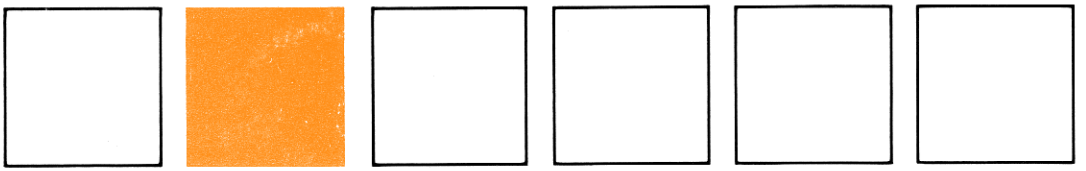
HEWLETT PACKARD

HP 82480AD

Mathematik-Paket

Benutzerhandbuch

Zur Verwendung mit dem HP-71



Hinweis

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

HEWLETT PACKARD

Mathematik-Paket

Benutzerhandbuch

Zur Verwendung mit dem HP-71

Februar 1984

82480-90003

Printed in Singapore

Einleitung

Das Mathematik-Paket beinhaltet einen Satz leistungsfähiger Werkzeuge zur Lösung einer Vielzahl von Problemstellungen aus den Bereichen der Mathematik, Physik und der Ingenieurwissenschaften. Der Zugriff auf diese Werkzeuge ist sehr einfach und bequem, da sie in Form von BASIC-Schlüsselworten zur Verfügung gestellt werden. Sobald das beiliegende Modul in Ihrem HP-71 eingesteckt ist, haben Sie sofort Zugriff auf die durch das Modul implementierten Schlüsselworte; Sie brauchen weder ein Programm zu laden, noch sonstige zeitraubende Operationen auszuführen. Diese Schlüsselworte können innerhalb eines Programms beliebig verwendet werden; Sie vermeiden dadurch die für Programmaufrufe geltenden Beschränkungen und reduzieren die für Unterroutinen erforderlichen Speicherplatzanforderungen.

Das Mathematik-Paket erweitert den Leistungsumfang Ihres HP-71 um die folgenden Funktionen und Operationen:

- Komplexwertige Variablen und Matrizen
- Höhere reell- und komplexwertige Funktionen
- Reell- und komplexwertige Matrizenoperationen
- Lösungen linearer Gleichungssysteme
- Nullstellen von Polynomen und beliebiger reeller Funktionen
- Numerische Integration
- Finite Fouriertransformation

Inhalt

Verwendung dieses Handbuchs

Dieses Handbuch unterstellt, daß Sie mit der Bedienung Ihres HP-71 allgemein vertraut und insbesondere in der Lage sind, Programme einzugeben, zu editieren, zu speichern und auszuführen. Des weiteren sollten Sie die mathematischen Grundlagen der Operationen, die Sie ausführen wollen, verstanden haben. Da die im Mathematik-Paket enthaltenen Funktionen und Operationen relativ viele Teilgebiete der Mathematik abdecken, kann dieses Handbuch aus Platzgründen nicht als Lehrbuch für die jeweils behandelten mathematischen Konzepte dienen.

Die Schlüsselworte des Mathematik-Pakets sind unabhängig voneinander; Sie können sich daher beim Durcharbeiten dieses Handbuchs auf diejenigen Schlüsselworte beschränken, die für Sie von besonderem Interesse sind. Jeder Abschnitt in diesm Handbuch enthält Information über Schlüsselworte, die ein bestimmtes mathematisches Teilgebiet abdecken - reellwertige Funktionen, Matrizenrechnungen usw. Alle nach Abschnitt 5 vorgestellten Schlüsselworte (mit Ausnahme von FNROOT und INTEGRAL) benutzen Felder bei der Ausführung. Die Verwendung von Feldern mit dem HP-71 wird in den Abschnitten 3 und 14 des *HP-71 Benutzerhandbuchs* beschrieben.

Feldtypen

Das Mathematikpaket unterscheidet zwei Typen von Feldern: *Vektoren* und *Matrizen*. In diesem Handbuch entspricht der Begriff Vektor einem einfach indizierten (*eindimensionalen*), der Begriff Matrix einem doppelt indizierten (*zweidimensionialen*) Feld. Indizes müssen reellwertige Ausdrücke sein. Bei der Programmausführung werden Indexausdrücke ganzzahlig gerundet. Der Wert dieser Ganzzahl muß sich im Bereich [0,65535] (OPTION BASE 0) oder [1,65535] (OPTION BASE 1) befinden. In praktisch allen Fällen ist die Anzahl der Elemente eines Felds nur durch die Größe des verfügbaren Speicherplatzes beschränkt.

Felder können vom (Daten-) Typ REAL, SHORT, INTEGER, COMPLEX oder COMPLEX SHORT (siehe unter COMPLEX und COMPLEX SHORT in Abschnitt 3) sein. MAT Anweisungen des Mathematik-Pakets ändern nicht die Typdeklaration eines Felds; bei der Zuweisung von Werten aus einem REAL Feld an ein SHORT oder INTEGER Feld werden die Werte gerundet, bevor sie in dem betreffenden Feld gespeichert werden.

Umdimensionieren von Feldern

Einige Schlüsselworte erlauben ein *optionales* Umdimensionieren eines Felds; diese Art der Umdimensionierung wird im folgenden als *explizite* Umdimensionierung bezeichnet. Andere Schlüsselworte dimensionieren Ergebnisfelder, wenn möglich, *automatisch* um, um die Anzahl der durch die Schlüsselwortoperation erzeugten Elemente zu verarbeiten. Dies wird als *implizite* Umdimensionierung bezeichnet. Die durch ein Schlüsselwort ausgeführte Art der Umdimensionierung, implizit oder explizit, wird in der Beschreibung des Schlüsselworts angegeben.

Eine explizite Umdimensionierung liegt vor, wenn die Größe des Felds und die Anzahl der Indizes durch Vorgabe der Anzahl und des Werts neuer Indizes geändert wird. Die 3 × 4 Matrix **A** vom Typ `REAL` wird beispielsweise mit der HP-71 Anweisung `REAL A(3)` *explizit* in einen 3-dimensionalen Vektor umdimensioniert. Beachten Sie, daß Felder mit expliziter Umdimensionierung von Matrizen in Vektoren und umgekehrt umgewandelt werden können. Bei einer expliziten Umdimensionierung wird auch `OPTION BASE` neu ausgewertet; d.h. die untere Grenze der Feldindizes wird bei veränderter `OPTION BASE` Einstellung zurückgesetzt.

Eine implizite Umdimensionierung liegt bei Operationen des Mathematik-Pakets nur in Form von

`MAT` *Ergebnisfeld = Operation (Operandenfeld(er))*

vor. Eine implizite Umdimensionierung ändert nur die Größe eines Felds und erlaubt weder das Umwandeln von Matrizen in Vektoren und umgekehrt noch wird `OPTION BASE` neu ausgewertet.

Schlüsselworterläuterungen

In jedem Abschnitt dieses Handbuchs wird zur Beschreibung von Name, Aufgabe, Syntax und Arbeitsweise der einzelnen Schlüsselworte das folgende Format verwendet:

Schlüsselwortname — Aufgabe des Schlüsselworts

Syntax Zulässige Datentypen und Wertebereiche für das Schlüsselwort.
Beschreibung der vom Schlüsselwort zurückgegebenen Werte und der allgemeinen Arbeitsweise des Schlüsselworts.

Schlüsselwortname. Innerhalb dieses Handbuchs wird über diesen Namen auf das jeweilige Schlüsselwort Bezug genommen. Der Name ist in der Regel eine mnemonische Umschreibung der von dem Schlüsselwort ausgeführten Funktion. In den meisten Fällen ist der Name in eine längere Anweisung einzubetten, die zusätzlich Argumente, Klammern und ähnliches enthält; der Name allein ist normalerweise keine zulässige BASIC-Anweisung.

Die Namen von mehreren Schlüsselworten sind identisch mit den Namen von Schlüsselworten, die bereits standardmäßig in Ihrem HP-71 vorhanden sind. Beispiele dafür sind DISP, + und *. In diesen Fällen bestimmt die Syntax, in die das Schlüsselwort jeweils eingebettet ist, welche Operation ausgeführt wird. Grundsätzlich sind alle Operationen, die durch den HP-71 selbst angeboten werden, auch nach dem Einsetzen des Mathematik-Moduls verfügbar.

Syntax. Dies ist eine Beschreibung der zulässigen BASIC-Anweisungen, in denen der Name des Schlüsselworts auftreten kann. Auf der nächsten Seite finden Sie eine Beschreibung der zur Erläuterung der Syntax eines Schlüsselworts benutzten Konventionen.

Typographische Darstellung	Bedeutung
Punktmatrix	In Punktmatrix gesetzte Elemente (wie COMPLEX) können in Groß- oder Kleinschreibung eingegeben werden. Die in den Beispielen dieses Handbuchs verwendeten Anweisungen, Funktionen und Operatoren werden in GROSSBUCHSTABEN eingegeben.
kursiv	Kursiv gesetzte Elemente wie *X* in der Anweisung SINH(*X*) sind von Ihnen einzugebende Variablen oder Parameter.
halbfett	In halbfett gesetzte Variablen stellen Felder dar.
[]	Eckige Klammern kennzeichnen optionale Elemente. COMPLEX *Indexgrenze* [, *Liste von Indexgrenzen*] gibt beispielsweise an, daß COMPLEX mehrere, jedoch mindestens eine Dimensionsspezifikation enthalten kann.
übereinander gesetzt	Übereinandergesetzte Elemente deuten an, daß genau eines der angegebenen Elemente auszuwählen ist.
...	Drei Punkte deuten als Wiederholungszeichen an, daß in eckige Klammern gesetzte Elemente wiederholt angegeben werden können. MAT INPUT **A**[,**B**]... gibt beispielsweise an, daß MAT INPUT mindestens eine Variable benötigt; die Anweisung akzeptiert jedoch auch mehrere durch Komma voneinander getrennte Variable.

Zulässige Datentypen und Wertebereiche. Den Angaben in der Syntaxbox können Sie entnehmen, welchen Datentyp ein Wert haben und in welchen Bereich er liegen muß, um als Argument für das Schlüsselwort verwendet werden zu können. Lesen Sie diese Information sorgfältig, um das Auftreten von Fehlern zu vermeiden bzw. um Fehlerursachen aufzuspüren. *Dies ist keine Beschreibung des mathematischen Definitionsbereichs der Funktion, die von dem Schlüsselwort berechnet wird.*

Zurückgegebene Werte und Operationseinzelheiten. Diese in der Box unterhalb der Syntaxbox gemachten Angaben erläutern die Funktion des Schlüsselworts und geben an, welche Werte das Schlüsselwort zurückgibt und ob eine Feldumdimensionierung (falls notwendig) implizit oder explizit durchgeführt wird.

Beispiele

Jeder Abschnitt enthält eine Reihe von Beispielen, die die Verwendung der in dem Abschnitt vorgestellten Schlüsselworte beschreiben. Wenn Sie eine Beispiel nachvollziehen wollen, sollten Sie die unter der Überschrift **Eingabe/Ergebnis** aufgeführten Anweisungen (in Groß- oder Kleinschreibung) eintasten und jede Zeile mit [END LINE] abschließen. Danach sollte die Anzeige Ihres HP-71 wie die unter dem Befehl erscheinende Anzeigeillustration aussehen - sofern Sie die Betriebszustände Ihres HP-71, wie nachstehend angegeben, eingestellt haben.

- Alle, außer den nachstehend genannten Betriebszuständen sollten gemäß den unter Rücksetzbedingungen in Abschnitt "Systemcharakteristika" des Referenzhandbuchs gemachten Angaben eingestellt werden.
- Stellen Sie die Zeilenbreite mit `WIDTH 22` [END LINE] auf 22 Zeichen ein.
- Stellen Sie die Zeitspanne zwischen aufeinanderfolgenden Anzeigen mit `DELAY` so ein, daß jede Anzeigezeile gelesen und verstanden werden kann. Die Anweisung `DELAY` wird im *HP-71 Referenzhandbuch* und in Abschnitt 1 des *HP-71 Benutzerhandbuchs* beschrieben. Dort können Sie nachlesen, wie die Zeitdauer, in der eine Anzeige sichtbar ist, eingestellt wird. Es empfiehlt sich zur Anzeige von Feldelementen die Einstellung `DELAY 8` zu verwenden. Dadurch bleibt jede Anzeige solange sichtbar, bis Sie eine beliebige Taste wie beispielsweise [END LINE] drücken.

Zusätzliche Information

Einige Abschnitte des Mathematik-Pakets enthalten Angaben zur effizienten Verwendung der jeweiligen Schlüsselworte bei der Durchführung anspruchsvollerer Operationen. Sollten Sie trotzdem noch weitere Angaben benötigen, schlagen Sie bitte im Handbuch *HP-15C Fortgeschrittene Funktionen* nach. Obwohl sich das Mathematik-Paket in seiner Arbeitsweise und seinen Fähigkeiten von dem technisch-wissen-schaftlichen Taschenrechner HP-15C unterscheidet, treffen viele der in dem genannten Handbuch enthaltenen Beschreibungen auch auf das Mathematik-Paket zu. Dies gilt insbesondere für die Techniken zur Steigerung der Effizienz der implementierten Algorithmen zur Lösung von Gleichungssystemen, numerischen Integration, Nullstellenbestimmung und zur Durchführung von Matrizenoperationen sowie für die Ausführungen hinsichtlich der Genauigkeit von numerischen Berechnungen.

Abschnitt 1

Einsetzen und Entfernen des Moduls

Sie können das Mathematik-Modul in jeden der vier Einschubschächte auf der Vorderseite des Computers einsetzen.

VORSICHT

- Achten Sie darauf, daß der HP-71 (durch Drücken von [f] [OFF]) ausgeschaltet ist, bevor Sie irgendein Applikations-Modul einsetzen oder entfernen.
- Wenn Sie ein Modul entfernt haben, um das Mathematik-Modul einsetzen zu können, sollten Sie zum Zurücksetzen interner Zeiger den Computer vor dem Einsetzen des Mathematik-Moduls ein- und ausschalten.
- Stecken Sie keine Finger, Werkzeuge oder sonstige Fremdobjekte in die Einschubschächte des Computers. Die Nichtbeachtung dieser Vorsichtsmaßnahme kann zu geringfügigen elektrischen Schlägen und Störungen von Herzschrittmacherfunktionen führen. Des weiteren könnten die Kontakte in den Einschubschächten sowie die internen Schaltkreise des Computers beschädigt werden
- Sollte ein Modul beim Einsetzen klemmen, könnten Sie es verkehrt herum halten. Der Versuch, das Modul mit Gewalt in den Einschubschacht zu drücken, kann zu einer Beschädigung des Computers oder des Moduls führen.
- Behandeln Sie nichteingesetzte Einsteck-Module besonders vorsichtig. Führen Sie keine Gegenstände in die Kontaktbuchsen des Moduls ein. Verschließen Sie ebenso nichtbenutzte Einschubschächte immer mit Modulattrappen. Die Nichtbeachtung dieser Vorsichtsmaßnahmen kann zu einer Beschädigung des Moduls oder des Computers führen.

Zum Einsetzen des Moduls sollten Sie das Modul mit der Beschriftung nach oben halten und dann in den Einschubschacht drücken, bis es einrastet. Dabei sollten die obigen Vorsichtsmaßnahmen beachtet werden.

Zum Entfernen des Moduls ist das Modul mit den Fingernägeln am Griffstück auf der Vorderseite anzufassen und dann aus dem Einschubschacht herauszuziehen. Anschließend ist der Einschubschacht mit einer Modulattrappe zu verschließen, um die Kontakte vor Staub und sonstigen Partikeln zu schützen.

Umwandlung zwischen Zahlensystemen

Binäre, oktale und hexadezimale Darstellung von Daten

Die in diesem Abschnitt beschriebenen Funktionen ermöglichen die Manipulation von Zahlen, die zu anderen Zahlensystemen als dem üblichen Dezimalsystem gehören.

Der HP-71 unterstellt, daß jede in einer numerischen Variablen abgelegte oder über das Tastenfeld eingegebene Zahl eine Zahl aus dem dezimalen Zahlensystem ist; aus diesem Grund muß jede Zahl aus einem anderen Zahlensystem als Zeichenkette (oder String) eingegeben und gespeichert werden. Insbesondere können derartige Zahlen nur in Variablen gespeichert werden, deren Namen mit einem Dollarzeichen ($) enden, und müssen bei der Eingabe über das Tastenfeld in Anführungszeichen gesetzt werden.

In den nachstehenden Tabellen repräsentiert *S$* einen Binär-, Oktal- oder Hexadezimalstring bzw. einen Stringausdruck des entsprechenden Typs.

- Ein *Binärstring* ist eine Zeichenkette, die nur aus den Werten 0 und 1 besteht und eine Zahl aus dem binären (dualen) Zahlensystem (Basis 2) darstellt. Ein *binärer Stingausdruck* ist entsprechend ein Stringausdruck, der bei der Auswertung einen Binärstring liefert.
- Ein *Oktalstring* ist eine Zeichenkette, die nur aus den Werten 0 bis 7 besteht und eine Zahl aus dem oktalen Zahlensystem (Basis 8) darstellt. Entsprechend ist ein *oktaler Stringausdruck* ein Stingausdruck, der bei der Auswertung einen Oktalstring liefert.
- Ein *Hexadezimalstring* besteht aus den Zahlen 0 bis 9 und den Buchstaben A bis F und stellt eine Zahl aus dem hexadezimalen Zahlensystem (Basis 16) dar. (Die Buchstaben können in Groß- oder Kleinschreibung eingegeben werden.) Ein *hexadezimaler Stringausdruck* ist ein Stringausdruck, der bei der Auswertung einen Hexadezimalstring liefert.

Funktionen zur Basisumwandlung

BVAL — Umwandlung von binär, oktal und hexadezimal in dezimal

BVAL(*S$*, *N*)

wo *S$* ein binärer Stingausdruck ist, mit einem Wert nicht größer als 11101000110101001010010100001111111111111 (binär), und *N* ein numerischer Ausdruck, dessen gerundeter Wert 2 ergibt;

oder *S$* ein oktaler Stringausdruck ist, dessen Wert nicht größer als 16432451207777 (oktal) ist, und *N* ein numerischer Ausdruck, dessen gerundeter Wert 8 ergibt;

oder *S$* ein hexadezimaler Stringausdruck ist, dessen Wert nicht größer als E8D4A50FFF (hexadezimal) ist, und *N* ein numerischer Ausdruck, dessen gerundeter Wert 16 ergibt.

BVAL (Fortsetzung)

Wandelt einen Stringausdruck *S$*, der eine Zahl zur Basis *N* repräsentiert, in den äquivalenten Dezimalwert um. Der Wert des Dezimaläquivalents darf nicht größer als 999 999 999 999 (dezimal) sein.

Kann nicht im CALC-Modus verwendet werden.

