{b}$$

mit $\|\delta\mathbf{A}\| < 10^{-9}n\|\mathbf{A}\|$.

Daher ist das Residuum $\mathbf{b} - \mathbf{A}\mathbf{c} = (\delta\mathbf{A})\mathbf{c}$ immer relativ klein; oft ist die Norm $\|\mathbf{b} - \mathbf{A}\mathbf{c}\|$ des Residuums kleiner als $\|\mathbf{b} - \mathbf{A}\bar{\mathbf{x}}\|$, wo $\bar{\mathbf{x}}$ aus der korrekten Lösung $\mathbf{x}$ durch Runden aller ihrer Elemente auf 10 signifikante Stellen erhalten wird. Daher kann $\mathbf{c}$ sich von $\mathbf{x}$ nur dann wesentlich unterscheiden, wenn $\mathbf{A}$ fast singulär ist, oder äquivalent dazu, wenn $\|\mathbf{A}^{-1}\|$ relativ groß im Vergleich zu $1/\|\mathbf{A}\|$ ist;

$$\begin{aligned}\|\mathbf{x} - \mathbf{c}\| &= \|\mathbf{A}^{-1}(\mathbf{b} - \mathbf{A}\mathbf{c})\| \\ &\leq \|\mathbf{A}^{-1}\| \|\delta\mathbf{A}\| \|\mathbf{c}\| \\ &\leq 10^{-9}n \|\mathbf{c}\| / \sigma(\mathbf{A})\end{aligned}$$

wo $\sigma(\mathbf{A}) = 1/(\|\mathbf{A}\| \|\mathbf{A}^{-1}\|)$ der Kehrwert der Konditionszahl ist und ein Maß dafür gibt, wie relativ nahe bei $\mathbf{A}$ die nächste singuläre Matrix $\mathbf{S}$ liegt, da gilt

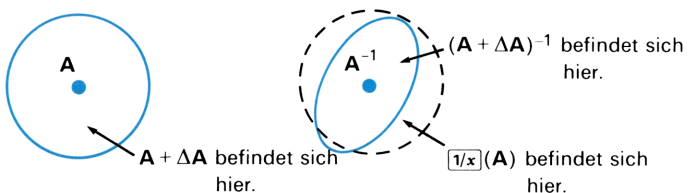
$$\min_{\det(\mathbf{S})=0} \|\mathbf{A} - \mathbf{S}\| = \sigma(\mathbf{A})\|\mathbf{A}\|.$$

Diese Beziehungen und einige ihrer Konsequenzen werden ausführlich im Abschnitt 4 erläutert.

Die Berechnung von $\mathbf{A}^{-1}$ ist komplizierter. Jede Spalte der berechneten Inversen $\boxed{1/\mathbf{x}}(\mathbf{A})$ ist die entsprechende Spalte einer Matrix $(\mathbf{A} + \delta\mathbf{A})^{-1}$, doch hat jede Spalte ihr eigenes kleines $\delta\mathbf{A}$. Daher braucht kein einziges kleines $\delta\mathbf{A}$ mit $\|\delta\mathbf{A}\| \leq 10^{-9}n\|\mathbf{A}\|$ zu existieren, das etwa

$$\|(\mathbf{A} + \delta\mathbf{A})^{-1} - \boxed{1/\mathbf{x}}(\mathbf{A})\| \leq 10^{-9}\|\boxed{1/\mathbf{x}}(\mathbf{A})\|$$

erfüllt. Meistens, doch nicht immer, existiert ein solches $\delta\mathbf{A}$. Damit wird die frühere Annahme, daß die Matrixoperationen $\boxed{1/\mathbf{x}}$ und $\boxed{\div}$ sich in Kategorie 2 befinden, nicht verletzt; denn sie fallen unter die zweite Aussage der Zusammenfassung auf Seite 194. Man kann die Genauigkeit von $\boxed{1/\mathbf{x}}(\mathbf{A})$ mit Hilfe der Inversen aller Matrizen $\mathbf{A} + \Delta\mathbf{A}$ beschreiben, die so nahe an $\mathbf{A}$ liegen, daß $\|\Delta\mathbf{A}\| \leq 10^{-9}n\|\mathbf{A}\|$ gilt. Die schlechteste all dieser $(\mathbf{A} + \Delta\mathbf{A})^{-1}$ unterscheidet sich in ihrer Norm von $\mathbf{A}^{-1}$ um mindestens soviel, wie das berechnete $\boxed{1/\mathbf{x}}(\mathbf{A})$. Die folgende Abbildung erläutert die Situation.



Wenn $\mathbf{A} + \Delta\mathbf{A}$ Matrizen durchläuft, für die $\|\Delta\mathbf{A}\|$ mindestens so groß wie der Rundungsfehler in $\|\mathbf{A}\|$ ist, muß ihre Inverse $(\mathbf{A} + \Delta\mathbf{A})^{-1}$ Werte durchlaufen, die mindestens so weit von $\mathbf{A}^{-1}$ entfernt sind, wie die Entfernung von $\mathbf{A}^{-1}$ zum berechneten $\overline{1/x}(\mathbf{A})$. All diese Abweichungen sind sehr klein, außer wenn $\mathbf{A}$ zu nahe an einer singulären Matrix liegt, und in diesem Fall sollte die Matrix von der nahen Singularität weg vorkonditioniert werden (siehe Abschnitt 4).

Wenn einige dieser benachbarten Matrizen $\mathbf{A} + \Delta\mathbf{A}$ singulär sind, dann können sich $(\mathbf{A} + \Delta\mathbf{A})^{-1}$ und $\overline{1/x}(\mathbf{A})$ sehr von $\mathbf{A}^{-1}$ unterscheiden. Die Norm des Residuums ist jedoch immer relativ klein:

$$\frac{\|\mathbf{A}(\mathbf{A} + \Delta\mathbf{A})^{-1} - \mathbf{I}\|}{\|\mathbf{A}\| \|(\mathbf{A} + \Delta\mathbf{A})^{-1}\|} \leq \frac{\|\Delta\mathbf{A}\|}{\|\mathbf{A}\|} \leq 10^{-9}n.$$

Die letzte Ungleichung stimmt auch dann, wenn man $\overline{1/x}(\mathbf{A})$ statt $(\mathbf{A} + \Delta\mathbf{A})^{-1}$ wählt.

Wenn $\mathbf{A}$ weit genug von einer Singularität entfernt ist, so daß

$$1/\|(\mathbf{A} + \Delta\mathbf{A})^{-1}\| > 10^{-9}n\|\mathbf{A}\| \geq \|\Delta\mathbf{A}\|,$$

gilt, dann gilt auch

$$\begin{aligned} \frac{\|\mathbf{A}^{-1} - (\mathbf{A} + \Delta\mathbf{A})^{-1}\|}{\|(\mathbf{A} + \Delta\mathbf{A})^{-1}\|} &\leq \frac{\|\Delta\mathbf{A}\| \|(\mathbf{A} + \Delta\mathbf{A})^{-1}\|}{1 - \|\Delta\mathbf{A}\| \|(\mathbf{A} + \Delta\mathbf{A})^{-1}\|} \\ &\leq \frac{10^{-9}n\|\mathbf{A}\| \|(\mathbf{A} + \Delta\mathbf{A})^{-1}\|}{1 - 10^{-9}n\|\mathbf{A}\| \|(\mathbf{A} + \Delta\mathbf{A})^{-1}\|} \end{aligned}$$

Diese Ungleichung ist auch dann richtig, wenn man $\overline{1/x}(\mathbf{A})$ statt $(\mathbf{A} + \Delta\mathbf{A})^{-1}$ wählt. In diesem Fall können die Größen der rechten Seite berechnet werden, so daß der Fehler in $\overline{1/x}(\mathbf{A})$ einen bekannten Wert nicht überschreiten kann. Das heißt der Radius des gestrichelten Kreises in der obigen Abbildung kann berechnet werden.

Die obigen Abschätzungen sind meistens pessimistisch. Um zu sehen, daß jedoch im allgemeinen nichts viel Besseres gilt, betrachten wir die Matrix

$$\mathbf{X} = \begin{bmatrix} 0.00002 & -50,000 & 50,000.03 & -45 \\ 0 & 50,000 & -50,000.03 & 45 \\ 0 & 0 & 0.00002 & -50,000.03 \\ 0 & 0 & 0 & 52,000 \end{bmatrix}$$

und

$$\mathbf{X}^{-1} = \begin{bmatrix} 50,000 & 50,000 & p & q \\ 0 & 0.00002 & 50,000.03 & 48,076.98077... \\ 0 & 0 & 50,000 & 48,076.95192... \\ 0 & 0 & 0 & 0.00001923076923... \end{bmatrix}$$

Eigentlich gilt $p = q = 0$, doch die Näherung für $\mathbf{X}^{-1}$, die der HP-15C berechnet, nämlich $\boxed{1/x}(\mathbf{X})$, liefert stattdessen $q = 9,643.269231$, was ein relativer Fehler von

$$\frac{\|\mathbf{X}^{-1} - \boxed{1/x}(\mathbf{X})\|}{\|\mathbf{X}^{-1}\|} = 0.0964...,$$

ist, fast 10 Prozent. Wenn sich andererseits $\mathbf{X} + \Delta\mathbf{X}$ von $\mathbf{X}$ nur in der zweiten Spalte unterscheidet, wo $-50,000$ und $50,000$ durch $-50,000.000002$ und $49,999.999998$ ersetzt wurden (eine Änderung in der 11ten Stelle), dann unterscheidet sich $(\mathbf{X} + \Delta\mathbf{X})^{-1}$ nur darin von $\mathbf{X}^{-1}$, daß $p = 0$ und $q = 0$ durch $p = 10,000.00600...$ und $q = 9,615.396154...$ ersetzt werden müssen. Daher gilt

$$\frac{\|\mathbf{X}^{-1} - (\mathbf{X} + \Delta\mathbf{X})^{-1}\|}{\|\mathbf{X}^{-1}\|} = 0.196...;$$

der relative Fehler in $(\mathbf{X} + \Delta\mathbf{X})^{-1}$ ist beinahe doppelt so groß wie in $\boxed{1/x}(\mathbf{X})$. Versuchen Sie daher nicht, $(\mathbf{X} + \Delta\mathbf{X})^{-1}$ direkt zu berechnen, sondern verwenden Sie stattdessen die Gleichung

$$(\mathbf{X} - \mathbf{c}\mathbf{b}^T)^{-1} = \mathbf{X}^{-1} + \mathbf{X}^{-1}\mathbf{c}\mathbf{b}^T\mathbf{X}^{-1} / (1 - \mathbf{b}^T\mathbf{X}^{-1}\mathbf{c}),$$

die für beliebige Spaltenvektoren $\mathbf{c}$ und Zeilenvektoren $\mathbf{b}^T$ gilt, insbesondere für

$$\mathbf{c} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \quad \text{und} \quad \mathbf{b}^T = [0 \quad 0.000002 \quad 0 \quad 0].$$

Obwohl

$$\|\mathbf{X}^{-1} - \boxed{1/x}(\mathbf{X})\| < \|\mathbf{X}^{-1} - (\mathbf{X} + \Delta\mathbf{X})^{-1}\|,$$

gilt, kann man zeigen, daß keine sehr kleine Störung $\delta\mathbf{X}$ der letzten Stelle existiert, für die die Norm von $(\mathbf{X} + \delta\mathbf{X})^{-1}$ mit $\boxed{1/x}(\mathbf{X})$ in mehr als fünf Stellen übereinstimmt.