BSTR$ — Umwandlung von dezimal in binär, oktal oder hexadezimal

```
BSTR$(X,N)
```

wo *X* ein numerischer Ausdruck im Bereich $0 \leqslant X < 999\,999\,999\,999{,}5$ und *N* ein numerischer Ausdruck ist, der bei der Auswertung nach Rundung auf eine ganze Zahl entweder 2, 8 oder 16 ergeben muß.

Wandelt den gerundeten ganzzahligen Wert von *X* (dezimal) in den äquivalenten String zur Basis *N* um.

Bei *N* = 16 werden die *Großbuchstaben* A bis F zurückgegeben.

Kann nicht im CALC-Modus verwendet werden.

Beispiele

Eingabe/Ergebnis

```
BVAL("1010",2) END LINE
```

```
 10
```

Dezimaläquivalent von 1010 (binär).

```
B$="1111" END LINE
BVAL(B$,2) END LINE
```

```
 15
```

Dezimaläquivalent des Binärstings "1111".

```
BVAL(B$&B$,2) END LINE
```

```
 255
```

Dezimaläquivalent des Binärstrings "11111111".

```
BSTR$(3,2) END LINE
```

```
 11
```

Binärdarstellung von 3 (dezimal).

```
BSTR$(72,8) END LINE
```

```
 110
```

Oktaldarstellung von 72 (dezimal).

```
BSTR$(BVAL("AF1C8",16),2)
END LINE
```

```
 10101111000111001000
```

Binärdarstellung von AF1C8 (hexadezimal).

```
BSTR$(BVAL("14772",8)
+BVAL("570",8),8) END LINE
```

```
 15562
```

Oktale Summe von 14772 (oktal) und 570 (oktal).

Weitere Informationen

Bei der Bestimmung des Bereichs der zulässigen Parameter für die drei Schlüsselworte zur Basisumwandlung lagen die folgenden Betrachtungen zugrunde:

- Die Schlüsselworte geben das exakte Resultat für jede ganze Zahl im Bereich der zulässigen Parameter zurück.
- Die Schlüsselworte erzeugen zu einander inverse Abbildungen; d.h. die Hintereinanderausführung liefert in beiden Richtungen für ganze Zahlen jeweils die Identitätsabbildung.
- Die ganzen Zahlen von 0 bis 999 999 999 999 bilden den größten Block von aufeinanderfolgenden, nichtnegativen ganzen Zahlen, die der HP-71 in einem Ganzzahl-Format anzeigen kann.

Abschnitt 3

Komplexe Variablen

Komplexe Daten

Die in diesem Abschnitt beschriebenen Anweisungen und Funktionen dienen zur Deklaration und Manipulation komplexer Zahlen. Folgende Operationen sind verfügbar:

- Deklaration von komplexen Variablen und Feldern mit den Anweisungen `COMPLEX` und `COMPLEX SHORT`.
- Erweiterung der Variablenzuweisungen des HP-71 und der Funktion `RES` auf komplexe Anwendungen.
- Erweiterung des HP-71 Formatstrings `IMAGE` auf komplexe Felder.
- Umwandlung von reellen in komplexe Zahlen.

Deklarieren von komplexen Variablen

COMPLEX

Erzeugen von komplexen Variablen mit 12-stelliger Genauigkeit

`COMPLEX` *Dimensionsspezifikator* [, *Dimensionsspezifikator*]

wo die Syntax von `COMPLEX` der Syntax der Schlüsselworte `REAL`, `SHORT` und `INTEGER` entspricht. *Dimensionsspezifikator* steht hier für *numerische Variable* [(*Dimension 1* [, *Dimension 2*])], wobei *Dimension 1* und *Dimension 2* reelle numerische Ausdrücke sind.

Kann nicht im CALC-Modus verwendet werden.

COMPLEX SHORT

Erzeugen von komplexen Variablen mit 5-stelliger Genauigkeit

`COMPLEX SHORT` *Dimensionsspezifikator* [, *Dimensionsspezifikator*]

wo die Syntax von `COMPLEX SHORT` der Syntax der Schlüsselworte `REAL`, `SHORT` und `INTEGER` entspricht. *Dimensionsspezifikator* steht hier für *numerische Variable* [(*Dimension 1* [, *Dimension 2*])], wobei *Dimension 1* und *Dimension 2* reelle numerische Ausdrücke sind.

Kann nicht im CALC-Modus verwendet werden.

Der von Variablen und Feldern der Typen COMPLEX und COMPLEX SHORT benötigte Speicherplatz wird bei der Ausführung der entsprechenden Deklarationsanweisung zugewiesen; zusätzlich werden nicht zuvor existierende Variablen und die Elemente noch nicht existierender Felder mit dem Wert (0,0) vorbesetzt. Die Auswertung von Feldobergrenzen erfolgt gleichfalls bei der Ausführung der Deklarationsanweisung. Die Untergrenze für jeden Feldindex ist in Abhängigkeit von der bei der Ausführung der Deklarationsanweisungen gültigen OPTION BASE Einstellung entweder 0 oder 1.

Die Anweisung COMPLEX dimensioniert bereits existierende Felder vom Typ COMPLEX um, setzt die Feldelemente jedoch nicht auf (0,0) zurück. Entsprechend dimensioniert die Anweisung COMPLEX SHORT bereits existierende Felder vom Typ COMPLEX SHORT um, setzt die Feldelemente jedoch nicht auf (0,0) zurück. Bei einer Erweiterung eines Felds werden alle neu erzeugten Elemente mit (0,0) vorbesetzt. Bei einer Umdimensionierung bleibt die Reihenfolge der Elemente innerhalb eines Felds erhalten, jedoch nicht notwendigerweise deren Lage innerhalb des Felds. Weitere Informationen können Sie unter "Deklarieren von Feldern (DIM, REAL, SHORT, INTEGER)" in Abschnitt 3 des *HP-71 Benutzerhandbuchs* nachlesen.

In der nachstehenden Tabelle werden die für Variablen und Felder vom Typ COMPLEX und COMPLEX SHORT geltenden Bedingungen angegeben.

Numerische Variablen vom Typ COMPLEX und COMPLEX SHORT

Anfangswert	(0,0)
Numerische Genauigkeit	
COMPLEX	12 Dezimalstellen
COMPLEX SHORT	5 Dezimalstellen
Bereich des Exponenten	±499
Maximale Anzahl der Felddimensionen	2
Maximale Anzahl von Feldelementen	65535
Speicherplatzbelegung von einfachen Variablen (in Bytes)	
COMPLEX	25.5
COMPLEX SHORT	18.5
Speicherplatzbelegung von Feldern	
COMPLEX	16 × (Dimension 1 − OPTION BASE Einstellung + 1) × (Dimension 2 − OPTION BASE Einstellung + 1) + 9.5
COMPLEX SHORT	9 × (Dimension 1 − OPTION BASE Einstellung + 1) × (Dimension 2 − OPTION BASE Einstellung + 1) + 9.5

Operationen mit komplexen Zahlen

(,) — Umwandlung reeller in komplexe Zahlen

(*X*,*Y*)

wo *X* und *Y* reell- oder komplexwertige numerische Ausdrücke sind.

Der HP-71 erkennt eine komplexe Zahl als geordnetes Paar reellwertiger Zahlen. (*X*,*Y*) ist als (Realteil von *X*, Realteil von *Y*) definiert. Daher entspricht (*X*,*Y*) bei komplexwertigem *X* oder *Y* nicht unbedingt

$X + iY$.

Kann im CALC-Modus verwendet werden.

REPT — Realteil einer komplexen Zahl

REPT(*Z*)

wo *Z* ein reell- oder kompexwertiger numerischer Ausdruck ist.

Gibt den Realteil (die erste Komponente) von *Z* zurück. Bei reellwertigem *Z* ist REPT(*Z*) = *Z*.

Kann im CALC-Modus verwendet werden.

IMPT — Imaginärteil einer komplexen Zahl

IMPT(*Z*)

wo *Z* ein reell- oder komplexwertiger numerischer Ausdruck ist.

Gibt den Imagiärteil (die zweite Komponente) von *Z* zurück. Bei reellwertigem *Z* ist IMPT(*Z*) = 0.

Kann im CALC-Modus verwendet werden.

Weitere komplexe Operationen

Das Mathematik-Paket erlaubt die Ausdehnung vieler Operationen des HP-71 auf den komplexen Fall. In Abschnitt 5 wird die Anwendung numerischer Funktionen wie SIN, * usw. auf den komplexen Fall beschrieben. Weitere Erweiterungen umfassen die Zuweisung von Werten auf komplexe Variablen, die Ausführung der Funktion RES, wenn das letzte Ergebnis komplexwertig war, usw. Der HP-71 ist also bei eingestecktem Modul in der Lage, mit komplexen Zahlen in fast der selben Weise zu arbeiten wie mit reellen Zahlen.

Ein weiteres Leistungsmerkmal des Mathematik-Pakets ist die nachstehend beschriebene Erweiterung von Formatstrings auf komplexe Feldspezifikatoren. Zusätzliche Informationen über die Verwendung von Formatstrings können Sie im *HP-71 Referenzhandbuch* unter dem Schlüsselwort IMAGE nachlesen.

C(,) — Komplexe Felder in Formatstrings

[*n*]C(*Formatstring*)

wo *n* ein optionaler Multiplikator ist.

Ein komplexwertiger Ausdruck wird bei der Ausgabe mit DISP oder PRINT dem *Formatstring* entsprechend formatiert. Zuerst wird der Realteil, anschließend der Imaginärteil formatiert. Bei der Ausgabe wird die Zahl in Klammern eingeschlossen, wobei Real- und Imaginärteil durch Komma getrennt werden. Das Komma wird nur bei vorhandenem Imaginärteil gesendet.

Im *Formatstring* darf nicht vorhanden sein:

- Ein Wagenrücklaufzeichen (#)
- Stringfelder
- Eingebettete komplexe Formatstrings

Der *Formatstring* muß *zwei* numerische Spezifikatoren enthalten. Für nichtnumerische Spezifikatoren gelten (außer den oben angegeben) keine Einschränkungen.

Kann nicht im CALC-Modus verwendet werden.

Komplexwertige Ausdrücke in einer DISP USING oder PRINT USING Ausgabeliste dürfen nur über ein in der IMAGE Liste angegebenes komplexes Feld formatiert werden. Reellwertige Ausdrücke in einer DISP USING oder PRINT USING Ausgabeliste dürfen jedoch nicht über ein in der IMAGE Liste angegebenes komplexes Feld formatiert werden.

Beispiele

COMPLEX, COMPLEX SHORT, (,), REPT, IMPT

Eingabe/Ergebnis

```
DESTROY ALL END LINE
```

Stellt sicher, daß keine der in den nachstehenden Anweisungen verwendeten Variablen und Felder existieren. Bereits existierende Felder und Variablen würden bei den anschließenden Feld- und Variablendeklarationen nicht mit (0,0) vorbesetzt werden.

```
COMPLEX Z, W1(3), V(7,7)
END LINE
```

Erzeugt eine komplexe Variable, einen komplexen Vektor und eine komplexe Matrix. Die Variable Z und alle Elemente der Felder W1 und V werden mit (0,0) vorbesetzt.

```
COMPLEX SHORT C(4,7), Y
END LINE
```

Erzeugt ein komplexes Feld und eine komplexe Variable vom Typ SHORT. Y und alle Elemente von C werden mit (0,0) vorbesetzt.

```
Z = (1,SQR(25)) END LINE
```

Weist der Variablen Z die komplexe Zahl $1 + 5i$ zu.

```
Z END LINE
```

```
(1,5)
```

Die Darstellung der komplexen Zahl $1 + 5i$ auf dem HP-71.

```
V(6,5)=3 END LINE
```

Weist dem Feldelement V(6,5) die reelle Zahl 3 zu.

```
V(1,1);V(6,5) END LINE
```

Zeigt die Werte von zwei Feldelementen an.

```
(0,0)  (3,0)
```

Dem komplexen Feldelement V(1,1) wurde bei der Erzeugung der Wert (0,0) zugewiesen. Die reelle Zahl 3 wurde durch die Zuweisung auf ein komplexes Feldelement in die komplexe Zahl (3,0) umgewandelt.

```
Y=((1,2),(3,4)) END LINE
```

Weist Y den komplexen Wert (1,3) zu (wegen (1,3) = (REPT(1,2),REPT(3,4)).

```
Y END LINE
```

Zeigt die komplexe Zahl Y an.

```
(1,3)
```

RES END LINE

Zeigt den Wert des zuletzt ausgeführten oder angezeigten numerischen Ausdrucks an, der in diesem Falle komplexwertig ist.

```
 (1,3)
```

REPT(Y);IMPT(Y) END LINE

```
 1  3
```

Komplexe Formatstrings

Eingabe/Ergebnis

```
5 STD @ COMPLEX Y
10 Y=(69.14,-12.7)
20 DISP USING 100; Y
30 DISP USING 200; Y,Y
40 DISP USING 300; Y,Y
50 DISP USING 400; Y,Y,Y
60 DISP USING "C(DDD,DDD)";Y
100 IMAGE C(2D.2D,4D.2D"i")
200 IMAGE C(4Z,XXX,4*),/,C(4Z,XXX4*)
300 IMAGE C(B,K"i"),X,C(^,4*.2DE)
400 IMAGE 3C(2(DDD,XX))
```

RUN

```
 (69.14, -12.70i)
 (0069   ,-*13)
 (0069,   -*13)
```

Von Zeile 100 erzeugte IMAGE Anzeige.
Von Zeile 200 erzeugte IMAGE Anzeige.

```
(E,-12.7i) (,-127.00E-
001)
```

Von Zeile 300 erzeugte IMAGE Anzeige.

```
( 69   ,-13   )( 69   ,-1
3   )( 69,   -13   )
```

Von Zeile 400 erzeugte IMAGE Anzeige.

```
( 69,-13)
```

Von Zeile 60 erzeugte IMAGE Anzeige.

Abschnitt 4

Reelle Skalarfunktionen

Hyperbolische Funktionen

Die (nachstehend beschriebenen) Funktionen SINH, COSH und TANH sind auch für komplexe Argumente definiert. Siehe Abschnitt 5.

SINH — Sinus Hyperbolicus

SINH(*X*)

wo *X* ein reellwertiger numerischer Ausdruck mit $|X| < 1151.98569368$ ist.

Kann im CALC-Modus verwendet werden.

COSH — Cosinus Hyperbolicus

COSH(*X*)

wo *X* ein reellwertiger numerischer Ausdruck mit $|X| < 1151.98569368$ ist.

Kann im CALC-Modus verwendet werden.

TANH — Tangens Hyperbolicus

TANH(*X*)

wo *X* ein reellwertiger numerischer Ausdruck ist.

Kann im CALC-Modus verwendet werden.

ASINH

Inverser Sinus Hyperbolicus

ASINH(*X*)

wo *X* ein reellwertiger numerischer Ausdruck ist.

Kann im CALC-Modus verwendet werden.

ACOSH

Inverser Cosinus Hyperbolicus

ACOSH(*X*)

wo *X* ein reellwertiger numerischer Ausdruck mit $X \geqslant 1$ ist.

Kann im CALC-Modus verwendet werden.

ATANH

Inverser Tangens Hyperbolicus

ATANH(*X*)

wo *X* ein reellwertiger numerischer Ausdruck mit $-1 < X < 1$ ist.

Kann im CALC-Modus verwendet werden.

Weitere numerische Funktionen

GAMMA

Gamma Funktion

GAMMA(*X*)

wo *X* ein reellwertiger numerischer Ausdruck ist, dessen Bereich wie folgt definiert ist:

X ungleich Null oder einer negativen ganzen Zahl und

$-253 < X < 254.1190554375$

Wie der Graph von GAMMA(X) zeigt, bedingen bestimmte *X*-Werte im Bereich $-263 < X < -253$ eine Bereichsunterschreitung von GAMMA(*X*).

Werte von $X < -263$ und $|\text{GAMMA}(X)| <$ MINREAL bedingen immer eine Bereichsunterschreitung von GAMMA(*X*).

GAMMA (Fortsetzung)

Wenn X eine positive ganze Zahl ist, gilt `GAMMA(X) = FACT(X-1)`.
Allgemein ist `GAMMA(X)` $= \Gamma(X)$ für $X > 0$ als

$$\Gamma(X) = \int_0^\infty t^{X-1} e^{-t} dt$$

und für andere Werte von X durch analytische Fortsetzung definiert.

Kann im CALC-Modus verwendet werden.

LOG2 — Logarithmus zur Basis 2

`LOG2(X)`

wo X ein reellwertiger numerischer Ausdruck mit $X > 0$ ist.

$$\texttt{LOG2(X)} = \log_2(X) = \frac{\ln(X)}{\ln(2)}$$

Kann im CALC-Modus verwendet werden.

SCALE10 — Skalierung mit Zehnerpotenzen

`SCALE10(X,P)`

wo X ein reellwertiger numerischer Ausdruck ist und P ein reellwertiger numerischer Ausdruck ist, der bei der Auswertung eine ganze Zahl ergeben muß.

Multipliziert X mit 10 hoch P durch Addition von P zum Exponenten von X. Bei langen Kettenrechnungen können Bereichsüberschreitungen und -unterschreitungen durch die Verwendung von `SCALE10` vermieden werden.

Kann im CALC-Modus verwendet werden.

Ganzzahlige Rundung

IROUND

Rundung auf eine ganze Zahl

IROUND(*X*)

wo *X* ein reellwertiger numerischer Ausdruck ist.

Rundet *X* unter Verwendung der momentanen OPTION ROUND Einstellung auf eine ganze Zahl.

Kann im CALC-Modus verwendet werden.

Information zurückgebende Funktionen

NAN$

NaN-Ursache

NAN$(*X*)

wo *X* ein reellwertiger numerischer Ausdruck ist.

Gibt einen String zurück, der die im NaN-Argument der Funktion enthaltene Fehlernummer enthält; d.h. es wird die Nummer desjenigen Fehlers zurückgegeben, durch den NaN erzeugt wurde. Der zurückgegebene String entspricht der von der Funktion ERRN zurückgegebenen Nummer. (Siehe *HP-71 Referenzhandbuch*.) Die LEX-Identifikationsnummer ist jedoch 0 für alle von Funktionen des Mathematik-Pakets erzeugte NaN's, da das Mathematik-Paket das Auftreten von NaN's nur durch HP-71 Fehlermeldungen kennzeichnet.

Wenn *X* ungleich NaN ist, gibt NAN$(*X*) einen Nullstring zurück.

Kann nicht im CALC-Modus verwendet werden.

NEIGHBOR

Nächste Maschinenzahl

NEIGHBOR(*X*,*Y*)

wo *X* und *Y* reellwertige numerische Ausdrücke sind.

Gibt die nächste zu *X* in Richtung von *Y* liegende, maschinendarstellbare Zahl zurück. Dies ist in Abhängigkeit von *Y* entweder die nächstkleinere oder nächstgrößere Machinenzahl zu *X*. NEIGHBOR dient speziell zur Auswertung einer Funktion in unmittelbarer Nachbarschaft eines gegebenen Wertes.

Kann im CALC-Modus verwendet werden.