Natürlich können all diese schrecklichen Dinge nur passieren, wenn $\mathbf{X}$ so fast singulär ist. Da $\|\mathbf{X}\| \|\mathbf{X}^{-1}\| > 10^{10}$ gilt, kann eine Änderung von $\mathbf{X}$, die weniger als eine Einheit in der letzten Stelle von $\|\mathbf{X}\|$ ausmacht, $\mathbf{X}$ singulär machen; eine solche Änderung könnte eines der Diagonalelemente 0.00002 von $\mathbf{X}$ durch 0 ersetzen. Da $\mathbf{X}$ fast singulär ist, ist $\boxed{1/x}(\mathbf{X})$ in diesem Fall genauer, als man allgemein erwarten würde. Das besondere an diesem Beispiel ist schlechtes Skalieren; $\mathbf{X}$ wurde aus der nicht ungewöhnlichen Matrix

$$\tilde{\mathbf{X}} = \begin{bmatrix} 2. & -5. & 5.000003 & -4.5 \times 10^{-12} \\ 0 & 5. & -5.000003 & 4.5 \times 10^{-12} \\ 0 & 0 & 2. & -5.000003 \\ 0 & 0 & 0 & 5.2 \end{bmatrix}$$

ermittelt, indem alle Zeilen und Spalten mit einer sorgfältig gewählten Zehnerpotenz multipliziert wurden. Zum Ausgleich wurden die Zeilen und Spalten der ebenfalls nicht ungewöhnlichen Matrix

$$\tilde{\mathbf{X}}^{-1} = \begin{bmatrix} 0.5 & 0.5 & p & q \\ 0 & 0.2 & 0.5000003 & 0.4807698077... \\ 0 & 0 & 0.5 & 0.4807695192... \\ 0 & 0 & 0 & 0.1923076923... \end{bmatrix}$$

dividiert, und dies lieferte $\mathbf{X}^{-1}$, mit $p=q=0$. Der HP-15C berechnet $\boxed{1/x}(\tilde{\mathbf{X}}) = \tilde{\mathbf{X}}^{-1}$, außer daß $q=0$ durch $q=9.6 \times 10^{-11}$ ersetzt ist, was ein vernachlässigbarer Fehler ist. Dieses Beispiel illustriert, wie sehr die Genauigkeit von berechneten Ergebnissen vom Skalieren abhängen kann. (Mehr Information über die Skalierung von Matrizen finden Sie in Abschnitt 4.)

Ist die Rückwärts-Fehleranalyse sinnvoll?

Der einzige Vorteil der Rückwärts-Fehleranalyse ist die Tatsache, daß sie interne Fehler auf eine Weise erklärt, die den Benutzer eines Systems davon befreit, über die internen Einzelheiten des Systems Bescheid zu wissen. Mit zwei gegebenen Toleranzen, eine für die Eingabestörung δx und die andere für die Ausgabestörung δf , kann der Benutzer die Folgen von internen Störungen in

$$F(x) = (f + \delta f)(x + \delta x)$$

analysieren, indem er die Fehlerfortpflanzungseigenschaften des idealen Systems f betrachtet, ohne sich weiter um die möglicherweise komplexe interne Struktur von F kümmern zu müssen.

Die Rückwärts-Fehleranalyse ist jedoch kein Allheilmittel; sie kann Fehler erklären, aber sie nicht entschuldigen. Wir haben versucht, die Notwendigkeit für sie zu eliminieren, wo immer das möglich war, da sie Berechnungen mit Singularitäten komplizierter macht. Wenn wir die Notwendigkeit für Rückwärts-Fehleranalyse in allen in den Rechner eingebauten Funktionen zu einem erträglichen Preis eliminieren könnten, würden wir dies tun und damit das Leben für alle einfacher machen. Doch bei dem heutigen Stand der Technik würde diese Vereinfachung zuviel Rechenzeit und Speicherplatz benötigen. Das folgende Beispiel illustriert einige der Nachteile, die man dabei in Kauf nehmen muß.

Fortsetzung Beispiel 6. Das folgende Programm findet die reellen oder komplexen Nullstellen der reellen quadratischen Gleichung $c - 2bz + az^2 = 0$.

Um das Programm zu benutzen, geben Sie die reellen Konstanten in den Stack ein (c **[ENTER]** c **[ENTER]** a) und starten Sie das Programm «A».

Die Nullstellen x und y werden im X- und Y-Register abgelegt. Wenn die Nullstellen komplex sind, erscheint die Anzeige **C** und zeigt, daß der Komplex-Modus aktiviert worden ist. Das Programm verwendet die Labels «A» und «.9» und das Indexregister (aber keines der übrigen Register 0 bis .9); daher kann das Programm leicht von anderen Programmen als Unterprogramm aufgerufen werden. Aufrufende Programme können feststellen, ob die Nullstellen reell oder komplex sind, indem sie Flag 8 prüfen, der nur gesetzt wird, wenn die Nullstellen komplex sind.

Die Nullstellen x und y sind so geordnet, daß $|x| \geq |y|$ gilt, es sei denn $|x|$ und $|y|$ stimmen in mehr als neun Stellen überein. Die Nullstellen sind so genau, als wenn der Koeffizient c in der 10ten Stelle gestört worden wäre, dann die gestörte Gleichung exakt gelöst worden wäre und ihre Nullstellen auf 10 Stellen gerundet worden wären. Folglich stimmen die berechneten Nullstellen mit denen der gegebenen Gleichung in mindestens fünf signifikanten Stellen überein. Allgemeiner, wenn die Nullstellen x und y für irgendein positives $n \leq 5$ in n signifikanten Stellen übereinstimmen, dann sind sie auf mindestens $10 - n$ Stellen korrekt, es sei denn, es tritt ein Overflow oder Underflow auf.

Tastenfolge

Anzeige

[g] [P/R]	000—
[f] [CLEAR] [PRGM]	001—42,21,11
[f] [LBL] [A]	002— 36
[ENTER]	003— 43 33
[g] [R↑]	004— 20
[x]	005— 43 36
[g] [LSTx]	

Tastenfolge

Anzeige

$x \geq y$	006– 34
$\frac{g}{R\uparrow}$	007– 43 33
$\frac{STO}{I}$	008– 44 25
$\frac{g}{x^2}$	009– 43 11
$-$	010– 30
$\frac{g}{TEST} 1$	011–43,30, 1
$\frac{GTO}{.9}$	012– 22 .9
$\frac{CHS}{}$	013– 16
$\sqrt{x}$	014– 11
$\frac{f}{x \geq} I$	015–42, 4,25
$\frac{g}{TEST} 2$	016–43,30, 2
$\frac{RCL}{-} I$	017–45,30,25
$\frac{g}{TEST} 3$	018–43,30, 3
$\frac{RCL}{+} I$	019–45,40,25
$\frac{g}{TEST} 0$	020–43,30, 0
$\div$	021– 10
$\frac{g}{LSTx}$	022– 43 36
$\frac{g}{R\uparrow}$	023– 43 33
$\div$	024– 10
$\frac{g}{RTN}$	025– 43 32
$\frac{f}{LBL} .9$	026–42,21, .9
$\sqrt{x}$	027– 11
$\frac{RCL}{I}$	028– 45 25
$\frac{g}{R\uparrow}$	029– 43 33
$\div$	030– 10
$x \geq y$	031– 34
$\frac{g}{LSTx}$	032– 43 36
$\div$	033– 10
$\frac{f}{I}$	034– 42 25
$\frac{ENTER}{}$	035– 36
$\frac{f}{Re\Im}$	036– 42 30
$\frac{CHS}{}$	037– 16
$\frac{f}{Re\Im}$	038– 42 30
$\frac{g}{RTN}$	039– 43 32
$\frac{g}{P/R}$	

Das Verfahren benutzt $d = b^2 - ac$.

Für $d < 0$ sind die Nullstellen ein komplex konjugiertes Paar

$$(b/a) \pm i\sqrt{-d}/a.$$

Für $d \geq 0$ sind die Nullstellen reelle Zahlen x und y , die aus

$$s = b + \sqrt{d} \operatorname{sgn}(b)$$

$$x = s/a$$

$$y = \begin{cases} c/s & \text{für } s \neq 0 \\ 0 & \text{für } s = 0 \end{cases}$$

berechnet werden. Die Berechnung von s vermeidet zerstörerische Auslöschung.

Für $a = 0 \neq b$ wird die größere Nullstelle x , die ∞ sein sollte, durch null geteilt (**Error 0**), die Meldung verschwindet nach dreimaligem Drücken von $\boxed{R\downarrow}$ und die korrekt berechnete kleinere Nullstelle y wird angezeigt. Wenn jedoch alle drei Koeffizienten gleich null sind, zeigt die **Error 0** Meldung an, daß beide Nullstellen beliebig sind.

Die Ergebnisse verschiedener Fälle sind hier zusammengefaßt.

	Fall 1	Fall 2	Fall 3	Fall 4
c	3	4	1	654,321
b	2	0	1	654,322
a	1	1	10^{-13}	654,323
Nullstellen	Reell	Komplex	Reell	Reell
	3	$0 \pm 2i$	2×10^{13}	0.9999984717
	1		0.5	0.9999984717
	Fall 5		Fall 6	
c	46,152,709		12,066,163	
b	735,246		987,644	
a	11,713		80,841	
Nullstellen	Reell		Komplex	
	62.77179203		$12.21711755 \pm 0.001377461i$	
	62.77179203			

Die letzten drei Fälle zeigen, wie stark sich eine Störung in der 10ten Stelle irgendeines Koeffizienten irgendeiner quadratischen Gleichung, deren Null-

stellen beinahe zusammenfallen, auswirkt. Die korrekten Nullstellen sind in diesen Fällen

Fall 4: 1 und 0.9999969434

Fall 5: $62.77179203 \pm i8.5375 \times 10^{-5}$

Fall 6: $12.21711755 \pm i0.001374514$

Für fast alle Anwendungen von quadratischen Gleichungen in Wissenschaft und Technik reicht das Unterprogramm «A» trotz Fehlern in der fünften Stelle des Ergebnisses aus. Für die meisten Daten sind die Ergebnisse, die es errechnet, auf neun signifikante Stellen exakt, und das schließt den Fall, daß c , b und a mit nur fünf signifikanten Stellen exakt dargestellt werden können, ein; die berechneten Nullstellen sind immer in mindestens fünf Stellen genau, da sie nicht wesentlich schlechter sein können, als wenn die Werte mit Fehlern in der 10ten Stelle eingegeben worden wären. Trotzdem werden sich einige Leser bei Ergebnissen unwohl fühlen, die auf 10 Stellen berechnet wurden, aber nur auf 5 Stellen exakt sind. Sie würden wohl Ergebnisse mit in jedem Fall neun exakten Stellen vorziehen, und sei es nur, um ihr Verständnis von dem Verhältnis zwischen eingegebenen Werten und ausgegebenen Ergebnissen zu vereinfachen.