TYPE

Typ und Dimension eines Ausdrucks

TYPE(*X*)

wo *X* ein reell- oder komplexwertiger Ausdruck, ein Stringausdruck oder ein Feld ist.

Gibt wie in nachstehender Tabelle angegeben je nach Typ und Dimension von *X* eine ganze Zahl im Bereich von 0 bis 8 zurück.

Kann außer bei String- und Feldargumenten im CALC-Modus verwendet werden.

X	TYPE(*X*)
Einfach reell (einschließlich einfache Variablen vom Typ INTEGER, SHORT und REAL).	0
Einfach komplex (einschließlich einfache Variablen vom Typ COMPLEX und COMPLEX SHORT.)	1
Einfacher String	2
Feld vom Typ INTEGER	3
Feld vom Typ SHORT	4
Feld vom Typ REAL	5
Feld vom Typ COMPLEX SHORT	6
Feld vom Typ COMPLEX	7
Stringfeld	8

Beispiele

COSH, SINH, ATANH, ACOSH

Eingabe/Ergebnis

COSH(0) END LINE

Cosinus Hyperbolicus einer numerischen Konstanten.

```
 1
```

```
SINH(1/3+2^3) END LINE
```

Sinus Hyperbolicus eines numerischen Ausdrucks.

```
 2080.1308825
```

```
X=9 END LINE
ATANH(1/SQR(X)) END LINE
```

Inverser Tangens Hyperbolicus eines numerischen Ausdrucks mit einer numerischen Variablen.

```
 .34657359028
```

```
ACOSH(COSH(200)) END LINE
```

Inverser Cosinus Hyperbolicus eines numerischen Ausdrucks.

```
 200
```

LOG2, IROUND

Eingabe/Ergebnis

```
LOG2(2^17) END LINE
```

Logarithmus (zur Basis 2) eines numerischen Ausdrucks.

```
 17
```

```
OPTION ROUND NEAR END LINE
IROUND(234.5) END LINE
```

Rundet auf die nächste ganze Zahl (im Zweifelsfall auf die nächste gerade ganze Zahl).

```
 234
```

```
OPTION ROUND POS END LINE
IROUND(234.5) END LINE
```

Rundet auf die nächstgrößere ganze Zahl.

```
 235
```

NAN$, NEIGHBOR, TYPE

Eingabe/Ergebnis

```
TRAP(IVL,2) END LINE
```

Setzt 2 als IVL-Auffangwert. Die Funktion TRAP wird im *HP-71 Referenzhandbuch* beschrieben.

```
SIN(INF) END LINE
```

```
WRN: Invalid Arg
```

Der Wert 2 als IVL-Auffangwert verursacht bei Ausführen der unzulässigen Operation SIN(INF) eine Warnung, jedoch nicht einen Fehler.

```
X END LINE
```

```
NaN
```

Da der IVL-Auffangwert auf 2 eingestellt ist, wird X durch die unzulässige Operation der Wert NaN (*Not-a-Number*) zugewiesen.

```
NAN$(X) END LINE
```

```
11
```

Die mit dem Wert NaN verbundene Meldungsnummer identifiziert die Meldung Invalid Arg.

```
NEIGHBOR(1,5) END LINE
```

```
1.00000000001
```

Die nächstgrößere Maschinenzahl zu 1.

```
NEIGHBOR(1,-10) END LINE
```

```
.999999999999
```

Die nächstkleinere Maschinenzahl zu 1.

```
NEIGHBOR(1E400,1E401) END LINE
```

```
1.00000000001E400
```

Die nächstgrößere Maschinenzahl zu 1E400.

```
NEIGHBOR(1.234E-63,0) END LINE
```

```
 1.23399999999E-63
```

Die nächstkleinere Maschinenzahl zu 1.234E-63.

```
INTEGER I,J(3,9) END LINE
COMPLEX SHORT Z(2),W END LINE
TYPE(2);TYPE(I);TYPE(J);TYPE(Z)
;TYPE(W) END LINE
```

```
 0  0  3  6  1
```

Die von TYPE zurückgegebenen Kennzahlen identifizieren den Typ und die Dimension jedes Ausdrucks.

Abschnitt 5

Komplexe Funktionen und Operationen

Viele mathematische Funktionen sind sowohl für reelle als auch für komplexe Argumente definiert. Das Mathematik-Paket erlaubt die Verwendung vieler HP-71 Schlüsselworte sowohl mit reellen als auch mit komplexen Argumenten. Zusätzlich werden in diesem Abschnitt speziell für komplexe Operationen definierte Schlüsselworte beschrieben.

Alle in diesem Abschnitt beschriebenen Funktionen (außer ABS, ARG, CONJ und den Verhältnisoperatoren) geben ein komplexwertiges Ergebnis zurück.

Es wird angenommen, daß außer bei der Funktion RECT alle komplexen Zahlen Z und W in kartesischer und nicht in polarer Form dargestellt sind.

Aufgrund der zweidimensionalen Natur dieser Funktionen ist es nicht möglich, einfache Schranken für die Funktionsargumente anzugeben, die ein Auftreten von Bereichsunterschreitungen oder Bereichsüberschreitungen verhindern würden.

Operatoren

+ **Addition**

$Z+W$ wo Z und/oder W komplexwertige numerische Ausdrücke sind.
Kann im CALC-Modus verwendet werden.

– **Einwertiges Minus**

$-Z$ wo Z ein komplexwertiger numerischer Ausdruck ist.
Kann im CALC-Modus verwendet werden.

−

Subtraktion

Z−W wo *Z* und/oder *W* komplexwertige numerische Ausdrücke sind.
Kann im CALC-Modus verwendet werden.

*

Multiplikation

Z✳*W* wo *Z* und/oder *W* komplexwertige numerische Ausdrücke sind.
Kann im CALC-Modus verwendet werden.

/

Division

Z⁄*W* wo *Z* und/oder *W* komplexwertige numerische Ausdrücke sind. $W \neq (0,0)$.
Kann im CALC-Modus verwendet werden.

^

Exponentiation

Z^*W* wo *Z* und/oder *W* komplexwertige numerische Ausdrücke sind.
Gibt den Hauptwert von $Z^W = e^{W\ln(Z)}$ zurück. Kann im CALC-Modus verwendet werden.

Logarithmische Funktionen

LOG

Natürlicher Logarithmus

LOG(*Z*) oder LN(*Z*)

wo *Z* ein komplexwertiger numerischer Ausdruck ungleich (0,0) ist.

Wenn $Z = x + iy$ und R (cos θ + i sin θ) die polare Darstellung von Z ist, gilt

$$\text{LOG}(Z) = \ln R + i\theta.$$

wo $-\pi \leqslant \theta \leqslant \pi$ (in Radiant).

Kann im CALC-Modus verwendet werden.

EXP

Exponential

EXP(*Z*)

wo *Z* ein komplexwertiger numerischer Ausdruck ist.

Wenn $Z = x + iy$, dann gilt

$$\text{EXP}(Z) = e^{x + iy} = e^x (\cos y + i \sin y).$$

wo y in Radiant gemessen wird.

Kann im CALC-Modus verwendet werden.

Trigonometrische und hyperbolische Funktionen

Unahängig von der Winkeleinstellung verwenden alle trigonometrischen Funktionen in Radiant angegebene Argumente.

SIN — Sinus

SIN(Z)

wo Z ein komplexwertiger numerischer Ausdruck ist.

Wenn $Z = x + iy$, dann gilt

$$\text{SIN(Z)} = \sin(x + iy) = \sin x \cosh y + i \cos x \sinh y.$$

Kann im CALC-Modus verwendet werden.

COS — Cosinus

COS(Z)

wo Z ein komplexwertiger numerischer Ausdruck ist.

Wenn $Z = x + iy$, dann gilt

$$\text{COS(Z)} = \cos(x + iy) = \cos x \cosh y - i \sin x \sinh y.$$

Kann im CALC-Modus verwendet werden.

TAN — Tangens

TAN(Z)

wo Z ein komplexwertiger numerischer Ausdruck ist.

Wenn $Z = x + iy$, dann gilt

$$\text{TAN(Z)} = \tan(x + iy) = \frac{\sin(x + iy)}{\cos(x + iy)} = \frac{\sin x \cos x + i \sinh y \cosh y}{\sinh^2 y + \cos^2 x}$$

Kann im CALC-Modus verwendet werden.

SINH

Sinus Hyperbolicus

SINH(Z)

wo Z ein komplexwertiger numerischer Ausdruck ist.

Wenn $Z = x + iy$, dann gilt

$$\text{SINH}(Z) = \sinh(x + iy) = (-i)\sin(-y + ix).$$

Kann im CALC-Modus verwendet werden.

COSH

Cosinus Hyperbolicus

COSH(Z)

wo Z ein komplexwertiger numerischer Ausdruck ist.

Wenn $Z = x + iy$, dann gilt

$$\text{COSH}(Z) = \cosh(x + iy) = \cos(-y + ix).$$

Kann im CALC-Modus verwendet werden.

TANH

Tangens Hyperbolicus

TANH(Z)

wo Z ein komplexwertiger numerischer Ausdruck ist.

Wenn $Z = x + iy$, dann gilt

$$\text{TANH}(Z) = \tanh(x + iy) = (-i)\tan(-y + ix).$$

Kann im CALC-Modus verwendet werden.

Umwandlungen zwischen Polar- und Rechteckskoordinaten

POLAR

Rechtecks/Polarumwandlung

POLAR(*Z*)

wo Z ein reell- oder komplexwertiger numerischer Ausdruck ist.

Wenn $Z = x + iy$ und R (cos θ + i sin θ) die Polardarstellung von Z ist, dann gilt

POLAR(*Z*) = (R, θ)

Der Winkel θ wird in Abhängigkeit von der momentanen Winkeleinstellung entweder in Grad ($-180 \leqslant \theta \leqslant 180$) oder in Radiant ($-\pi \leqslant \theta \leqslant \pi$) angegeben.

Kann im CALC-Modus verwendet werden.

RECT

Polar/Rechtecksumwandlung

RECT(*Z*)

wo Z ein reell- oder komplexwertiger numerischer Ausdruck ist.

RECT ist das einzige Schlüsselwort in diesem Abschnitt, das ein Argument Z nur in polarer Form verarbeitet.

Wenn $Z = (R,\theta)$, wo R (cos θ + i sin θ) die Polardarstellung der komplexen Zahl $x + iy$ ist, dann gilt

RECT(*Z*) = $x + iy$

Der Winkel θ wird in Abhängigkeit von der momentanen Winkeleinstellung in Grad oder in Radiant interpretiert.

Kann im CALC-Modus verwendet werden.

Allgemeine Funktionen

SQRT

Quadratwurzel

SQRT(*Z*) oder SQR(*Z*)

wo Z ein komplexwertiger numerischer Ausdruck ist.

Gibt den komplexen Hauptwert der Quadratwurzel von Z zurück.

Kann im CALC-Modus verwendet werden.

SGN

Einheitsvektor

SGN(Z)

wo Z ein komplexwertiger numerischer Ausdruck ist.

Gibt den Einheitsvektor in Richtung von Z zurück, d.h.

$$\text{SGN}(Z) = \frac{Z}{|x + iy|} = \frac{x + iy}{\sqrt{x^2 + y^2}}$$

wo $Z = x + iy$.

Wenn $Z = (0,0)$, dann gilt $\text{SGN}(Z) = Z$.

Kann im CALC-Modus verwendet werden.

ABS

Betrag

ABS(Z)

wo Z ein komplexwertiger numerischer Ausdruck ist.

Wenn $Z = x + iy$, dann gilt

$$\text{ABS}(Z) = |x + iy| = \sqrt{x^2 + y^2}$$

ABS(Z) gibt immer einen reellen Wert zurück.

Kann im CALC-Modus verwendet werden.

ARG

Argument

ARG(*Z*)

wo Z ein reell- oder komplexwertiger numerischer Ausdruck ist.

Wenn $Z = x + iy$ und $R(\cos\theta + i\sin\theta)$ die Polardarstellung von Z ist, dann gilt

$$\text{ARG}(Z) = \theta.$$

Der Winkel θ wird in Abhängigkeit von der momentanen Winkeleinstellung in Grad ($-180 \leq \theta \leq 180$) oder in Radiant ($-\pi \leq \theta \leq \pi$) angegeben.

ARG(*Z*) gibt immer einen reellen Wert zurück.

Kann im CALC-Modus verwendet werden.

CONJ

Komplexe Konjugation

CONJ(*Z*)

wo Z ein reell- oder komplexwertiger numerischer Ausdruck ist.

Wenn $Z = x + iy$, dann gilt

$$\text{CONJ}(Z) = x - iy$$

CONJ(*Z*) gibt immer einen Wert vom gleichen Typ (reell oder komplex) wie Z zurück.

Kann im CALC-Modus verwendet werden.

PROJ

Projektion auf ∞

PROJ(*Z*)

wo Z ein reell- oder komplexwertiger numerischer Ausdruck ist.

Wenn $Z = x + iy$, dann gilt

$$\text{PROJ}(Z) = Z, \text{ falls } \text{ABS}(Z) \neq \text{Inf}$$

oder

$$\text{PROJ}(Z) = \text{Inf} + i0, \text{ falls } \text{ABS}(Z) = \text{Inf}.$$

Kann im CALC-Modus verwendet werden.

Verhältnisoperatoren

=, <, >, #, ?

Gleich oder Ungeordnet

Z Verhältnisoperator W

wo *Z* und/oder *W* komplexwertige numerische Ausdrücke sind.

Wenn mindestens einer der zwei Ausdrücke komplexwertig ist, sind nur zwei Vergleichsergebnisse möglich: Die Ausdrücke sind entweder gleich oder ungeordnet (oder verschieden, was in diesem Fall ungeordnet entspricht).

Es sei $Z = x + iy$ und $W = u + iv$.

Wenn $x = u$ und $y = v$, dann ist jeder Vergleich, der = enthält, wahr (d.h. gibt den Wert 1 zurück).

Wenn $x \neq u$ oder $y \neq v$, dann ist jeder Vergleich, der # oder ? enthält, wahr.

Jeder Vergleich, der < oder > ohne ? oder # enthält, resultiert in einer Ausnahme.

Kann im CALC-Modus verwendet werden.

Beispiele

+, −, *, /

Eingabe/Ergebnis

```
STO @ COMPLEX Z,W
```
END LINE

```
Z=(4,5) @ W=(-3,2)
```
END LINE

```
Z+W
```
END LINE

```
(1,7)
```

```
3+Z+W+1
```
END LINE

```
(5,7)
```

```
Z-W
```
END LINE

```
(7,3)
```

(2,3)*(4,5) END LINE

```
(-7,22)
```

(1,2)/(3,4) END LINE

```
(.44,.08)
```

2/(3,4) END LINE

```
(.24,-.32)
```

^, LOG, EXP

Eingabe/Ergebnis

FIX4 END LINE

(3,4)^(6,9) END LINE

```
(1.3472,3.4565)
```

LOG((1,2)) END LINE

```
(0.8047,1.107)
```

EXP((1,2)) END LINE

```
(-1.1312,2.4717)
```

SIN, TAN, COSH

Eingabe/Ergebnis

```
FIX4 [END LINE]
SIN((21,2)) [END LINE]
```

```
     (3.1477,-1.9865)
```

```
TAN((5,5)) [END LINE]
```

```
     (-4.9401E-5,1.0001)
```

```
COSH((2,3)) [END LINE]
```

```
     (-3.7245,0.5118)
```

ABS, ARG, CONJ, PROJ

Eingabe/Ergebnis

```
FIX4 [END LINE]
ABS((3,4)) [END LINE]
```

```
     5.0000
```

```
DEGREES [END LINE]
ARG((3,4)) [END LINE]
```

```
     53.1301
```

```
RADIANS [END LINE]
ARG((3,-7)) [END LINE]
```

```
    -1.1659
```

Der in Radiant gemessene Winkel im vierten Quadranten, der das Argument der komplexen Zahl $3 - 7i$ ist.

```
STD @ CONJ((1,2)) END LINE
```

```
(1,-2)
```

```
PROJ((-Inf,-Inf)) END LINE
```

```
(Inf,0)
```

```
PROJ((1,2)) END LINE
```

```
(1,2)
```

POLAR, RECT, SGN

Eingabe/Ergebnis

```
STD END LINE
DEGREES END LINE
POLAR(-1) END LINE
```

Rechteck/Polarumwandlung für ein reellwertiges Argument.

```
(1,180)
```

Der Betrag (r) ist 1 und das Argument (θ) ist 180 Grad.

```
FIX4 END LINE
POLAR((3,4)) END LINE
```

Rechteck/Polarumwandlung für eine komplexwertiges Argument.

```
(5.0000,53.1301)
```

Der Betrag (r) ist 5.0000 und das Argument (θ) ist 53.1301 Grad.

```
RADIANS END LINE
RECT((-5,PI/4)) END LINE
```

Polar/Rechteckumwandlung für ein komplexwertiges Argument. Der Betrag (r) ist 5 und das Argument (θ) ist $-3*\pi/4$ Radiant. Der für R angegebene negative Wert stellt die Spiegelung des in Polarkoordinaten angegebenen Punkts $(5,\pi/4)$ am Ursprung dar.

```
(-3.5355,-3.5355)
```

Sowohl Real- (x) als auch Imaginärteil (y) haben den Wert -3.5355.

SGN((1,1)) END LINE

```
(0.7071,0.7071)
```

SQRT, LOG

Beachten Sie das Verhalten an der Sprungstelle in den Zweigen von SQRT und LOG. Die Zweige von komplexen Funktionen werden nachfolgend unter "Weitere Informationen" erläutert.

Eingabe/Ergebnis

SQRT((1,2)) END LINE

```
(1.2720,0.7862)
```

SQRT((-16,0)) END LINE

```
(0.0000,4.0000)
```

SQRT((-16,-0)) END LINE

```
(0.0000,-4.0000)
```

LOG((-EXP(5),0)) END LINE

```
(5.0000,3.1416)
```

LOG((-EXP(5),-0)) END LINE

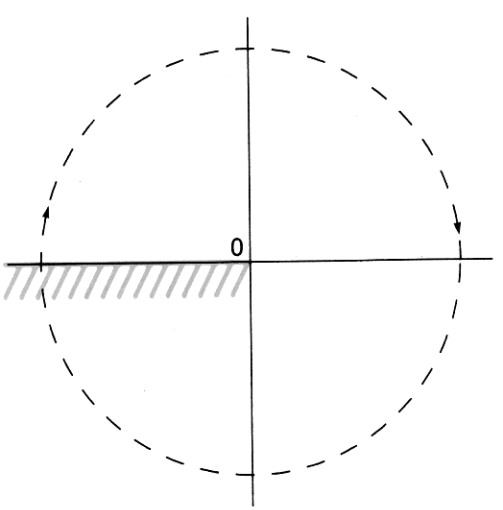
```
(5.0000,-3.1416)
```

Weitere Informationen

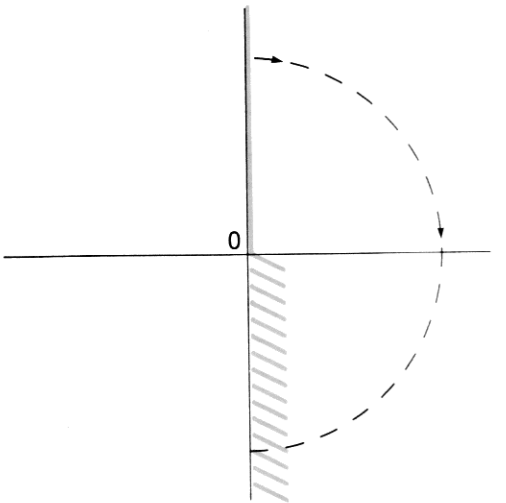
Im allgemeinen hat die Inverse $f^{-1}(z)$ einer Funktion $f(z)$ mehrere Funktionswerte für ein gegebenes Argument z. Das Mathematik-Paket berechnet jedoch für jede gegebene Umkehrfunktion $f^{-1}(z)$ immer den eindeutigen *Hauptwert*, der im als *Hauptzweig* definierten Teil des Wertebereichs der Umkehrfunktion liegt.

Die nachfolgenden Illustrationen zeigen die Hauptzweige der vom Mathematik-Paket berechneten Funktionen SQRT und LOG. Der linke Graph in jeder Abbildung stellt den Definitionsbereich der Umkehrfunktion dar; der rechte Graph zeigt jeweils den Wertebereich für den Hauptzweig. Die blauen und schwarzen Linien im linken Graph werden jeweils unter der Umkehrfunktion auf die entsprechenden blauen und schwarzen Linien im rechten Graph abgebildet.

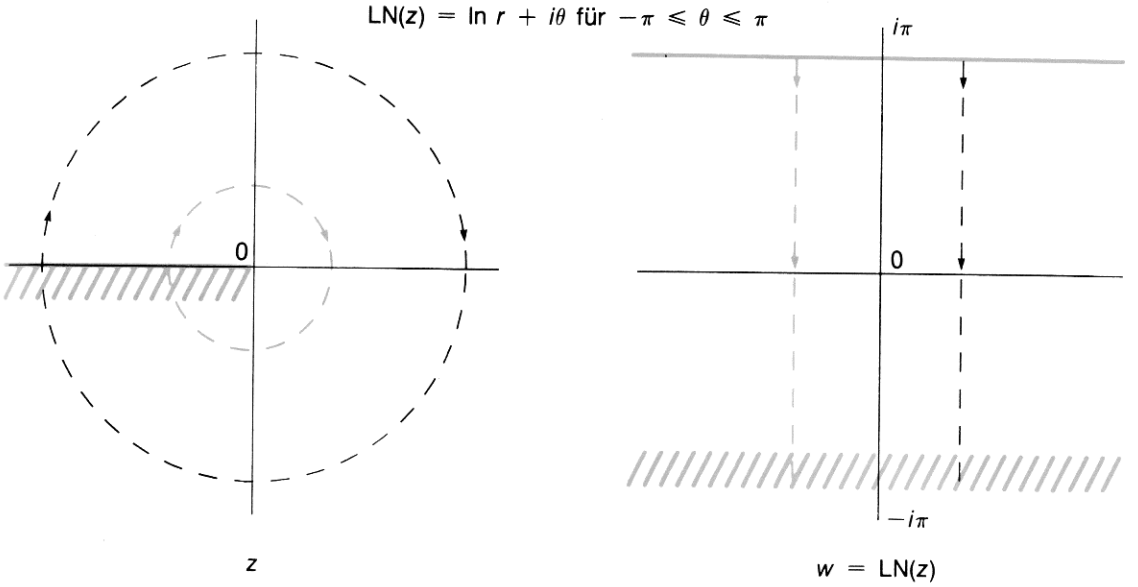
SQRT



z



$w = \sqrt{z}$



Der Hauptzweig von w^z leitet sich aus dem Hauptzweig der Logarithmusfunktion und der Gleichung

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

ab, wo LN die einwertige Funktion bezeichnet.

Wenn Sie *sämtliche* Werte einer Umkehrfunktion bestimmen wollen, können Sie diese mit Hilfe der nachstehenden Ausdrücke von dem vom Mathematik-Paket berechneten Hauptwert ableiten. In diesen Ausdrücken steht k für eine beliebige ganze Zahl. Eine einwertige Funktion wird durch Großbuchstaben gekennzeichnet.

$\sqrt{z} = \pm\text{SQR}(z)$ $\qquad \ln(z) = \text{LN}(z) + 2\pi ik$ $\qquad w^z = w^z e^{2\pi ikz}$

Abschnitt 6

Einlesen und Ausgeben von Feldern

Die in diesem Abschnitt beschriebenen Schlüsselworte ermöglichen die folgenden Operationen:

- Besetzen eines Felds mit Werten
- Anzeigen und Ausdrucken der in einem Feld befindlichen Werte

Wertzuweisungen

= **Einfache Zuweisung**

MAT **A=B**

wo **A** und **B** entweder beide Vektoren oder beide Matrizen sind.
Das Feld **B** kann reell oder komplex sein.
Wenn **B** komplex ist, dann muß **A** komplex sein.
Wenn **B** reell ist, dann kann **A** reell oder komplex sein; bei komplexem **A** werden die Imaginärteile von **A** auf Null gesetzt.

Dimensioniert **A** automatisch auf die Größe von **B** um und weist jedem Element von **A** den Wert des entsprechenden Elements von **B** zu.

Die Operation kann durch zweimaliges Drücken von [ATTN] angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

= ()

Zuweisung von numerischen Ausdrücken

```
MAT A=(X)
```

wo *X* ein reell- oder komplexwertiger numerischer Ausdruck ist.
Wenn *X* komplex ist, dann muß das Feld **A** ebenfalls komplex sein.
Wenn *X* reell ist, dann kann das Feld **A** reell oder komplex sein; bei komplexem **A** werden die Imaginärteile von **A** auf Null gesetzt.

Weist *X* allen Elementen von **A** zu. Das Feld **A** wird nicht umdimensioniert.

Die Operation kann durch zweimaliges Drücken von [ATTN] angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

CON

Initialisieren auf 1

```
MAT A=CON [(X [,Y] )]
```

wo **A** ein reelles oder komplexes ist und die optionalen Umdimensionierungsindizes *X* und *Y* reellwertige numerische Ausdrücke sind. *X* und *Y* werden wie Indizes in DIM Anweisungen auf die nächste ganze Zahl gerundet.

Weist allen Elementen von **A** den reellen Wert 1 zu. Bei Angabe von Umdimensionierungsindizes wird **A** nach Maßgabe dieser Werte explizit umdimensioniert.

Kann nicht im CALC-Modus verwendet werden.

IDN

Einheitsmatrix

```
MAT A=IDN [(X,Y)]
```

wo **A** ein reelles oder komplexes Feld ist und die optionalen Umdimensionierungsindizes *X* und *Y* reellwertige numerische Ausdrücke mit dem gleichen gerundeten ganzzahligen Wert sind. *X* and *Y* werden wie Indizes in DIM Anweisungen auf die nächste ganze Zahl gerundet. **A** muß bei nichtangegebenen *X* und *Y* eine quadratische Matrix sein (d.h. zwei gleiche Indizes haben).

Bei fehlenden Angaben der Indizes *X* und *Y* wird **A** in eine Einheitsmatrix umgewandelt. Bei angegebenen Umdimensionierungsindizes wird **A** explizit in eine quadratische Matrix umdimensioniert, wobei die Obergrenze für jeden Index durch den auf eine ganze Zahl gerundeten gemeinsamen Wert von *X* und *Y* bestimmt wird; anschließend werden dem Feld die Werte einer Einheitsmatrix zugewiesen.

Kann nicht im CALC-Modus verwendet werden.

ZER

Initialisieren auf Null

MAT **A**=ZER [(*X* [,*Y*])] oder MAT **A**=ZERO [(*X* [,*Y*])]

wo **A** ein reelles oder komplexes Feld ist und die optionalen Umdimensionierungsindizes *X* und *Y* reellwertige numerische Ausdrücke sind. *X* and *Y* werden wie Indizes in DIM Anweisungen auf die nächste ganze Zahl gerundet.

Weist jedem Element von **A** den Wert 0 zu. Bei Angabe von Umdimensionierungsindizes wird **A** nach Maßgabe dieser Werte explizit umdimensioniert.

Kann nicht im CALC-Modus verwendet werden.

Einlesen von Feldern

INPUT

Einlesen über das Tastenfeld

MAT INPUT **A** [,**B**]...

wo **A** (und **B**) reelle oder komplexe Felder sind.

Weist den spezifizierten Feldern reelle oder komplexe Werte zu. Komplexe Werte können nicht reellen Feldelementen zugewiesen werden. MAT INPUT fordert Sie durch Anzeige des Namens eines Feldelements zur Eingabe eines numerischen Ausdrucks über das Tastenfeld auf. Anschließend wird dieser Ausdruck ausgewertet und das Ergebnis als Wert dem Feldelement zugeordnet. Für jedes Feld wird zeilenweise (von links nach rechts und von der obersten zur untersten Zeile) zur Eingabe von Werten aufgefordert. Bei Angabe mehrerer Felder werden diese in der spezifizierten Reihenfolge abgearbeitet.

Sobald der Name eines Feldelements angezeigt wird, können Sie den für dieses Element vorgesehenen numerischen Ausdruck eintasten und die Eingabe wie üblich mit [END LINE] abschließen. Sie können gleichzeitig die Werte für mehrere aufeinanderfolgende Feldelemente eingeben, indem Sie die einzelnen Zahlen durch Kommas trennen. Sobald ein Feld gefüllt ist, werden die verbleibenden Werte automatisch in das nächste Feld eingetragen. Nach dem Drücken von [END LINE] zeigt der Computer den Namen des nächsten Feldelements (wenn vorhanden) an, dem ein Wert zuzuweisen ist.

INPUT (Fortsetzung)

Die Arbeisweise von MAT INPUT ist im übrigen mit der von INPUT vergleichbar:

- Der Befehls-Stack ist während der Ausführung von MAT INPUT immer aktiv. Mit [▲], [▼], [g] [⤒], und [g] [⤓] können Sie durch den Befehls-Stack gehen, ohne zuvor [g] [CMDS] gedrückt zu haben.
- Sie können mit einer benutzerdefinierten Direktausführungstaste auf die MAT INPUT Eingabeaufforderung antworten.
- Die Tastenfolgen [f] [VIEW] und [g] [ERRM] sind während der Ausführung von MAT INPUT aktiv.
- Wenn Sie auf eine MAT INPUT Anweisung antworten, können Sie die Eingabe vor dem Drücken von [END LINE] durch einmaliges Drücken von [ATTN] löschen. Wenn Sie [ATTN] zweimal drücken, löscht der HP-71 die Eingabe, hält die Programmausführung an und löscht die Anzeige.

Kann nicht im CALC-Modus verwendet werden.

Ausgeben von Feldern

Die Operation der nachstehend beschriebenen Schlüsselworte kann durch einmaliges Drücken der Taste [ATTN] angehalten werden.

DISP

Anzeige im Standardformat

MAT DISP **A** [, ; **B**] ... [, ;]

wo **A** (und **B**) reelle oder komplexe Felder sind.

Zeigt die Werte der Elemente der spezifizierten Felder an. Die Anzeige erfolgt zeilenweise; jede Feldzeile beginnt auf einer neuen Anzeigezeile. Zusätzlich werden die letzte Zeile eines Feldes und die erste Zeile des nächsten Felds durch eine Leerzeile getrennt.

Der Terminator (Komma oder Semikolon) bestimmt die Abstände zwischen den einzelnen Feldelementen.

Terminator		Abstände zwischen Elementen
;	Eng:	Die einzelnen Elemente werden durch je zwei Leerstellen getrennt. Bei negativen Werten belegt das Minuszeichen die zweite Leerstelle.
,	Breit:	Jedes Element wird in einer aus 21 Spalten bestehenden Anzeigezone abgelegt.

Bei fehlender Angabe eines Terminators für das letzte Feld werden die Elemente dieses Felds mit breiten Abständen angezeigt.

Kann nicht im CALC-Modus verwendet werden.

PRINT

Ausdruck im Standardformat

MAT PRINT **A** [, ; **B**]... [, ;]

wo **A** (und **B**) reelle oder komplexe Felder sind.

Druckt die Werte der Elemente der spezifizierten Felder aus. Die Arbeitsweise von MAT PRINT entspricht der von MAT DISP mit der Ausnahme, daß die Ausgaben an die momentane PRINTER IS Einheit gesendet werden. Zur Deklaration einer PRINTER IS Einheit muß ein HP-IL Interfacemodul HP 82401A in den HP-71 eingesetzt sein. Wenn keine PRINTER IS Einheit deklariert ist, werden die Ausgaben auf die Anzeige oder die momentane HP-IL DISPLAY IS Einheit gelenkt. MAT PRINT sendet standardmäßig am Ende einer Zeile eine Wagenrücklauf/Zeilenvorschub-Sequenz an die PRINTER IS Einheit. Diese Sequenz kann durch die ENDLINE Anweisung modifiziert werden. ENDLINE wird im *HP-71 Referenzhandbuch* und in Abschnitt 13 des *HP-71 Benutzerhandbuchs* beschrieben.

Kann nicht im CALC-Modus verwendet werden.

DISP USING

Anzeige im Benutzerformat

MAT DISP USING *Formatstring* / *Zeilennummer* ; **A** [, ; **B**]... [, ;]

wo **A** (und **B**) reelle oder komplexe Felder sind.

Zeigt die Werte der Elemente der angegebenen Felder in dem durch den *Formatstring* oder die (über die Zeilennummer) spezifizierte IMAGE Anweisung bestimmten Format an. (Eine Diskussion von Formatstrings und Beschreibungen der Anweisungen DISP USING und IMAGE finden Sie im *HP-71 Referenzhandbuch*.

Zur Anzeige komplexer Felder muß der entsprechende Feldspezifikator des *Formatstrings* oder der IMAGE Anweisung komplexwertig sein. Der komplexe Feldspezifikator (C(,)) wird in Abschnitt 3 auf Seite 22 beschrieben.

Die Werte werden zeilenweise angezeigt. Jede Zeile beginnt auf einer neuen Anzeigezeile; die letzte Zeile eines Felds und die erste Zeile des nächsten Felds werden durch eine Leerzeile getrennt.

Die Interpunktionszeichen (Kommata oder Semikolons) zwischen den einzelnen Feldern dienen lediglich als Trennzeichen und haben keinerlei Auswirkung auf das Anzeigeformat.

Das Mathematik-Modul muß zum Umnumerieren (mit RENUMBER) eines Programms, das eine MAT DISP USING [*Zeilennummer*] Anweisung enthält, eingesteckt sein; andernfalls wird die *Zeilennummer* nicht korrekt aktualisiert.

Kann nicht im CALC-Modus verwendet werden.

PRINT USING

Ausdruck im Benutzerformat

MAT PRINT USING *Formatstring* / *Zeilennummer* ; **A** [, | ; **B**] ... [, | ;]

wo **A** (und **B**) reelle oder komplexe Felder sind.

Die Arbeitsweise von MAT PRINT USING entspricht der von MAT DISP USING mit der Ausnahme, daß die auszugebende Information an die momentane PRINTER IS Einheit gesendet wird. Zur Deklaration einer PRINTER IS Einheit muß ein HP-IL Interfacemodul HP 82401A in den HP-71 eingesetzt sein. Wenn keine PRINTER IS Einheit deklariert ist, werden die Ausgaben auf die Anzeige oder die momentane HP-IL DISPLAY IS Einheit gelenkt. MAT PRINT USING sendet standardmäßig am Ende einer Zeile eine Wagenrücklauf/Zeilenvorschub-Sequenz an die PRINTER IS Einheit. Diese Sequenz kann durch die ENDLINE Anweisung modifiziert werden. ENDLINE wird im *HP-71 Referenzhandbuch* und in Abschnitt 13 des *HP-71 Benutzerhandbuchs* beschrieben.

Kann nicht im CALC-Modus verwendet werden.

Beispiele

Wenn Sie die Anzeigeverzögerung auf 8 oder größer setzen, bleibt die momentan angezeigte Zeile unbegrenzt lange in der Anzeige stehen. Erst durch Drücken von [END LINE] (oder einer beliebigen anderen Taste) wird die nächste Zeile angezeigt. Dies gibt Ihnen die Möglichkeit, die Anzeigedauer jeder Feldzeile selbst zu bestimmen.

CON, IDN, ZER, DISP

Eingabe/Ergebnis

```
OPTION BASE 1 @ STD [END LINE]
DIM A(3,3),B(1) [END LINE]
```

B wird als einelementiger Vektor dimensioniert.

```
COMPLEX C(10,20) [END LINE]
MAT A=IDN [END LINE]
MAT DISP A; [END LINE]
```

Zeigt die Einheitsmatrix A mit engem Elementabstand an.

```
 1  0  0
 0  1  0
 0  0  1
```

```
MAT B=ZER(2,2) END LINE
```

Dimensioniert B von einem einelementigen Vektor in eine 2 × 2 Matrix um und initialisiert diese auf Null.

```
MAT DISP B; END LINE
```

```
 0  0
 0  0
```

```
MAT C=CON(3,3) END LINE
```

Dimensioniert C um und belegt C mit der Konstanten 1.

```
MAT DISP C; END LINE
```

```
 (1,0)  (1,0)  (1,0)
 (1,0)  (1,0)  (1,0)
 (1,0)  (1,0)  (1,0)
```

INPUT

Eingabe/Ergebnis

```
OPTION BASE 1 END LINE
DIM A(2,3),B(3) END LINE
OPTION BASE 0 END LINE
COMPLEX C(2,1) END LINE
```

Deklariert C als komplexe 3 × 2 Matrix. (OPTION BASE 0 ist auch Systemvoreinstellung.)

```
MAT INPUT A,B,C END LINE
```

```
A(1,1)? ■
```

Fordert zur Eingabe des ersten Elements auf.

```
1,2,3,4 END LINE
```

Es können mehrere Werte eingegeben werden.

```
A(2,2)? ■
```

Fordert zur Eingabe des fünften Elements auf.

```
5,6,7 END LINE
```

Weist den letzten beiden Elementen von A und dem ersten Element von B Werte zu.

```
B(2)? ■
```

```
8,9,10 END LINE
```

Weist den letzten beiden Elementen von B und dem ersten Element der komplexen Matrix C Werte zu.

```
C(0,1)? ■
```

```
1,2,(5,6),(7,8) END LINE
```

Weist den nächsten vier Elementen von C Werte zu.

```
C(2,1)? ■
```

```
NAN END LINE
```

Weist dem letzten Element von C den Wert NAN zu.

```
STD @ MAT DISP A;B;C; END LINE
```

Zeigt nacheinander jedes Feld durch eine Leerzeile getrennt an.

```
 1   2   3
 4   5   6

 7
 8
 9

(10,0)   (1,0)
(2,0)   (5,6)
(7,8)   (NaN,0)
```

DISP USING

Eingabe/Ergebnis

```
10 OPTION BASE 1 @ INTEGER A(5,5)
15 WIDTH 22 @ DELAY 8
```

Stellt die Anzeige auf die nachstehend dargestellte Anzeigeweise ein. Der Anzeigevorgang kann durch Drücken einer beliebigen Taste (wie END LINE) fortgesetzt werden.

```
20 COMPLEX SHORT Z(3,4)
25 MAT A=IDN @ MAT Z=((4,5))
```

Besetzt A als Einheitsmatrix und weist jedem Element von Z die komplexe Zahl (4,5) zu.

```
30 MAT DISP USING 'DDD,ZZZ';A,A
```

Dieser Formatstring besteht aus zwei Feldspezifikatoren, DDD und ZZZ. Mit Hilfe dieser Feldspezifikatoren werden nacheinander alle Elemente von A angezeigt. Das letzte Element von A wird DDD entsprechend formatiert und angezeigt. Anschließend wird eine Leerzeile und dann alle Elemente von A ein weiteres Mal angezeigt. Dabei wird das erste Element von A über den Feldspezifikator ZZZ (dem nächsten Spezifikator im Formatstring) formatiert.

```
35 MAT DISP USING '#,D';A @ DISP 4
```

Das Symbol # unterdrückt die automatisch zum Abschluß der Anzeige von A erzeugte Wagenrücklauf/Zeilenvorschub-Sequenz. Dadurch wird die Zahl 4 auf der gleichen Zeile angezeigt wie das letzte Element von A.

```
40 MAT DISP USING 100;Z
45 DELAY 1
100 IMAGE C(K,2D,'i')
```

Die Anweisung IMAGE muß zur Formatierung eines komplexen Felds in der Form C(,) aufgebaut sein. Die Klammern müssen zwei numerische Feldspezifikatoren enthalten.