Es existieren tatsächlich Programme, die die Nullstellen jeder beliebigen quadratischen Gleichung auf mindestens neun Stellen genau berechnen, unabhängig davon, wie nahe diese Nullstellen zusammenliegen mögen, obwohl ihre Arithmetik nur mit 10 Stellen rechnet. Alle solche Programme berechnen $d = b^2 - ac$ mit irgendeinem Trick, der darauf hinausläuft, immer dann, wenn b^2 und ac fast gleich sind, mit 20 Stellen Genauigkeit zu rechnen, so daß solche Programme viel länger und langsamer sind, als unser einfaches Unterprogramm «A». Das folgende Unterprogramm «B», das einen solchen Trick* verwendet, ist ein sehr kurzes Programm, das auf einem Rechner mit 10stelliger Rechengenauigkeit neun korrekte Stellen garantiert. Es verwendet die Labels «B», «.7» und «.8» und die Register R_0 bis R_9 und das Indexregister. Um es zu benutzen, geben Sie c [ENTER] b [ENTER] a ein, starten Sie das Programm «B», und warten Sie wie vorher auf die Ergebnisse.

Tastenfolge

g P/R
f CLEAR PRGM
f LBL B
STO I
R↑

Anzeige

000–
001–42,21,12
002– 44 25
003– 33

* Programm «B» verwendet eine spezielle Eigenschaft der Tasten Σ^- und Σ^+ , wo gewisse Berechnungen mit 13stelliger Genauigkeit durchgeführt werden, bevor das Ergebnis auf 10 Stellen gerundet wird.

Tastenfolge

STO 0
 STO 8
 $x \geq y$
 STO 1
 STO 9
 f SCI 2
 f LBL .8
 f CLEAR Σ
 RCL 8
 STO 7
 RCL $\div$ 1
 g RND
 RCL 1
 g Σ^-
 RCL 9
 f $x \geq y$ 7
 $x \geq y$
 RCL 8
 g Σ^-
 R#
 g Σ^-
 RCL 7
 g ABS
 RCL 9
 g ABS
 g $x \leq y$
 GTO B
 ENTER
 g R#
 STO 8
 RCL 7
 STO 9
 g ABS
 EEX
 2
 0
 x
 RCL 1
 g ABS
 g $x \leq y$

Anzeige

004- 44 0
 005- 44 8
 006- 34
 007- 44 1
 008- 44 9
 009-42, 8, 2
 010-42,21, .8
 011- 42 32
 012- 45 8
 013- 44 7
 014-45,10,25
 015- 43 34
 016- 45 25
 017- 43 49
 018- 45 9
 019-42, 4, 7
 020- 34
 021- 45 8
 022- 43 49
 023- 33
 024- 43 49
 025- 45 7
 026- 43 16
 027- 45 9
 028- 43 16
 029- 43 10
 030- 22 12
 031- 36
 032- 43 33
 033- 44 8
 034- 45 7
 035- 44 9
 036- 43 16
 037- 26
 038- 2
 039- 0
 040- 20
 041- 45 1
 042- 43 16
 043- 43 10

Tastenfolge

GTO .8
 f LBL B
 f FIX 9
 RCL 8
 g x^2
 STO 7
 RCL I
 RCL 9
 g Σ^-
 RCL 7
 g TEST 2
 GTO .7
 $\sqrt{x}$
 f $x \geq 0$
 g TEST 2
 RCL - 0
 g TEST 3
 RCL + 0
 f $x \geq 1$
 g TEST 0
 RCL $\div$ 1
 RCL 1
 RCL $\div$ I
 g RTN
 f LBL .7
 CHS
 $\sqrt{x}$
 RCL $\div$ I
 ENTER
 CHS
 RCL 0
 RCL I
 $\div$
 $x \geq y$
 f I
 ENTER
 g R $\uparrow$
 f I
 g RTN
 g P/R

Anzeige

044- 22 .8
 045-42,21,12
 046-42, 7, 9
 047- 45 8
 048- 43 11
 049- 44 7
 050- 45 25
 051- 45 9
 052- 43 49
 053- 45 7
 054-43,30, 2
 055- 22 .7
 056- 11
 057-42, 4, 0
 058-43,30, 2
 059-45,30, 0
 060-43,30, 3
 061-45,40, 0
 062-42, 4, 1
 063-43,30, 0
 064-45,10, 1
 065- 45 1
 066-45,10,25
 067- 43 32
 068-42,21, .7
 069- 16
 070- 11
 071-45,10,25
 072- 36
 073- 16
 074- 45 0
 075- 45 25
 076- 10
 077- 34
 078- 42 25
 079- 36
 080- 43 33
 081- 42 25
 082- 43 32