RUN

```
    1000   0000   0
```

Das Formatsymbol D ersetzt führende Nullen durch Leerzeichen. Das Element (1,1) der Einheitsmatrix A ist 1. Daher werden die zwei führenden Nullen durch Leerzeichen ersetzt und das Element (1,1) als 1 angezeigt. Das Formatsymbol Z füllt jede führende Null mit 0 auf, so daß das Element (1,2) als 000 angezeigt wird. Die restlichen Elemente der Zeile werden durch wiederholte Auswertung des Formatstrings DDD,ZZZ formatiert und angezeigt.

```
000   1000   0000
```

Nachdem das letzte (fünfte) Element der ersten Zeile angezeigt ist, wird eine Wagenrücklauf/Zeilenvorschub-Sequenz gesendet, so daß die Anzeige von Element (2,1) mit einer neuen Zeile beginnt.

```
   0000   1000   0
000   0000   1000
   0000   0000   1
```

Der Feldspezifikator DDD bestimmt das Anzeigeformat des letzten Elements von A, so daß 1 angezeigt wird.

Auf das letzte Element der letzten Zeile folgend wird eine Wagenrücklauf/Zeilenvorschub-Sequenz gesendet, so daß zwischen den zwei Anzeigeausgaben von Feld A eine Leerzeile eingefügt wird.

```
001   0000   0000
```

Da die auf den Formatstring in Zeile 30 folgende Variablenliste zweimal das Feld A enthält, wird das Feld A zweimal angezeigt. Bei der zweiten Anzeige von A wird das Element (1,1) dem Spezifikator ZZZ entsprechend angezeigt, da DDD bei der ersten Anzeige von A bereits zur Formatierung des letzten Elements von A verwendet wurde.

```
   0001   0000   0
000   0001   0000
   0000   0001   0
000   0000   0001
```

Dies ist die Anzeige des *letzten* Felds in der Variablenliste von Zeile 30. Obwohl diese Anzeigezeile mit dem letzten Element der letzten Zeile von A endet, wird daher trotzdem keine Leerzeile eingefügt.

```
10000
```

Da derjenige Teil des Formatstrings in Zeile 35, der die Anzeigeweise von Zeichen steuert, nur aus einem D besteht, werden die Elemente einer Zeile von A ohne Zwischenräume und ohne zusätzliche Zeichen hintereinander angezeigt.

```
01000
00010
00001 4
```

Das # Symbol in dem Formatstring in Zeile 35 unterdrückt die normalerweise nach dem Anzeigen der letzten Zeile des letzten Felds in der Variablenliste gesendete Wagenrücklauf/Zeilenvorschub-Sequenz.

```
(4, 5i)(4, 5i)(4, 5i)(
```

Das Symbol K in dem Formatstring in Zeile 100 spezifiziert ein Kompaktfeldformat, bei dem keine führenden oder nachgestellten Leerzeichen angezeigt werden. Das Anzeigeformat der Realteile der (identischen) Elemente von Z wird über dieses Symbol bestimmt. Das Anzeigeformat der Imagnärteile wird über 2D (DD) bestimmmt. Da der Imaginärteil (5) einstellig ist, wird bei der Anzeige ein Leerzeichen vorangestellt. Die Anzeige der Klammern und des Kommas wird durch die Zeichenfolge D(,) erzeugt.

```
4, 5i)
```

Die Anzeige jeder Zeile wird mit einer Wagenrücklauf/Zeilenvorschub-Sequenz abgeschlossen, so daß mit jeder neuen Matrizenzeile eine neue Anzeigezeile begonnen wird.

```
(4, 5i)(4, 5i)(4, 5i)(
4, 5i)
(4, 5i)(4, 5i)(4, 5i)(
4, 5i)
```

Abschnitt 7

Matrizenrechnung

Die nachstehend gelisteten Schlüsselworte führen arithmetische Grundoperationen auf Feldern aus. Dabei ist darauf zu achten, daß die Dimensionen der Operandenfelder mit der jeweiligen Operation kompatibel sind.

- Für Addition und Subtraktion müssen beide Operandenfelder Vektoren oder Matrizen sein und jeweils die gleiche Anzahl von Zeilen und Spalten haben. (Matrizen müssen jedoch nicht notwendigerweise quadratisch sein.) Felder, die diese Anforderungen erfüllen, werden im folgenden als *vereinbar bezüglich Additionen* bezeichnet.
- Für die Multiplikation von zwei Feldern muß das erste Feld eine Matrix sein, während das zweite Feld eine Matrix oder ein Vektor sein kann. Die Anzahl der *Spalten* des ersten Felds muß gleich der Anzahl der *Zeilen* des zweiten Felds sein. Felder, die diese Anforderungen erfüllen, werden im folgenden als *vereinbar bezüglich Multiplikationen* bezeichnet.
- Für die transponierte Multiplikation von zwei Feldern muß das erste Feld eine Matrix sein, während das zweite Feld eine Matrix oder ein Vektor sein kann. Die Anzahl der *Zeilen* des ersten Felds muß gleich der Anzahl der *Zeilen* des zweiten Felds sein. Felder, die diese Anforderungen erfüllen, werden im folgenden als *vereinbar bezüglich transponierter Multiplikationen* bezeichnet.

Operatoren

= –

Negation

MAT **A**=-**B**

wo **A** and **B** entweder beide Vektoren oder beide Matrizen sind.
Das Feld **B** kann reell oder komplex sein.
Wenn **B** komplex ist, dann muß **A** komplex sein.
Wenn **B** reell ist, dann kann **A** reell oder komplex sein; für komplexe Felder **A** werden in diesem Fall die Imaginärteile aller Elemente auf Null gesetzt.

Bedingt eine automatische Umdimensionierung von **A** auf die Größe von **B** und weist jedem Element von **A** den Wert des entsprechenden Elements von **B** mit umgekehrtem Vorzeichen zu.

Die Operation kann durch zweimaliges Drücken von [ATTN] angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

+ Addition

MAT **A=B+C**

wo **A**, **B** und **C** entweder sämtlich Vektoren oder sämtlich Matrizen und **B** und **C** vereinbar bezüglich Additionen sind.
Die Felder **B** und **C** können reell oder komplex sein.
Wenn entweder **B** oder **C** komplex ist, dann muß **A** komplex sein.
Wenn sowohl **B** als auch **C** reell ist, dann kann **A** reell oder komplex sein; für komplexe Felder **A** werden in diesem Fall die Imaginärteile aller Elemente von **A** auf Null gesetzt.

Bedingt eine automatische Umdimensionierung von **A** auf die Größe von **B** und **C** und weist jedem Element von **A** die Summe der entsprechenden Elemente von **B** und **C** zu.

Die Operation kann durch zweimaliges Drücken von [ATTN] angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

− Subtraktion

MAT **A=B−C**

wo **A**, **B** und **C** entweder sämtlich Vektoren oder sämtlich Matrizen und **B** und **C** vereinbar bezüglich Additionen sind.
Die Felder B und C können reell oder komplex sein.
Wenn entweder **B** oder **C** komplex ist, dann muß **A** komplex sein.
Wenn sowohl **B** als auch **C** reell ist, dann kann **A** reell oder komplex sein; für komplexe Felder **A** werden in diesem Fall die Imaginärteile aller Elemente von **A** auf Null gesetzt.

Bedingt eine automatische Umdimensionierung von **A** auf die Größe von **B** und **C** und weist jedem Element von **A** die Differenz der entsprechenden Elemente von **B** und **C** zu.

Die Operation kann durch zweimaliges Drücken von [ATTN] angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

( )* Multiplikation mit einem Skalar

MAT **A**=(*X*)***B**

wo **A** und **B** entweder beide Matrizen oder beide Vektoren sind und *X* ein numerischer Ausdruck ist.
Feld **B** kann reell oder komplex und der Ausdruck *X* reell- oder komplexwertig sein.
Wenn entweder **B** oder *X* komplex ist, dann muß **A** komplex sein.
Wenn sowohl **B** als auch *X* reell ist, dann kann **A** reell oder komplex sein; für komplexe Felder **A** werden in diesem Fall die Imaginärteile aller Elemente von **A** auf Null gesetzt.

Bedingt eine automatische Umdimensionierung von **A** auf die Größe von **B** und weist jedem Element von **A** das Produkt des Werts von *X* und des entsprechenden Elements von **B** zu.

Die Operation kann durch zweimaliges Drücken von [ATTN] angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

* Matrixmultiplikation

MAT **A**=**B***C

wo **B** eine Matrix, **A** und **C** entweder beide Vektoren oder beide Matrizen und **B** und **C** vereinbar bezüglich Multiplikationen sind.
Die Felder **B** und **C** können reell oder komplex sein.
Wenn entweder **B** oder **C** komplex ist, dann muß **A** komplex sein.
Wenn sowohl **B** als auch **C** reell ist, dann kann **A** reell oder komplex sein; für komplexe Felder **A** werden in diesem Fall die Imaginärteile aller Elemente von **A** auf Null gesetzt.

Bedingt eine automatische Umdimensionierung von **A** auf die Anzahl der Zeilen von **B** und die Anzahl der Spalten von **C**. Die Werte der Elemente von **A** werden nach den üblichen Regeln der Matrixmultiplikation gebildet.

Die Operation kann durch zweimaliges Drücken von [ATTN] angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

TRN *

Transponierte Multiplikation

```
MAT A= TRN(B)*C
```

wo **B** eine Matrix, **A** und **C** entweder beide Vektoren oder beide Matrizen und **B** und **C** vereinbar bezüglich transponierter Multiplikationen sind.
Die Felder **B** und **C** können reell oder komplex sein.
Wenn entweder **B** oder **C** komplex ist, dann muß **A** komplex sein.
Wenn sowohl **B** als auch **C** reell ist, dann kann **A** reell oder komplex sein; für komplexe Felder **A** werden in diesem Fall die Imaginärteile aller Elemente auf Null gesetzt.

Bedingt eine automatische Umdimensionierung von **A**, so daß die Anzahl der Zeilen von **A** gleich der Anzahl der Spalten von **B** und die Anzahl der Spalten von **A** gleich der Anzahl der Spalten von **C** ist.

Das Ergebnis dieser Operation ist das gleiche, als wenn zuerst die Transponierte von **B** (oder die konjugiert komplexe Transponierte von **B** bei komplexem **B**) berechnet und anschließend das Ergebnis mit **C** multipliziert wird. Das Mathematik-Paket verwendet jedoch spezielle Multiplikationsregeln, so daß **B** vor der Multiplikation nicht explizit transponiert werden muß.

Die Operation kann durch zweimaliges Drücken von [ATTN] angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

Beispiele

-, *, ( )*, TRN *

Eingabe/Ergebnis

```
OPTION BASE 1 @ STD [END LINE]
REAL A(2,3),B(3,4) [END LINE]
COMPLEX SHORT C(3,1),D(2),E(9)
[END LINE]
MAT A=IDN(2,2) [END LINE]
MAT C=((3,4))*A [END LINE]
```

`C` wird zu einer 2 × 2 Matrix umdimensioniert. Jedem Element von `C` wird das Produkt der komplexen Zahl `(3,4)` mit dem entsprechenden Element von `A` zugewiesen.

```
MAT DISP C; [END LINE]
```

```
 (3,4)   (0,0)
 (0,0)   (3,4)
```

Die Matrix `C`.

```
MAT A=CON @ MAT C=C+A END LINE
```

Weist C die Summe von A und C zu. Eine Umdimensionierung von C ist nicht erforderlich, da C bereits korrekt dimensioniert ist.

```
MAT DISP C; END LINE
```

```
 (4,4)  (1,0)
 (1,0)  (4,4)
```

Die Matrix C.

```
MAT B=A*A END LINE
```

Weist B das Matrixpodukt A*A zu. Dazu wird B zu einer 2 × 2 Matrix umdimensioniert.

```
MAT DISP B; END LINE
```

```
 2  2
 2  2
```

Die Matrix B.

```
MAT INPUT D END LINE
```

```
D(1)? ■
```

```
(1,2),(3,4) END LINE
```

```
MAT E=TRN(C)*D END LINE
```

Weist E das Produkt der konjugiert komplexen Transponierten von C mit dem Vector D zu. Dazu wird E zu einem zweielementigen Vektor umdimensioniert.

```
MAT DISP E END LINE
```

```
(15,8)
(29,6)
```

Die Matrix E.

Abschnitt 8

Skalarwertige Matrixfunktionen

Die in diesem Abschnitt beschriebenen Schlüsselworte repräsentieren Funktionen, die reelle oder komplexe Felder als Argumente verwenden (DET verwendet nur reelle Matrizen) und reelle Zahlen als Ergebnis zurückgeben. (DOT kann sowohl reelle als auch komplexe Zahlen zurückgeben.) Wie alle übrigen Funktionen des HP-71 können diese Funktionen einzeln oder zusammen mit anderen Funktionen zum Aufbau von numerischen Ausdrücken benutzt werden.

Determinantenfunktionen

DET

Determinante

DET(**A**)

wo **A** eine quadratische reelle Matrix ist.

Gibt die Determinante der Matrix **A** zurück.

Die Operation kann durch zweimaliges Drücken von [ATTN] angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

DETL

Determinante der letzten Matrix

DETL oder DET

Gibt die Determinante der letzten reellen Matrix zurück, die

- in einer MAT...INV Anweisung (siehe Abschnitt 9) oder
- als erstes Argument in einer MAT...SYS Anweisung (siehe Abschnitt 9)

spezifiziert wurde. Der von DETL zurückgegebene Wert bleibt (selbst bei ausgeschaltetem HP-71) solange erhalten, bis eine andere MAT...INV Anweisung (mit reellem Argument) oder eine MAT...SYS Anweisung (bei der das erste Argment reellwertig ist) ausgeführt wird.

Kann nicht im CALC-Modus verwendet werden.

Matrixnormen

CNORM

Eins-Norm (Spaltensummennorm)

CNORM(**A**)

wo **A** ein reelles oder komplexes Feld ist.

Gibt das Maximum (über alle Spalten von **A**) der Summen der Beträge aller Elemente in einer Spalte zurück. Die Definition des Betrags einer komplexen Zahl ist unter der Beschreibung des Schlüsselworts ABS auf Seite 45 in Abschnitt 5 zu finden.

Die Operation kann durch zweimaliges Drücken von ATTN angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

RNORM

Unendlich-Norm (Zeilensummennorm)

RNORM(**A**)

wo **A** ein reelles oder komplexes Feld ist.

Gibt das Maximum (über alle Zeilen von **A**) der Summen der Beträge aller Elemente in einer Zeile zurück. Die Definition des Betrags einer komplexen Zahl ist unter der Beschreibung des Schlüsselworts ABS auf Seite 45 in Abschnitt 5 zu finden.

Die Operation kann durch zweimaliges Drücken von ATTN angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

FNORM

Frobenius-Norm (euklidische Norm)

FNORM(**A**)

wo **A** ein reelles oder komplexes Feld ist.

Gibt die Quadratwurzel der Summe der Quadrate der Beträge aller Elemente von **A** zurück. Die Definition des Betrags einer komplexen Zahl ist unter der Beschreibung des Schlüsselworts ABS auf Seite 45 in Abschnitt 5 zu finden.

Die Operation kann durch zweimaliges Drücken von ATTN angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

Punktprodukt

DOT

Punktprodukt (Skalarprodukt)

DOT(**X**,**Y**)

wo **X** und **Y** reelle oder komplexe Vektoren mit der gleichen Anzahl von Elementen sind.

Gibt das Punktprodukt **X**•**Y** der Vektoren **X** und **Y** zurück. Das Ergebnis ist reell, wenn sowohl **X** als auch **Y** reell sind. Das Ergebnis ist komplex, wenn entweder **X** oder **Y** komplex ist.

Bei einem komplexen Vektor **X** werden zur Berechnung des Punktprodukts die konjugiert komplexen Elemente von **X** verwendet.

Die Operation kann durch zweimaliges Drücken von [ATTN] angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

Feldgrenzen

Die nachstehenden Funktionen sind besonders nützlich zur Kontrolle der möglicherweise durch Dimensionieren oder Umdimensionieren von Feldern geänderten OPTION BASE Einstellung, der Anzahl der Dimensionen eines Felds und der Größe in jeder Dimension.

UBND

Feldobergrenze

UBND(**A**,*N*) oder UBOUND(**A**,*N*)

wo **A** ein reelles oder komplexes Feld und *N* ein numerischer Ausdruck ist, dessen auf eine ganze Zahl gerundeter Wert 1 oder 2 ergeben muß.

Gibt die Obergrenze für den *N*-ten (ersten oder zweiten) Feldindex von **A** zurück. Für Vektoren **A** gilt UBND(A,2) = −1.

Kann nicht im CALC-Modus verwendet werden.

LBND — Felduntergrenze

LBND(**A**,*N*) oder LBOUND(**A**,*N*)

wo **A** ein reelles oder komplexes Feld und *N* ein numerischer Ausdruck ist, dessen auf eine ganze Zahl gerundeter Wert 1 oder 2 ergeben muß.

Gibt den Wert der bei der Dimensionierung von **A** gültigen OPTION BASE Einstellung zurück. Für Vektoren **A** gilt LBND(A,2) = −1.

Kann nicht im CALC-Modus verwendet werden.

Beispiele

DET, DOT

Eingabe/Ergebnis

```
OPTION BASE 1 [END LINE]

DIM A(10,10) [END LINE]

MAT A=IDN [END LINE]

MAT A=(-3)*A [END LINE]
```

Weist jedem Diagonalelement den Wert −3 zu; alle anderen Elemente bleiben Null.

```
DET(A) [END LINE]
```

Zeigt die Determinante von A an.

```
 59049
```

```
MAT A=IDN(3,3) [END LINE]

MAT A=(2)*A [END LINE]
```

Weist jedem Diagonalelement den Wert 2 zu; alle anderen Elemente bleiben Null.

```
MAT A=INV(A) [END LINE]
```

Berechnet die Inverse von A.

```
DET [END LINE]
```

Zeigt die Determinante der zuletzt mit einer MAT . . . INV Anweisung invertierten oder als erstes Argument in einer MAT . . . SYS Anweisung verwendeten reellen Matrix an. INV und SYS werden in Abschnitt 9 auf den Seiten 000 bis 000 beschrieben.

```
 8
```

```
DIM A(10),B(10) END LINE
```

```
MAT A=(2) END LINE
```
Weist jedem Element von A den Wert 2 zu.

```
MAT B=CON END LINE
```
Initialisiert die Matrix B auf Eins.

```
DOT(A,B) END LINE
```
Zeigt das Punktprodukt von A und B an.

```
 20
```

```
COMPLEX C(10) END LINE
```

```
MAT C=((1,2)) END LINE
```
Weist jedem Element von C die komplexe Zahl (1,2) zu.

```
DOT(C,A) END LINE
```
Zeigt das Punktprodukt (ein komplexer Wert) von C und A an.

```
 (20,-40)
```

RNORM, CNORM, FNORM, UBND, LBND

Eingabe/Ergebnis

```
OPTION BASE 1 END LINE
```

```
DIM A(3,5) END LINE
```

```
MAT A=CON END LINE
```
Initialisiert die Matrix A auf Eins.

```
RNORM(A) END LINE
```
Zeigt die Zeilensummennorm von A an.

```
 5
```

```
COMPLEX SHORT A(2,4) END LINE

MAT INPUT A END LINE

(1,2),(3,4),(5,6),(7,8),(9,10)
(11,12),(13,14),(15,16) END LINE
```

```
RNORM(A) END LINE
```

Zeigt die Zeilensummennorm von A an.

```
 70.7691300172
```

```
CNORM(A) END LINE
```

Zeigt die Spaltensummennorm von A an.

```
 32.5618580122
```

```
FNORM(A) END LINE
```

Zeigt die Frobenius-Norm von A an.

```
 38.6781592117
```

```
COMPLEX B(3) END LINE

UBND(A,1);UBND(A,2) END LINE
```

Zeigt die Obergrenze des ersten Index und dann die Obergrenze des zweiten Index von A an.

```
 2  4
```

```
UBND(B,1);UBND(B,2) END LINE
```

Zeigt zuerst die Obergrenze des ersten Index von B an und versucht anschließend die Obergrenze des zweiten Index von B anzuzeigen. Da B ein Vektor ist, gibt UBND(B,2) den Wert -1 zurück.

```
 3  -1
```

```
LBND(A,1) END LINE
```

Zeigt den Wert der bei der letzten Dimensionierung von A gültigen OPTION BASE Einstellung an.

```
 1
```

Abschnitt 9

Matrixinversion, -transposition und Gleichungssysteme

Operationen

INV

Matrixinversion

```
MAT A=INV(B)
```

wo **A** und **B** reelle oder komplexe quadratische Matrizen sind, wobei gilt:
Wenn **B** komplex ist, dann muß **A** ebenfalls komplex sein.
Wenn **B** reell ist, dann kann **A** reell oder komplex sein; für komplexe Felder **A** werden in diesem Fall die Imaginärteile aller Elemente auf Null gesetzt.

Bedingt eine automatische Umdimensionierung von **A** auf die gleiche Größe von **B** und weist **A** die Invertierte der Matrix **B** zu.

Die Operation kann durch zweimaliges Drücken von [ATTN] angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

TRN

Transponierte Matrix oder konjugiert komplexe transponierte Matrix

```
MAT A=TRN(B)
```

wo **A** und **B** Matrizen sind. **B** kann reell oder komplex sein; wenn **B** komplex ist, dann muß **A** ebenfalls komplex sein.
Wenn **B** reell ist, dann kann **A** reell oder komplex sein; für komplexe **A** werden in diesem Fall die Imaginärteile aller Elemente auf Null gesetzt.

Bedingt eine automatische Umdimensionierung von **A** auf die Größe von **B**. Wenn **B** reell ist, dann wird **A** die Transponierte der Matrix **B** zugewiesen. Wenn **B** komplex ist, dann wird **A** die konjugiert komplexe Transponierte von **B** zugewiesen.

Die Operation kann durch zweimaliges Drücken von [ATTN] angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

Lösen eines linearen Gleichungssystems

Mit Hilfe des Mathematik-Pakets können Sie auf sehr einfache Weise die Lösung eines Systems von linearen Gleichungen mit reellen und komplexen Koeffizienten exakt bestimmen. Als erster Schritt ist dabei zunächst das Gleichungssystem in ein Tripel von Feldern umzusetzen: Ergebnisfeld, Koeffizientenfeld, Konstantenfeld. Das *Ergebnisfeld* entspricht dabei den Variablen in den Gleichungen; das *Koeffizientenfeld* nimmt die Werte der Koeffizienten der Variablen auf; im *Konstantenfeld* werden die Werte der (konstanten) rechten Seiten der Gleichungen abgelegt. Betrachten Sie beispielsweise das nachstehende Gleichungssystem:

$$5x + 3y + 2z = 4$$

$$7x + y + 3z = 14$$

$$6x + 4y + 9z = 1$$

Hier würde das Ergebnisfeld dem Vektor

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

entsprechen; das Koeffizientenfeld wäre die Matrix

$$\begin{bmatrix} 5 & 3 & 2 \\ 7 & 1 & 3 \\ 6 & 4 & 9 \end{bmatrix}$$

und das Konstantenfeld wäre der Vektor

$$\begin{bmatrix} 4 \\ 14 \\ 1 \end{bmatrix}$$

Wenn das Ergebnisfeld mit **X**, das Koeffizientenfeld mit **A** und das Konstantenfeld mit **B** bezeichnet wird, läßt sich das Gleichungssystem in Matrizenschreibweise formulieren als **AX=B**. Dies ist die vom Schlüsselwort `SYS` benötigte Darstellung.

SYS

Lösen eines linearen Gleichungssystems

MAT**X**=SYS(**A**,**B**)

wo **A** eine quadratische Matrix, **X** und **B** entweder beide Vektoren oder beide Matrizen und **A** und **B** vereinbar bezüglich Multiplikationen sind. Zu Beginn von Abschnitt 7, Seite 63 wird der Begriff "vereinbar bezüglich Multiplikationen" definiert.

Die Felder **A** und **B** können reell oder komplex sein.

Wenn entweder **A** oder **B** komplex ist, dann muß **X** ebenfalls komplex sein.

Wenn sowohl **A** als auch **B** reell sind, dann kann **X** reell oder komplex sein; für komplexes **X** werden in diesem Fall die Imaginärteile aller Elemente auf Null gesetzt.

Bedingt eine Umdimensionierung von **X** auf die Größe von **B** und weist den Elementen von **X** Werte zu, die die Matrizengleichung **AX=B** erfüllen.

Die Operation kann durch zweimaliges Drücken von [ATTN] angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

Beispiele

INV, TRN

Eingabe/Ergebnis

```
OPTION BASE 1 [END LINE]
DIM A(3,3) [END LINE]
MAT A=IDN [END LINE]
MAT A=(2)*A [END LINE]
```

Weist allen Diagonalelementen von A den Wert 2 zu. Alle anderen Elemente sind Null.

```
MAT A=INV(A) [END LINE]
MAT DISP A; [END LINE]
```

Zeigt die Inverse von A an.

```
 .5  0  0
 0  .5  0
 0  0  .5
```

```
DIM C(3,2) END LINE
```

```
MAT C=CON END LINE
```

Initialisiert Matrix C auf Eins.

```
MAT DISP C; END LINE
```

Zeigt C an.

```
 1   1
 1   1
 1   1
```

```
DIM D(2,2) END LINE
```

```
MAT D=TRN(C) END LINE
```

Berechnet die Transponierte von C und dimensioniert D zu einer 2 × 3 Matrix um.

```
MAT DISP D; END LINE
```

Zeigt die Transponierte von C an.

```
 1   1   1
 1   1   1
```

```
COMPLEX SHORT D(2,3),C(3,3)
END LINE
```

```
MAT D=((1,2)) END LINE
```

Weist allen Elementen von D den komplexen Wert (1,2) zu.

```
MAT DISP D; END LINE
```

```
 (1,2)   (1,2)   (1,2)
 (1,2)   (1,2)   (1,2)
```

Die komplexe Matrix D.

```
MAT D=TRN(D) END LINE
```

Dimensioniert D zu einer 3 × 2 Matrix um und weist D die Werte der konjugiert komplexen Transponierten von D zu.

```
MAT DISP D; END LINE
```

```
 (1,-2)   (1,-2)
 (1,-2)   (1,-2)
 (1,-2)   (1,-2)
```

Die konjugiert komplexe Transponierte von D.

```
MAT INPUT C END LINE
```

```
C(1,1)? ■
```

```
1,(1,2),(2,10) END LINE
```

```
C(2,1)? ■
```

```
(1,1),(0,3),(-5,14) END LINE
```

```
C(3,1)? ■
```

```
(1,1),(0,5),(-8,20) END LINE
MAT DISP C; END LINE
```

```
 (1,0)   (1,2)   (2,10)
 (1,1)   (0,3)
 (-5,14)
 (1,1)   (0,5)
 (-8,20)
```

Die komplexe Matrix C.

```
MAT D=INV(C) END LINE
```

Dimensioniert D zu einer 3 × 3 Matrix um und weist D die Werte der Inversen der Matrix C zu.

```
MAT DISP D; END LINE
```

```
 (10,1)   (-2,6)
 (-3,-2)
 (9,-3)
 (-7.09E-11,8)
 (-3,-2)
 (-2,2)   (-1,-2)
 (1,-1.1032E-11)
```

Die exakte Inverse der komplexen Matrix C ist die Matrix

$$\begin{bmatrix} 10+i & -2+6i & -3-2i \\ 9-3i & 8i & -3-2i \\ -2+2i & -1-2i & 1 \end{bmatrix}$$

SYS

Das bereits auf Seite 78 betrachtete Gleichungssystem

$$5x + 3y + 2z = 4$$

$$7x + y + 3z = 14$$

$$6x + 4y + 9z = 1$$

könnte wie folgt gelöst werden.

Eingabe/Ergebnis

```
OPTION BASE 1 @ STD [END LINE]
DIM X(3),B(3),A(3,3) [END LINE]
MAT INPUT B,A [END LINE]
```

```
   B(1)? ■
```

```
4,14,1 [END LINE]
```

Weist den Elementen von **B** Werte zu.

```
   A(1,1)? ■
```

```
5,3,2,7,1,3,6,4,9 [END LINE]
```

Weist den Elementen von **A** Werte zu.

```
MAT X=SYS(A,B) [END LINE]
MAT DISP X [END LINE]
```

Zeigt die Werte des Ergebnisfelds an.

```
 2.55660377358
-2.65094339623
-.415094339623
```

$= x.$
$= y.$
$= z.$

Obwohl das Ergebnisfeld **X** und das Konstantenfeld **B** in typischen Anwendungen Vektoren sind, ist die Verwendung von SYS nicht nur auf einspaltige Felder beschränkt. Dadurch kann eine beliebige, nur durch den vorhandenen Speicherplatz beschränkte Anzahl von Gleichungssystemen mit n Gleichungen und n Unbekannten simulatan gelöst werden, vorausgesetzt, daß die Koeffizienten jedes Gleichungssystems identisch sind. Das nachstehende Beispiel soll diese Verwendung von SYS verdeutlichen.

Beispiel. Die Abteilung für Öffentlichkeitsarbeit der Firma XYZ will die von zwei externen Druckereien verwendeten Kostenfaktoren bestimmen. Es ist bekannt, daß jede der Druckereien einen Auftrag auf der Basis der Anzahl der Seiten und der Anzahl der Illustrationen plus einen Festkostenanteil kalkuliert. Unter Benutzung von jeweils drei Angeboten pro Druckerei (siehe unten) ist ein Programm zu schreiben, das die Kosten pro Seite, pro Illustration und den Festkostenanteil berechnet.

Auftrag	Anzahl der Seiten	Anzahl der Illustrationen	Gesamtkosten	
			Druckerei A	Druckerei B
1	273	35	5835.00 DM	7362.50 DM
2	150	8	3240.00 DM	4085.00 DM
3	124	19	2775.00 DM	3517.50 DM

Zur Schätzung der Kosten ist für jede der Druckereien das folgende Gleichungssystem zu lösen:

$$273x_1 + 35x_2 + x_3 = Angebot_1$$

$$150x_1 + 8x_2 + x_3 = Angebot_2$$

$$124x_1 + 19x_2 + x_3 = Angebot_3$$

Diese Gleichungen lassen sich in Matrizenschreibweise als **AX** = **B** darstellen, wobei:

- **A** die Koeffizientenmatrix mit der Anzahl der Seiten in Spalte 1, der Anzahl der Illustrationen in Spalte 2 und dem Festkostenanteil (jeweils 1) in Spalte 3 ist. Jede Zeile enthält diese Daten für die einzelnen Aufträge.
- **B** die Konstantenmatrix, in diesem Fall ein Feld der Dimension 3 × 2 ist. Jede Zeile enthält die Angebote der beiden Druckereien für die drei Aufträge.
- **X** das Ergebnisfeld mit den unbekannten Kostenfaktoren x_1, x_2 und x_3. Dabei sind x_1 die Kosten pro Seite, x_2 die Kosten pro Illustration und x_3 stellt den Festkostenanteil dar. Da zwei Gleichungssysteme simultan gelöst werden sollen, muß das Ergebnisfeld eine Matrix sein; d.h. es sollte zweidimensional deklariert werden. (Wenn die Größe des Ergebnisfelds nicht mit der Größe der Konstantenmatrix **B** übereinstimmt, wird das Ergebnisfeld automatisch vor der Ausführung von `SYS` auf die Größe von **B** umdimensioniert.) Jede Spalte enthält dann die Kostenfaktoren für eine Druckerei.

```
10 OPTION BASE 1 @ STD
20 DIM A(3,3),X(3,2),B(3,2)
30 DATA 273,35,1          Spezifikationen für Auftrag 1.
40 DATA 150,8,1           Spezifikationen für Auftrag 2.
50 DATA 124,19,1          Spezifikationen für Auftrag 3.
60 DATA 5835,7362.5       Angebote für Auftrag 1.
70 DATA 3240,4085         Angebote für Auftrag 2.
80 DATA 2775,3517.5       Angebote für Auftrag 3.
90 READ A,B
100 MAT X=SYS(A,B)
110 DISP USING '11A,3X,11A,/';
    'DRUCKEREI A','DRUCKEREI B'
120 MAT DISP USING '2X3D.2D,8X,
    3D.2D';X
```

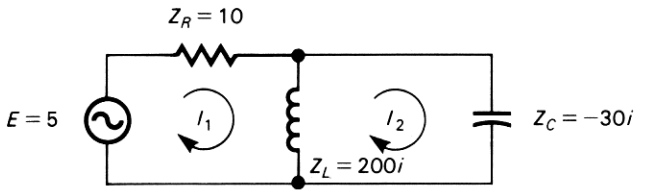
RUN

```
DRUCKEREI A    DRUCKEREI B

   20.00          25.00
    5.00           7.50
  200.00         275.00
```

Festkostenanteil
Kosten pro Seite
Kosten pro Illustration

Beipiel. Dieses Beipiel demonstriert die Anwendung von SYS bei der Berechnung eines Schaltkreises. Die Impedanzen der Schaltelemente in dem nachstehend abgebildeten Schaltkreis sind in komplexer Form angegeben. Die komplexe Darstellung der Stöme I_1 und I_2 soll bestimmt werden.



Dieses System kann durch die folgende komplexe Matrizengleichung

$$\begin{bmatrix} 10+200i & -200i \\ -200i & (200-30)i \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \end{bmatrix} = \begin{bmatrix} 5 \\ 0 \end{bmatrix}$$

oder durch

AX = B

dargestellt werden. Das nachstehende Programm bestimmt I_1 und I_2.

```
10 OPTION BASE 1 @ STD
20 COMPLEX SHORT A(2,2),X(2)

30 DIM B(2)
40 MAT INPUT A,B
50 MAT X=SYS(A,B)
60 MAT DISP X
```

Wenn entweder **A** oder **B** komplex ist, dann muß **X** ebenfalls komplex sein.

RUN

```
A(1,1)? ■
```

(10,200),(0,-200),(0,-200),
(0,170) END LINE

Weist den Elementen von **A** Werte zu.

```
B(1)? ■
```

5,0 END LINE

Weist den Werten von **B** Werte zu.

```
(.037156,.13114)
(.043713,.15428)
```

I_1.
I_2.

Zusätzliche Information

Für reelle quadratische Matrizen **A** benutzen die Operationen DET(**A**), MAT **B**=INV(**A**) und MAT **X**=SYS(**A**,**B**) des Mathematik-Pakets die *LR*-Zerlegung von **A** als Zwischenschritt. Dabei wird das Verfahren von Crout mit partieller Pivotsuche und erhöhter arithmetischer Genauigkeit zur Konstruktion der *LR*-Zerlegung verwendet. Die *LR*-Zerlegung kann durch die Gleichung **PA** = **LR** beschrieben werden, wobei

- **L** eine linke untere Dreiecksmatrix (alle Elemente oberhalb der Diagonalen sind 0) ist.
- **R** eine rechte obere Dreiecksmatrix (alle Elemente unterhalb der Diagonalen sind 0) ist.
- **P** eine Permutationsmatrix ist, die die von der partiellen Pivotsuche herrührende Zeilenvertauschung in der Matrix **A** repräsentiert.

Die Faktorisierung **PA** = **LR** kann auf jede beliebige nichtsinguläre Matrix angewendet werden. Im Falle von singulären oder "maschinensingulären" Matrizen wird die *LR*-Zerlegung geringfügig geändert, wobei der resultierende Fehler klein im Vergleich zum Rundungsfehler ist. Die resultierende *LR*-Zerlegung von **A** stimmt dann fast mit der *LR*-Zerlegung einer anderen Matrix **A'** überein, d.h. die Norm der Matrix **A** entspricht fast der Norm der Matrix **A'**, vorausgesetzt, daß keine Bereichsunterschreitung bzw. -überschreitung eintritt.