Die Genauigkeit dieses Programmes ist phänomenal, mehr als neun Stellen Genauigkeit selbst für die Imaginärteile von fast nicht unterscheidbaren komplexen Nullstellen (wie z.B. $c = 4,877,163,849$, $b = 4,877,262,613$ und $a = 4,877,361,379$). Wenn $a = 1$ ist und die Nullstellen ganze, reelle oder komplexe Zahlen sind, werden diese exakt berechnet (wie z.B. $c = 1,219,332,937 \times 10^1$, $b = 111,111.5$ und $a = 1$). Doch das Programm hat seinen Preis; es verbraucht sowohl für das Programm als auch für die Daten mehr als doppelt soviel Speicherplatz wie das Unterprogramm «A» und viel mehr Zeit, um in einigen wenigen Fällen, wo es eigentlich auch gar nicht darauf ankommt – einfach deshalb, da man kaum jemals die Koeffizienten der quadratischen Gleichung so genau berechnen kann –, statt fünf signifikanten Stellen neun zu erhalten. Wenn auch nur einer der Koeffizienten c , b oder a auch nur um eine Einheit in der 10ten Stelle ungenau ist, dann ist das Unterprogramm «B» sinnlos. Es ist wie Großmutter's teures Porzellan ganz besonderen Anlässen vorbehalten, während das Unterprogramm «A» für den Alltag bleibt.

Index

A

Abbildung, Konturen, 89
Ableitung, 10, 17–20, 192
Absoluter Fehler, 173, 182
Äquipotentiallinie, 89–95
Äquivalente Gleichungen, 9–10
Analyse diskontierter Zahlungsströme, 39–44
Annuitäten, 26–39
Anzahl genauer Stellen, 103, 121
Anzeigeformat, 45–46, 48
ANOVA-Tabelle, 133, 134, 140
Aufeinanderfolgende Zeilen, 140–148
Ausgaberauschen, 188–192
Ausgabestörungen, 188–192
Auslöschung, 176–178, 200, 207

B

Beispiele
 Äquipotentiallinien, 95
 Annuitäten, 34–39
 Besondere Funktionen, 64
 Deklination der Sonne, 11–15
 Der Rechner funktioniert nicht, 172, 175–176
 Eigenvektoren, 157–159
 Feldintensität einer Antenne, 17–25
 Filternetzwerk, 128–131
 Gammafunktion, 65–68
 Konturintegral, 88–89
 Mondphasen, 186
 N-te Wurzeln einer komplexen Zahl, 80
 Normalverteilungsfunktion, 64
 Nullstellen einer quadratischen Gleichung, 191, 205–211
 Optimierungsprogramm für maximales Volumen einer Kiste, 168–171
 Pfennige, 173, 180–183
 Pi, 173, 184–186
 Quadratische Fläche, 153–154

Residuumskorrektur, 121
 Speichern und Zurückrufen komplexer Zahlen, 77–78
 Stromlinien, 93–94
 Test einer Hypothese, 122–128
 Unterteilung des Integrationsintervalles, 51–54
 Variablentransformation, 54–55
 Verbraucherpreisindexregression, 137–140, 147–148
 Verzögerungsdifferentialgleichung, 81–85
 Winkel im Dreieck, 194–199
 Zahlungsreihen, 34–39
 Zahlungsstrom, 43–44
 Zu kurze Brücke, 174
 Berechnen der komplexen Nullstellen einer Gleichung, 80–85
 Beschränkte Suche, 161, 162
 Binomischer Lehrsatz, 176

C

Charakteristische Gleichung, 148

D

Deflation, 10
 Deklination, 11–15
 Deskriptor der LR -Zerlegung, 97
 Determinante, 97–98, 118
 Diagramm, Zahlungsstrom, 28, 28–44
 Doolittle Verfahren, 97
 Dreieck, Winkel im, 194–199

E

Eigenvektor, 149, 154–160
 Eigenwert, 148–160
 Eigenwert, Speichern von, 159–160
 Eindeutige Funktionen, komplexe, 69
 Eindeutige komplexe Funktionen, 69
 Eingabestörungen, 187–192
 Einheitsmatrix, 119
 Elektrostatisches Feld, 59
 Endpunkte, Untersuchung von, 46–47, 56
Error 0, 29, 196, 199, 207
Error 1, 162, 167
Error 4, 29, 40
Error 8, 9, 23

Erweiterte Matrix, 141
 Erweiterte Normalgleichungen, 111
 Erweiterte Rechengenauigkeit, 47, 104, 208
 Erweitertes Gleichungssystem, 142
 Extremwerte einer Funktion, 17–25

F

F -Ratio, 132–140
 Faktorisierung, Orthogonal-, 113–116, 140–148
 Fast singuläre Matrix, 107, 117–118, 201, 204
 Fehler bei $\left[\frac{x}{y}\right]$, 45–46
 Fehler einer Matrix, 100
 in Matricelementen, 100–101
 Mißverständnisse im Zusammenhang mit, 172–178
 relative, 173, 182, 183
 Fehlerhierarchie, 178
 Feldstärke, 17–25
 Finanzaufgaben, 26–44
 Finanzgleichung, 29, 39
 Flanke einer Funktion, 57–58
 Format, Anzeige-, 45–46, 48
 Freiheitsgrade, 132
 Frobeniusnorm, 99
 Funktionen, komplexe, 68–73

G

Gammafunktion, komplexe, 65–68
 Genauigkeit
 der Lösungen eines linearen Gleichungssystems, 103–104
 des Integranden, 47–49
 erweiterte, 47, 104, 208
 im Komplex-Modus, 73–75
 numerischer Rechnungen, 172–211
 Gewichtete Kleinste Quadrate Methode, 111, 115, 143
 Gewichtete Normalgleichungen, 111
 Gleichungen
 Äquivalente, 9–10
 komplexe, Lösen eines großen -systems, 128–131
 Lösen eines nichtlinearen Systems, 122–128
 Lösen ungenauer, 10
 mit mehreren Nullstellen, 10