Die fast singuläre Matrix

$$\begin{bmatrix} 1 & 3 & 0 \\ 0 & 0 & 1 \\ .666666666667 & 2 & 0 \end{bmatrix}$$

kann mit dem Schlüsselwort INV erfolgreich invertiert werden:

Eingabe/Ergebnis

```
OPTION BASE 1 [END LINE]
DIM A(3,3),B(3,3) [END LINE]
MAT INPUT A [END LINE]
```

```
A(1,1)? ■
```

```
1,3,0,0,0,1 [END LINE]
```

```
A(3,1)? ■
```

```
.666666666667,2,0 END LINE
```

A enthält nun die oben abgebidete Matrix.

```
MAT B=INV(A) END LINE
```

B ist die berechnete Inverse der Matrix **A**.

```
MAT B=B*A END LINE
```

```
MAT DISP B; END LINE
```

Zeigt die Einheitsmatrix **B** an, die hier als Produkt der Matrix **A** mit ihrer Inversen erzeugt wurde.

```
 1   0   0
 0   1   0
 0   0   1
```

Das Schlüsselwort SYS löst die Matrizengleichung **AX** = **B** in mehreren Schritten nach **X** auf. Zuerst wird zur Ermittlung von **PA** = **LR** die *LR*-Zerlegung von **A** gebildet.

Unter Verwendung von **PA** = **LR** stellt sich das Problem als die Auflösung von **LRX** = **PB** nach **X** dar. Dazu wird zunächst **LY** = **PB** nach **Y** (*Vorwärtssubstitution*) und anschließend **RX** = **Y** nach **X** (*Rücksubstitution*) aufgelöst. Der Wert von **X** wird nun als erste Näherung der tatsächlichen Lösung in einer iterativen Verfeinerungsroutine verwendet, die das Endergebnis ermittelt.

In vielen Fällen ermittelt das Mathmatik-Paket eine korrekte Lösung selbst dann, wenn die Koeffizientenmatrix singulär (d.h. die Gleichung $\mathbf{X} = \mathbf{A}^{-1}\mathbf{B}$ nicht erfüllt) ist. Durch diese Eigenschaft sind Sie in der Lage, mit SYS unter- und überbestimmte Gleichungssysteme zu lösen.

In einem unterbestimmten System (mehr Unbekannte als Gleichungen) enthält die Koeffizientenmatrix weniger Zeilen als Spalten. Zur Lösung dieses Systems mit SYS:

- Fügen Sie genügend Nullzeilen an Ihre Koeffizientenmatrix von unten an, so daß Sie eine quadratische Matrix erhalten.
- Fügen Sie entsprechende Nullzeilen an das Konstantenfeld an.

Durch Anwendung des Schlüsselworts SYS auf diese Felder erhalten Sie eine Lösung für das ursprüngliche Gleichungssystem.

In einem überbestimmten System (weniger Unbekannte als Gleichungen) enthält die Koeffizientenmatrix weniger Spalten als Zeilen. Zur Lösung des Problems mit SYS:

- Fügen Sie genügend Nullspalten an Ihre Koeffizientenmatrix von rechts an, so daß Sie eine quadratische Matrix erhalten.
- Stellen Sie sicher, daß Ihr Ergebnisfeld so dimensioniert ist, daß die Zahl der Zeilen mindestens der Zahl der Spalten der neuen Koeffizienmatrix entspricht.

Durch die Anwendung des Schlüsselworts SYS auf diese Felder erhalten Sie eine Lösung für das ursprüngliche Gleichungssystem. Jedoch sind nur diejenigen Elemente des Ergebnisfelds von Bedeutung, die Ihren ursprünglichen Variablen entsprechen.

Sowohl unterbestimmte als auch überbestimmte Systeme haben singuläre Koeffizientenmatrizen. Daher sollten Sie immer prüfen, ob die von SYS zurückgegebenen Ergebnisse die Originalgleichung erfüllen.

MAT **C**=INV(**A**) und MAT **X**=SYS(**A**,**B**) wenden die gleichen, oben genannten Techniken bei einer komplexen quadratischen Matrix **A** an, wobei die Matrizen **A** und **B** durch entsprechende reelle zerlegte Matrizen ersetzt werden.

Das Schlüsselwort SYS kann auch zur Invertierung einer quadratischen Matrix **A** verwendet werden. MAT **X**=SYS(**A**,**B**) gibt die Invertierte von **A** zurück, wenn **X**, **A** und **B** die gleiche Dimension haben und **B** als Einheitsmatrix gewählt wurde. Dieses Verfahren ist im allgemeinen schneller und genauer als MAT **X**=INV(**A**), jedoch benötigt dieses Verfahren mehr Speicherplatz. (Speicherplatzanforderungen werden in Anhang B erläutert.)

Nullstellen einer reellen Funktion

Schlüsselworte

Mit Hilfe der in diesem Abschnitt beschriebenen Schlüsselworte können Sie Nullstellen oder Minima von Funktionen mit bis zu fünf reellen Variablen bestimmen.

Der größte Teil dieses Abschnitts behandelt die Anwendung dieser Schlüsselworte auf Funktionen einer Variablen. Funktionen mehrerer Varaiblen werden unter *Schachtelungsregeln* behandelt.

Das Schlüsselwort FNROOT kann über das Tastenfeld oder in einem Programm zur Bestimmung eines x-Wertes, für den $f(x)$ Null oder ein Minimum ist, verwendet werden, vorausgesetzt, daß die Definition der Funktion zusammen mit dem Schlüsselwort eingegeben wurde oder im Programm enthalten ist.

Die Schlüsselworte FVALUE und FGUESS unterstützen die Verwendung von FNROOT und sind nützlich bei der Interpretation der zurückgegebenen Ergebnisse. Alle drei Schlüsselworte geben einzelne numerische Werte zurück und können daher einzeln oder zusammen mit anderen numerischen Funktionen oder Variablen zu numerischen Ausdrücken kombiniert werden. Ein viertes Schlüsselwort FVAR repräsentiert die Variable in der Funktion, deren Nullstellen mit FNROOT zu bestimmen sind. Des weiteren gibt das Schlüsselwort den während der Ausführung von FNROOT zuletzt berechneten Nährungswert für die Nullstelle zurück.

FNROOT

Nullstellen einer Funktion

FNROOT(*A*,*B*,*F*)
wo *A*, *B* und *F* reelle numerische Ausdrücke sind.

Sucht von den Anfangsnäherungen *A* und *B* ausgehend nach einer reellen Nullstelle der Funktion *F*. Die Anfangsnäherungen können gleich sein; in diesem Fall wird jedoch vor Beginn der Berechnung einer dieser Werte gestört.
Der erste Wert erfüllt eine der folgenden Bedingungen:

1. Der zurückgegebene Wert ist eine exakte Nullstelle der spezifizierten Funktion.
2. Der zurückgegebene Wert ist eine auf 12 Stellen genaue Näherung einer Nullstelle der spezifizierten Funktion.
3. Der zurückgegebene Wert ist eine Näherung für eine lokales Minimum der Absolutwerte der spezifizierten Funktion.
4. Der zurückgegebene Wert ist der Wert der spezifizierten Funktion in einem Bereich, in dem diese konstant ist.
5. Der zurückgegebene Wert ist ±9.99999999999E499, wenn die gesuchte Nullstelle nicht im Bereich der darstellbaren Zahlen gefunden werden konnte.

FNROOT (Fortsetzung)

Kann nicht im CALC-Modus verwendet werden. Zusätzliche Informationen über FNROOT und den CALC-Modus finden Sie auf Seite 97.

Auf den Seiten 97-99 erhalten Sie weitere Information über FNROOT Schachtelungen und die Wechselwirkungen zwischen FNROOT und [ATTN] und zwischen FNROOT und benutzerdefinierten Funktionen.

FVAR

Funktionsvariable

FVAR

Repräsentiert die Variable *x* in *f*(*x*), d.h. die Variable, deren Wert von FNROOT bestimmt wird.

Gibt ebenso den während der Ausführung von FNROOT zuletzt berechneten Näherungswert zurück.

Kann im CALC-Modus verwendet werden.

FVALUE

Funktionswert

FVALUE

Gibt den Wert der Funktion *F* (drittes Argument von FNROOT), der mit der letzten Ausführung von FNROOT berechnet wurde zurück.

Der Wert von FVALUE bleibt (auch nach einem Ausschalten des HP-71) bis zum Abschluß der nächsten Ausführung von FNROOT erhalten.

Kann im CALC-Modus verwendet werden.

FGUESS

Vorletzte Nullstellennäherung

FGUESS

Gibt den in der letzten Ausführung von FNROOT als vorletzte Näherung für die Nullstelle berechneten Wert zurück.

Der Wert von FGUESS bleibt (auch nach einem Ausschalten des HP-71) solange erhalten, bis FNROOT erneut ausgeführt wird.

Kann im CALC-Modus verwendet werden.

Durch Überprüfen der Werte von FVALUE und FGUESS können Sie das Ergebnis von FNROOT wie folgt kontrollieren:

- Wenn FVALUE = 0 gilt, ist das Ergebnis von FNROOT eine exakte Nullstelle der spezifizierten Funktion, und das Ergebnis von FGUESS liegt sehr nahe an der Nullstelle.
- Wenn sich die Ergebnisse von FNROOT und FGUESS nur in der zwölften signifikanten Stelle unterscheiden und FVALUE und F(FGUESS) verschiedene Vorzeichen haben, begrenzen diese beiden Werte die exakte Nullstelle.
- Wenn sich das Ergebnis von FNROOT und das Ergebnis von FGUESS unterscheiden, der Funktionswert FVALUE jedoch gleich dem Wert der Funktion an der Stelle FGUESS ist, liegen beide Ergebnisse in einem Bereich, in dem FNF konstant ist.

Gehen Sie zur Auflösung einer Gleichung nach einer bestimmten Variablen wie folgt vor:

1. Schreiben Sie die zu lösende Gleichung in der Form $f(x) = 0$.
2. Ersetzen Sie in der die Funktion $f(x)$ definierenden Formel die Variable, nach der Sie die Gleichung lösen wollen, durch das Schlüsselwort FVAR.
3. Verwenden Sie die $f(x)$ definierende Formel als drittes Argument von FNROOT.
4. Wählen Sie zwei Anfangswerte (die gleich sein können) und verwenden Sie diese als erste Argumente von FNROOT. Da FNROOT immer drei Argumente benötigt, sollten Sie selbst bei Verwendung von nur einem Anfangswert diesen Wert für A und B einsetzen.

Beispiele

Lösen der Gleichung $x^2 = 2$ (FNROOT, FVALUE, FVAR)

Die folgenden sechs Beispiele verdeutlichen verschiedene Verwendungsmöglichkeiten von FNROOT und FVAR zur Lösung der Funktion $x^2 = 2$. Als Anfangswerte werden 1 und 2 benutzt. Das Ergebnis wird im ersten und sechsten Beipiel angegeben.

Beispiel 1:

Eingabe/Ergebnis

```
FNROOT(1,2,FVAR^2-2) END LINE
```

FNROOT kann sowohl über das Tastenfeld eingegeben als auch in einem Programm verwendet werden.

```
 1.41421356238
```

Beispiel 2:

```
10 DISP FNROOT(COS(0),LOG2(4),
   FVAR^2-2)
20 DISP 'FVALUE =';FVALUE
```

Als Startwert kann auch ein arithmetischer Ausdruck verwendet werden.

Beispiel 3:

```
10 DEF FNG=FVAR^2-2
20 DISP FNROOT(1,2,FNG)
30 DISP 'FVALUE=';FVALUE
```

Das dritte Argument von FNROOT kann ein Ausdruck oder der Aufruf einer benutzerdefinierten Funktion sein.

Beispiel 4:

```
10 DEF FNF(X)=X^2-2
20 DISP FNROOT(1,2,FNF(FVAR))
30 DISP 'FVALUE=';FVALUE
```

FVAR kann in einer benutzerdefinierten Funktion oder, wie oben, als drittes Argument von FNROOT verwendet werden.

Beispiel 5:

```
10 DEF FNH
20 FNH=FVAR^2-2
30 END DEF
40 DISP FNROOT(1,2,FNH)
50 DISP 'FVALUE=';FVALUE
```

Die benutzerdefinierte Funktion kann ein- oder mehrzeilig sein.

Beispiel 6:

```
10 DEF FNJ(X)
20 FNJ=X^2-2
30 END DEF
40 DEF FNF(X)=2*X
50 DISP FNROOT(1,FNF(1),FNJ(FVAR))
60 DISP 'FVALUE =';FVALUE
```

Das erste und zweite Argument von FNROOT kann ebenfalls einen Aufruf einer benutzerdefinierten Funktion enthalten.

Eingabe/Ergebnis

RUN

```
 1.41421356238
FVALUE =  .00000000002
```

Die Lösung von $x^2 = 2$.

Lösen der Gleichung log (x) = e/x (FNROOT, FVALUE, FVAR, FGUESS)

Zur Lösung von $\log(x) = e/x$ ist diese Gleichung erst in die Form $f(x) = 0$ zu bringen. Dazu ist e/x von beiden Seiten der Gleichung zu subtrahieren, was zu $\log(x) - e/x = 0$ führt. Dies ist äquivalent zu $x \log(x) - e = 0$. Da die linke Seite dieser Gleichung für $x \leqslant 0$ nicht definiert ist und der Algorithmus bei der Suche nach einer Nullstelle möglicherweise auch die negative Halbachse erreicht, soll anstelle der obigen Gleichung die Gleichung $|x| \log|x| - e = 0$ gelöst werden. Diese Gleichung hat die gleichen positiven Lösungen wie die erste Gleichung, ist jedoch zusätzlich auch für negative x (jedoch wie die erste Gleichung nicht an der Stelle $x = 0$) definiert. Das nachstehende Programm enthält eine benutzerdefnierte Funktion zur Berechnung der linken Seite der Gleichung und verwendet anschließend FNROOT zur Bestimmung einer Lösung der Gleichung.

```
10 STD @ DESTROY ALL
20 DEF FNF(X)                          Benutzerdefinierte Funktion zur Berechnung der
                                       linken Seite der Gleichung.
30 FNF=ABS(X)*LOG(ABS(X))-EXP(1)
40 END DEF
50 INPUT A,B                           Die beiden Anfangsnäherungen.
60 R=FNROOT(A,B,FNF(FVAR))
70 DISP 'R=';R
80 DISP 'FNF(R)=';FVALUE
90 DISP 'FGUESS=';FGUESS
```

Um dieses Programm verwenden zu können, müssen Sie zwei Anfangsnäherungen vorgeben. Obwohl diese Anfangsnäherungen nicht aufsteigend geordnet oder sogar verschieden sein müssen, führt die Verwendung von Anfangsnäherungen, die die Nullstelle eingrenzen, in der Regel zu einer Verkürzung der benötigten Rechenzeit. In dem hier betrachteten Beispiel ist FNF(FVAR) für $|FVAR| < 1$ negativ und für große |FVAR| (etwa FVAR = 100) positiv; daher sind $x = 0.5$ und $x = 100$ sinnvolle Anfangsnäherungen.

Tasten Sie das Programm ein und starten Sie es durch Drücken von [RUN]; wenn die Eingabeaufforderung ? erscheint, sollten Sie .5,100 als Anfangsnäherungen eingeben und die Eingabe mit [END LINE] abschließen. Der Computer zeigt dann an:

```
R= 2.71828182846
FNF(R)= 0
FGUESS= 2.76000738029
```

Wegen FNF(R) = 0 ist der berechnete Wert eine exakte Nullstelle von FNF.

Weitere Informationen

Wahl der Anfangsnäherungen

Bei der Bestimmung von Nullstellen mittels FNROOT legen die Anfangsnäherungen fest, wo die Suche nach einer Nullstelle begonnen werden soll. Wenn die beiden Anfangsnäherungen eine ungerade Anzahl von Nullstellen begrenzen (was durch unterschiedliche Vorzeichen der Funktionswerte der beiden Näherungen gekennzeichnet ist), findet FNROOT relativ schnell eine Nullstelle zwischen den gegebenen Näherungen. Wenn die Funktionswerte der beiden Näherungen gleiche Vorzeichen besitzen, muß FNROOT zunächst nach einem Bereich suchen, in dem eine Nullstelle liegt. Die Auswahl von Anfangsnäherungen, die möglichst dicht an einer Nullstelle liegen, führt zu einer Beschleunigung dieser Suche. Wenn Sie lediglich das Verhalten der Funktion in der Nähe der Anfangsnäherungen untersuchen wollen (etwa ob in diesem Bereich irgendwelche Nullstellen oder lokale Extrema liegen), können Sie beliebige Anfangsnäherungen vorgeben.

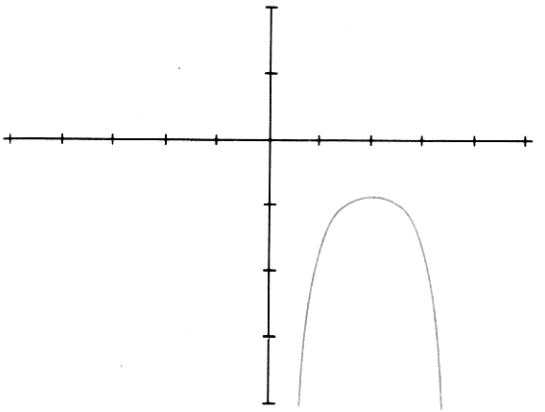
Des weiteren sollte bei der Auswahl von Anfangsnäherungen ebenfalls mit in Betrachtung gezogen werden, in welchem Bereich die Gleichung sinnvoll ist. Bei der Lösung von $f(x) = 0$ hat die Variable x möglicherweise nur einen eingeschränkten Bereich, in dem eine Lösung logisch sinnvoll ist. In diesem Fall sollten die Anfangsnäherungen sinnvollerweise aus diesem Bereich gewählt werden. Häufig hat eine Gleichung, die ein physikalisches Modell repräsentiert, zusätzlich zu der gewünschten Lösung auch noch weitere Lösungen, die physikalisch nicht sinnvoll sind. Diese Situation ist insbesondere dann gegeben, wenn die zu analysierende Gleichung das physikalische Modell nur für bestimmte Bereiche der Variablen beschreibt. Sie sollten bei der Interpretation von Ergebnissen derartige Einschränkungen berücksichtigen.

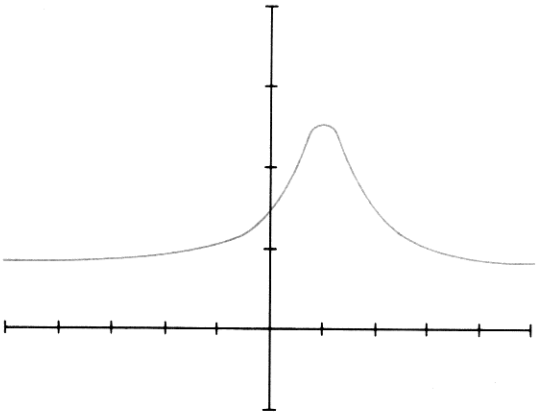
Interpretieren von Ergebnissen

Bei der Verwendung von FNROOT sollten Sie immer die Funktion an der zurückgegebenen Nullstelle in der zuvor beschriebenen Weise auswerten. Auf diese Weise können Sie das von FNROOT zurückgegebene Ergebnis interpretieren. Bei der Auswertung der Funktion an der von FNROOT zurückgegebenen Nullstelle lassen sich zwei Fälle unterscheiden: 1) Der Funktionswert liegt dicht an Null; 2) der Funktionswert liegt nicht dicht an Null. Es ist von Ihrer jeweiligen Problemstellung abhängig, wie dicht der Funktionswert an Null liegen muß, um das Ergebnis von FNROOT als Nullstelle zu akzeptieren.