Gleichungssystem, Lösen, 15–17, 98, 100–101, 122–128
 Gleichungssystem, schlechtkonditioniertes, 104–110
 Gradient, 160, 162
 Größter Abfall, 160
 Komplexes Gleichungssystem, Lösen von, 128–131

H

Hauptwert, 69–72
 Hauptzweig, 69–72
 Horner Schema, 11, 12
 Hyperbolischer Zylinder, 153–154

I

Integral
 Berechnung schwieriger, 55–60
 Konturen-, 85–89
 Uneigentliches, 55–60
 Integration im Komplex-Modus, 73
 Integration, numerische, mit $\frac{f}{x}$, 45–64
 Integrationsintervall, Unterteilung, 50–54, 58
 Interne Verzinsung, 39–44
 Intervallverkleinerung, 161, 162
 Inverse Funktion, 69
 Inverse Iteration, 155
 Inverse Matrix, 98, 101–102, 110, 118, 187
 Inverse Matrix, Rückwärts-Fehleranalyse, 200–204
IRR, 39–44
 Iteration, 103–104, 119–121

J

Jordanform, 155

K

Kartesische Koordinaten, 68
 Kategorie 0, 178
 Kategorie 1, 179–183, 190, 194
 Kategorie 1C, 183
 Kategorie 2, 184–211
 Kategorie ∞ , 179
 Kleinere Nullstelle einer quadratischen Gleichung, 191, 205–211
 Kleinste Quadrate Aufgabe mit Nebenbedingungen, 111, 115–116, 143

Kleinste Quadrate Aufgabe, 110–116, 131–148, 187
 gewichtete, 111, 115, 143
 mit linearer Nebenbedingung, 111, 115–116, 143
 Komplementäre Fehlerfunktion, 60–64
 Komplementäre Normalverteilungsfunktion, 60–64
 Komplex-Modus, 65–95
 Komplex-Modus, Genauigkeit, 73–75
 Komplex-Modus, **SOLVE** und $\left[\frac{x}{y}\right]$, 73
 Komplexe mathematische Funktionen, 68–72
 Komplexe Nullstellen einer Gleichung, 16–17, 80–85
 Komplexe Nullstellen einer quadratischen Gleichung, 205–211
 Komplexe Potentialfunktion, 89–95
 Komplexe Zahl, n -te Wurzeln, 69, 78–80
 Komplexe Zahl, Speichern und Zurückrufen, 76–78
 Komplexes Gleichungssystem, Lösen eines großen, 128–131
 Komponenten, komplexe, 74
 Konditionszahl, 98–102, 107, 201
 Konforme Abbildung, 89
 Konturintegral, 85–89
 Korrekt gerundetes Ergebnis, 179–183
 Korrekt gerundetes Ergebnis, gestörte Eingabe, 184–211
 Kovarianzmatrix, 131
 Kritischer Punkt, 160, 162, 163

L

Lineare Nebenbedingung, Kleinste Quadrate Aufgabe, 111, 115–116, 143
 Lineare Regression, 131. Siehe auch Kleinste Quadrate Aufgabe
 Lineares Gleichungssystem, Genauigkeit der numerischen Lösung, 103–104
 Lineares Modell, 131
 Liniensuche, 161
 Lösen eines großen, komplexen Gleichungssystems, 128–131
 Lösen eines Gleichungssystems, 15–17, 98, 100–101, 118, 122–128
 Lösen eines nichtlinearen Gleichungssystems, 122–128
 Lösungen eines linearen Gleichungssystems, Genauigkeit, 103–104
 LR-Zerlegung, 96–98, 117, 118

M

Mathematische Funktionen, komplexe, 68–72
 Mathematische Funktionen, reine, 47–49
 Mathematisches Modell, 48

Matricelemente, Fehler in, 100–101
 Matrixinversion, Rückwärts-Fehleranalyse, 200–204
 Matrixoperationen, 76–78, 96–171
 Matrixoperationen, Fehlerebenen, 178, 179, 192
 Maximum einer Funktion, 17–25, 160
 Mehrdeutige komplexe Funktionen, 69–72
 Mehrfache Nullstelle, 10
 Mehrwertige Funktionen, komplexe, 69–72
 Minimum einer Funktion, 17–25, 160
 Modell, lineares, 131
 Modell, mathematisches, 48
 Modusanzeige, C, 205
 Modusanzeige, trigonometrische, 68
 Momentaner Nettobarwert, 39–44
 Mondphasen, 186
 Monotonie, 180, 186
 Multiple lineare Regression, 131. Siehe auch Kleinste Quadrate Aufgaben

N

n -te Wurzel einer komplexen Zahl, 69, 78–80
 Netzwerk, Filter, 128–131
 Newton'sche Iterationsverfahren, 80–82, 122
 Nichtlineares Gleichungssystem, Lösung eines, 122–128
 Nichtsinguläre Matrix, 101–102, 117
 Norm, 99, 106, 200
 Normalgleichungen, 110–113, 131–140
 erweiterte, 111
 gewichtete, 111
 Normalverteilungsfunktion, 48, 60–64
 Normalverteilungsfunktion, komplementäre, 60–64
 Nullstellen
 eines Polynoms, 10
 einer komplexen Zahl, 69, 78–80
 einer komplexen Gleichung, 80–85
 einer quadratischen Gleichung, 191, 205–211
 Gleichungen mit mehreren, 10
 komplexe, 16–17
 mehrfache, 10
 nicht gefunden, 9, 29, 92
 ungenau, 9–10

Numerische Bestimmung von Nullstellen, 6, 6–44
 Numerische Integration, 45–64
 Numerische Lösung eines Gleichungssystems, Genauigkeit, 103–104
 Numerische Rechnung, Genauigkeit, 172–211
 Näherung, iterative, 103–104, 119–121

O

Obere Dreiecksmatrix, 96, 113–114, 141
 Optimierung, 160–171
 Orthogonale Faktorisierung, 113–116, 140–148
 Orthogonalmatrix, 113, 141, 142, 149
 Overflow, 179

P

Phasen, des Mondes, 186
 Pi, 173, 174–186
 Pivotelemente, 118
 Polarform, 68
 Polynome, 10–15
 Potentialfunktion, komplexe, 89–95

Q

Quadratische Fläche, 149, 153–154
 Quadratische Gleichung, Nullstellen, 191, 205–211
 Quadratsummen, 132, 140

R

Radiant, im Komplex-Modus, 68
 Ratenzahlungen, 27–28
 Rauschen, Eingabe- und Ausgabe-, 187–192
 Rechenzeit, 49–55
 Rechnungen, numerische Genauigkeit, 172–211
 Regression, multiple lineare, 131. Siehe auch Kleinste Quadrate Aufgaben
 Regressionsquadratsumme, 132–140
 an das Mittel angepaßte, 134
 Relativer Fehler, 173, 182, 183
 einer Matrix, 100
 komplexer, 73–75
 Rendite, 39
 Residuum, 103–104, 110, 132, 201
 Residuumskorrektur, 103–104, 119–121

Residuumsquadratsumme, 132–140
 Resonanz, 46
 Rombergverfahren, 46
 Rundungsfehler, 47, 49, 111, 113, 118, 172–211
 Rückwärts-Fehleranalyse, 187–211

S

Sattelpunkt, 162
 Schiefsymmetrische Matrix, 149
 Schlechtkonditionierte Matrix, 98–102, 107, 155
 Schlechtkonditioniertes Gleichungssystem, 104–110
 Schlußzahlung, 27, 29, 36
 Sekantenverfahren, 7
 Singuläre Matrix, 101–102, 117–118, 201
 Singularitäten und Rückwärts-Fehleranalyse, 192–194
 Skalieren einer Matrix, 104–107, 204
 Skalieren eines Gleichungssystems, 107
SOLVE, 6–44
 Algorithmus, 6–9, 73
 im Komplex-Modus, 73
 Spaltensummennorm, 99
 Speichern komplexer Zahlen, 76–78
 Steigung, 20–22
 Stichproben, **IF**, 46–47, 50, 56, 73
 Stichproben, **SOLVE**, 7–9, 73
 Stromlinie, 89–94
 Störungen der Eingabewerte, 187–192
 Symmetrie des Vorzeichens, 180, 185
 Symmetrische Matrix, 148–149
 System nichtlinearer Gleichungen, Lösen, 122–128

T

Taylorreihe, 182
 Totale Quadratsumme, 132–140
 Trigonometrische Funktionen, 184–186
 Trigonometrische Modi, 68

U

Underflow, 50–51, 118, 179
 Uneigentliche Integrale, 55–60
 Ungenaue Gleichungen, Lösen von, 10
 Ungenaue Nullstellen, 9–10

Untere Dreiecksmatrix, 96

Unterteilen des Integrationsintervalls, 50–54, 58

V

Variable, transformieren von, 54–55

Variablentransformation, 54–55

Varianzanalyse, 133–140

Verbrauchspreisindex, 137–140, 147–148

Vertauschen von Zeilen, 97, 117

Verzweigung, Haupt-, 69–72

Verzögerungsdifferentialgleichung, 81–85

Vorkonditionieren eines Systems, 107–110

Vorzeichenregel von Descartes, 10–11

Vorzeichenwechsel, 8

W

Wiederholte Abschätzung, 23–25

Winkel im Dreieck, 194–199

Z

Zahlung, 26–39

Zahlungsreihen, 26–39

Zahlungsstromdiagramm, 28, 28–44

Zahlungsströme, Analyse diskontierter, 39–44

Zahlungsweise, nachschüssige, 27

Zeilennorm, 99, 200

Zeilenvertauschungen, 97, 117

Zerlegung, LR , 96–98, 117, 118

Zerlegung, LR , Deskriptor, 97

Zinssatz, 26–44

Zukünftiger Wert, 26–39

Zurückrufen komplexer Zahlen, 76–78

Zuwachsrate, 39–44

VERKAUFSNIEDERLASSUGEN:

1000 Berlin 30, Keithstraße 2-4, Tel. (030) 249086
2000 Hamburg 60, Kapstadtring 5, Tel. (040) 63804-0
3000 Hannover 61, Heidering 37-39, Tel (0511) 5706-0
4030 Ratingen 3, Berliner Straße 111, Tel.(02102) 494-0
4600 Dortmund 41, Schleefstraße 28, Tel. (0231) 45001-0
6380 Bad Homburg, Hewlett-Packard-Straße, Tel (06172) 400-0
6800 Mannheim, Rosslauer Weg 2-4, Tel. (0621) 7005-0
7030 Böblingen, Schickardstraße 2, Tel. (07031) 645-0
7517 Waldbronn 2, Ermlis-Allee, Tel. (07243) 601-0
7910 Neu-Ulm, Messerschmittstraße 7, Tel. (0731) 7073-0
8028 Taufkirchen, Eschenstraße 5, Tel. (089) 61207-0
8500 Nürnberg 10, Emmericher Straße 13, Tel. (0911) 5205-0

SERVICENIEDERLASSUNGEN:

6000 Frankfurt 56, Berner Straße 117, Tel. (069) 500001-0



**HEWLETT
PACKARD**