Wenn der Funktionswert zu groß ist, können Sie anhand der obigen Betrachtungen und der von FGUESS zurückgegebenen Information das allgemeine Verhalten der Funktion in diesem Bereich erkennen. Es sei beispielsweise unterstellt, daß Sie mittels FNROOT eine Nullstelle einer Funktion $f(x)$ finden wollen, und daß der Funktionswert $f(r)$ des von FNROOT zurückgegebenen Werts r zu groß ist, um r als Nullstelle akzeptieren zu können. In diesem Fall kann sich die Funktion in dem betrachteten Bereich wie folgt verhalten:

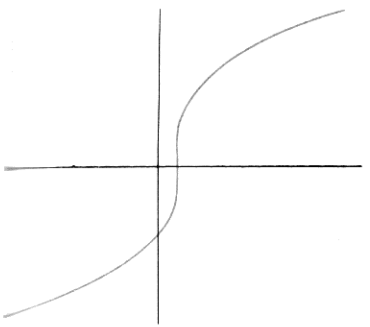
Wenn FVALUE und f(FGUESS) beide das gleiche Vorzeichen besitzen, ist r entweder eine Näherung für ein lokales Minimum von $|f(x)|$ oder r liegt in einem Bereich, in dem der Graph der Funktion horizontal verläuft (d.h. die Funktionswerte konstant sind).

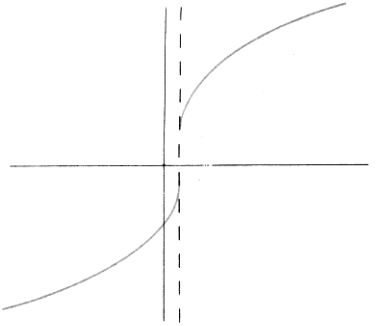


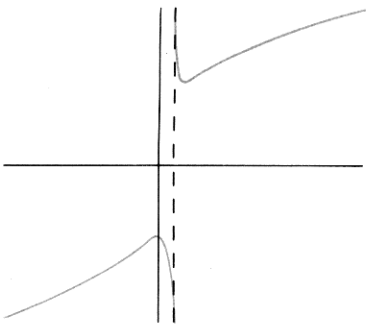


In diesen beiden Fällen kann FNROOT keine abnehmende Tendenz in den Absolutwerten feststellen, d.h. die Funktion strebt nicht gegen die x-Achse. FNROOT versucht dann, einen lokalen Extremwert anzunähern (sofern ein solcher vorhanden ist). Sie können diese Näherungen anschließend verfeinern, indem Sie wiederholt FNROOT mit r und FGUESS als Anfangsnäherungen ausführen. Die wiederholte Ausführung von FNROOT auf diese Weise führt in vielen Fällen zu einer Konvergenz gegen den Extremwert. Die zugrundeliegende Idee ist, daß Sie entweder mittels FNROOT lokale Extrema auffinden können oder daß Sie die Information über die Lage von lokalen Extrema dazu benutzen, die Suche nach einer Nullstelle in andere Bereiche zu lenken.

Wenn |FVALUE| zu groß ist, um r als Nullstelle zu akzeptieren, bleibt als zweite Möglichkeit, daß FVALUE und f(FGUESS) unterschiedliche Vorzeichen haben. Auf den ersten Blick sollte in diesem Fall die Funktion eine Nullstelle zwischen diesen Werten haben, da normalerweise erst ein Überschreiten der x-Achse einen Vorzeichenwechsel in den Funktionswerten bedingt. Bei zwei Näherungen auf gegenüberliegende Seiten der x-Achse verfeinert FNROOT diese Näherungen solange, bis die letzte und die vorletzte Näherung zwei aufeinanderfolgende Maschinenzahlen darstellen. In diesem Fall existiert keine maschinendarstellbare Zahl zwischen r und FGUESS; das Verhalten der Funktion zwischen diesen Punkten kann folglich nicht untersucht werden. Die drei folgenden Graphen illustrieren mögliche Ursachen für eine derartige Situation:







In Fall 1 liefern r und FGUESS die besten Näherungen für eine Nullstelle, die auf der Maschine darstellbar sind. Für FNROOT unterscheidet sich Fall 2 nicht von Fall 1; hier existiert jedoch keine Nullstelle, da eine Sprungstelle der Funktion vorliegt. Fall 3 zeigt eine Pol, der wiederum wie eine Nullstelle aussehen kann, wenn Anfangsnäherungen auf beiden Seiten des Pols gewählt werden. Anhand der von FNROOT und FGUESS zurückgegebenen Information können Sie normalerweise feststellen, ob eine Konvergenz gegen einen Pol oder eine Sprungstelle vorliegt.

Verringern der Ausführungszeiten

Aufgrund des großen Exponentenbereichs des HP-71 von ±499 (TRAP(UNF) = 2 dehnt den Bereich des negativen Exponenten sogar auf −510 aus) sind sehr genaue Untersuchungen über das Verhalten einer Funktion möglich. Dies gilt selbst in extremer Nähe einer Nullstelle. FNROOT macht sich diesen dynamischen Zahlenbereich zunutze, indem eine Näherung nur dann als Nullstelle akzeptiert wird, wenn der Funktionswert an dieser Stelle Null ist oder einen Bereichsunterlauf bedingt, oder bis zwei aufeinanderfolgende Maschinenzahlen gefunden sind, die die Nullstelle eingrenzen. Diese hohe Genauigkeit hat natürlich einen Preis; es kann gelegentlich vorkommen, daß die Bestimmung einer Nullstelle auf alle zwölf Stellen einen gewissen Rechenzeitaufwand bedingt. Es kann nun Situationen geben, in denen Sie die volle Rechengenauigkeit nicht benötigen und daher eine größere Fehlerschranke vorziehen würden. Wenn Sie beispielsweise lediglich wissen wollen, an welchen Stellen eine Funktion kleiner als 1E−20 ist,können Sie Ihre benutzerdefinierte Funktion so abändern, daß der Funktionswert vor der Wertzuweisung an die Funktionsvariable auf die Fehlerschranke abgeprüft und gegebenenfalls durch Null ersetzt wird. Es sei zum Beispiel unterstellt, daß Sie alle Nullstellen der Funktion $f(x) = x^4$ bestimmen wollen und daß Sie einen Funktionswert $|f(x)| \leq 1E-32$ als Kriterium für eine Nullstelle akzeptieren. In diesem Fall könnten Sie das nachstehende Programm verwenden.

```
10 STD @ DESTROY ALL
20 DEF FNF(X)                          Mehrzeilige Funktion für f(x) = x^4.
30 F=X^4
40 IF F<=1.E-32 THEN FNF=0 ELSE        Abfrage auf die Fehlerschranke und
   FNF=F                               entsprechende Zuweisung eines Funktionswerts.
50 END DEF
60 DISP FNROOT(2,3,FNF(FVAR))          Berechnet die Nullstelle und zeigt sie an.
70 DISP FVALUE                         Anzeige des Funktionswerts an der berechneten
                                       Nullstelle.
```

Eingabe/Ergebnis

RUN

```
8.30442502653E-9
0
```

Ohne Verwendung dieser *Fehlerschranken-Technik* hätte die Berechnung der Nullstelle wesentlich mehr Zeit in Anspruch genommen. Die Gründe hierfür liegen darin, daß bei der Berechnung der Funktion erst dann ein Bereichsunterlauf auftreten würde, wenn x sehr nahe an Null ist (da die Nullstelle im Punkt 0 liegt), und daß die maschinendarstellbaren Zahlen sich in der Nähe von Null häufen. Daher würde FNROOT eine Vielzahl von Näherungen benötigen, bis eine akzeptable Näherung der Nullstelle erreicht ist.

Eine weitere Technik zur Verringerung der Ausführungszeiten besteht darin, die Funktion so zu transformieren, daß die Nullstelle nicht mehr im Punkt 0 liegt, die Nullstelle dann zu berechnen und danach zurückzutranformieren. Dieses Verfahren führt für gewisse Funktionen mit Nullstellen in der Nähe von Null zu verringerten Ausführungszeiten, hat jedoch den Nachteil, daß ein Genauigkeitsverlust in Kauf genommen werden muß. Nachstehend finden Sie ein Beispielprogramm für $f(x) = x^4$.

```
10 STD @ DESTROY ALL
20 DEF FNF(X)=(X-1)^4          Transformation von x^4 um 1.
30 R=FNROOT(3,4,FNF(FVAR))     Berechnung der Nullstelle.
40 DISP R-1                    Rücktransformation der Nullstelle und Anzeige
                               von Nullstelle und Funktionswert.
50 DISP FVALUE
```

Schließlich sei noch eine Technik erwähnt, die sowohl die Ausführungszeit als auch die Genauigkeit von FNROOT verbessern kann. Jede Gleichung gehört zu einer unendlich großen Familie äquivalenter Gleichungen, die alle die gleichen Lösungen besitzen. Unter Umständen sind jedoch einige dieser Gleichungen einfacher zu lösen als andere. Beispielsweise haben die beiden Gleichungen $f(x) = 0$ und $\exp(f(x)) - 1 = 0$ die gleichen reellen Nullstellen; jedoch wird eine dieser Gleichungen fast immer einfacher zu lösen sein als die andere. Für $f(x) = x^4 - 6x - 1$ ist die erste Gleichung einfacher; für $f(x) = \ln(x^4 - 6x - 1)$ jedoch die zweite. Obwohl FNROOT für eine Vielzahl von Problemstellungen exakte Ergebnisse liefert, kann es jedoch unter Umständen ratsam erscheinen, die hier vorgestellten Möglichkeiten zu berücksichtigen.

Anhalten von FNROOT mit ATTN

Wenn keines der Argumente von FNROOT einen Aufruf einer mehrzeiligen benutzerdefinierten Funktion enthält, kann die Operation von FNROOT bis zum Speichern von Zwischenergebnissen nicht mit ATTN abgebrochen werden. Die Operation von FNROOT läuft im einzelnen wie folgt ab: FNROOT gibt den momentanen Wert von FVAR als Wert der angeblichen Wurzel zurück und speichert diesen Wert zugleich ab. Der zuletzt angenommene Wert der Wurzel wird als FGUESS und der Wert von $f(x)$ an dem momentanen Wert von FVAR als FVALUE gespeichert. Erst nach Abschluß dieser Operationen hält die Ausführung von FNROOT an.

Wenn dagegen FNROOT eine oder mehrere mehrzeilige benutzerdefinierte Funktionen als Argumente enthält (d.h., wenn die Berechnung von FNROOT die Ausführung mehrerer BASIC-Programmzeilen umfaßt), wird [ATTN] solange ignoriert, bis eine dieser benutzerdefinierten Funktionen aufgerufen wird. Die Ausführung hält dann an einer Zeile der benutzerdefinierten Funktion an. Dadurch sind Sie in der Lage, wichtige Werte wie beispielsweise den momentanen Wert von FVAR zu untersuchen und anschließend die Ausführung von FNROOT fortzusetzen (falls gewünscht).

Ein weiterer Vorteil in der Verwendung von mehrzeiligen benutzerdefinierten Funktionen als Argument(en) von FNROOT bestehtdarin, daß die Umgebung von FNROOT bei Auftreten eines Fehlers in der benutzerdefinierten Funktion nicht zerstört wird. Damit stehen Ihnen die Korrektur- und Fortsetzungsmöglichkeiten des HP-71 vollständig zur Verfügung.

CALC-Modus

FNROOT kann im CALC-Modus weder direkt noch indirekt aufgerufen werden. Wenn Ihr momentaner File beispielsweise eine einzeilige benutzerdefinierte Funktion FNF enthält, deren Definition das Schlüsselwort FNROOT enthält, führt der Versuch, FNF im CALC-Modus aufzurufen, zu einer Fehlerbedingung.

Schachtelungsregeln

Wenn das dritte Argument *F* von FNROOT eine Formel definiert, deren Auswertung einen weiteren Aufruf von FNROOT impliziert, spricht man von einer FNROOT Schachtelung. Die Schachtelungstiefe von FNROOT Schachtelungen ist auf 5 Ebenen beschränkt.

Als Beispiel für die Schachtelung von FNROOT soll das nachstehende Programm betrachtet werden, das die Funktion $f(x,y) = x^2 + y^2 - 2x - 2y + 2$ nach x und y auflöst.

```
10 STD @ DESTROY ALL
20 DEF FNF(X,Y)=X^2+Y^2-2*X-2*Y+2
30 DEF FNG(X)
40 R=FNROOT(-4,4,FNF(X,FVAR))
50 FNG=FVALUE

60 END DEF
70 DISP FNROOT(-3,3,FNG(FVAR));R
```

Definiert die obige Funktion.

Die Zeilen 30 bis 60 definieren eine Funktion $g(x)$ mit nur noch einer Variablen, die einen festen Wert für x (nämlich FVAR) von Zeile 70 bezieht.

Wenn von Zeile 50 ein Ergebnis ungleich 0 zurückgegeben wird, dann wird ein weiterer x-Wert für FNROOT in Zeile 40 ausgewählt. Eine Nullstelle von $f(x,y)$ ist gefunden, wenn von Zeile 50 der Wert 0 zurückgegeben wird.

Eingabe/Ergebnis

RUN

```
 1   .999999999999
```

Die von FNROOT in Zeile 70 zurückgegebenen x- und y-Werte. Der x-Wert wird links angezeigt.

.999999999999 ist die beste Näherung, die FNROOT für den wahren y-Wert 1 finden kann, da bereits dieser y-Wert (zusammen mit dem x-Wert 1) den Funktionswert 0 liefert.

FVALUE END LINE

```
 0
```

Diese x- und y-Werte liefern bei Einsetzen in $f(x,y)$ das Ergebnis 0.

Eine weit verbreitete Verwendung von FNROOT ist die Bestimmung lokaler Minima. Zur Erläuterung dieser Anwendung soll die obige Funktion durch Addition von 1 modifiziert werden. Dadurch wird sichergestellt, daß die Funktion keine Nullstelle hat, d.h. der durch die modifizierte Funktion repräsentierte Paraboloid schneidet die xy-Ebene nicht mehr. Das Programm wird nur in Zeile 20 modifiziert:

```
20 DEF FNF(X,Y)=X^2+Y^2-2*X-2*Y+3
```

Alle weiteren Programmzeilen bleiben unverändert.

Das zuvor verwendete geschachtelte FNROOT-Programm benötigte etwa 20 Sekunden zum Ermitteln einer Lösung. Da FNROOT mit besonderer Sorgfalt sicherstellt, daß ein echtes Minimum berechnet wird, benötigt das modifizierte Programm etwa 3½ Minuten, um den x- und y-Wert des Funktionsminimums zu finden.

Eingabe/Ergebnis

RUN

```
 1.00000191832   1.0000
014444
```

Der x- und y-Wert des berechneten Funktionsminimums.

FVALUE END LINE

Zeigt den Wert der modifizierten Funktion für den gegebenenen x- und y-Wert an.

```
 1
```

Sie müssen jedoch nicht die vollen 3½ Minuten abwarten, bis das Ergebnis angezeigt wird. Sie können, wie bereits auf Seite **97** beschrieben, die Ausführung von FNROOT anhalten und anschließend Zwischenergebnisse anzeigen. Wenn sich bei aufeinanderfolgenden Untersuchungen dieser Werte keine wesentlichen Veränderungen, können Sie die Zwischenergebnisse als genügend genau akzeptieren.

Verwendung von benutzerdefinierten Funktionen

Die Funktion FNROOT kann nicht über das Tastenfeld ausgeführt werden, wenn das dritte Argument von FNROOT eine benutzerdefinierte Funktion auswertet. In diesem Fall muß FNROOT als Programmanweisung ausgeführt werden. Ebenso können Sie keine benutzerdefnierte Funktion, weder im BASIC- noch im CALC-Modus, über das Tastenfeld ausführen, wenn die Ausführung von FNROOT angehalten wurde.

Numerische Integration

Schlüsselworte

Mit Hilfe der in diesem Abschnitt beschriebenen Schlüsselworte können Sie das Intergral einer Funktion von maximal 5 Variablen mit einer von Ihnen gewählten Genauigkeit berechnen.

Der größte Teil dieses Abschnitts behandelt die Anwendung dieser Schlüsselworte auf Funktionen einer Variablen. Funktionen mehrerer Variablen werden unter *Schachtelungsregeln und Volumenintegration* auf Seite 109/110 beschrieben.

Das Schlüsselwort INTEGRAL kann über das Tastenfeld oder in einem Programm zur Berechnung des Integrals einer Funktion verwendet werden, vorausgesetzt, daß die Funktion zusammen mit dem Schlüsselwort eingegeben oder in dem Programm definiert wird.

Die Schlüsselworte IBOUND und IVALUE geben zusätzliche Information zurück, die die Interpretation des Integralwerts vereinfachen. INTEGRAL, IBOUND und IVALUE geben jeweils einzelne numerische Werte zurück, so daß Sie diese Schlüsselworte zusammen mit anderen numerischen Funktionen und Variablen in numerischen Ausdrücken verwenden können. Ein viertes Schlüsselwort, IVAR, repräsentiert die Integrationsvariable (oder eine der Integrationsvariablen) der mit INTEGRAL zu integrierenden Funktion. Ebenso gibt IVAR die letzte von INTEGRAL benutzte Stützstelle zurück.

INTEGRAL

Bestimmtes Integral

INTEGRAL(*A*,*B*,*E*,*F*)

wo *A*, *B*, *E* und *F* reellwertige numerische Ausdrücke sind.

Gibt eine Näherung für das Integral von *A* bis *B* der Funktion *F* zurück. Der relative Fehler *E* (1E−12≤*E*≤1) deutet die Genauigkeit von *F* an und wird zur Berechnung einer Fehlertoleranz in der Näherung für das Integral verwendet.

Das berechnete Wert für das Integral kann sein:

- Eine Näherung für das Integral mit einer durch den relativen Fehler *E* vorgegeben Genauigkeit.
- Die letzte von 16 Näherungen für das Integral, bei denen der Integrand an 65535 Stützpunkten ausgewertet wurde, ohne daß dabei das Konvergenzkriterium erfüllt wurde.
- Die beim Drücken von [ATTN] zurückgegebene momentan beste Näherung für das Integral, wenn *F* keine mehrzeilige benutzerdefinierte Funktion aufruft.

INTEGRAL (Fortsetzung)

INTEGRAL erzeugt eine Folge von immer genaueren Näherungen für das gesuchte bestimmte Integral. Wenn drei aufeinanderfolgende Näherungen jeweils innerhalb der Fehlertoleranz voneinander liegen (d.h. die erste liegt dicht an der zweiten und die zweite liegt dicht an der dritten), wird die Folge abgebrochen, und die dritte Näherung wird als der gesuchte Integralwert zurückgegeben. Wenn dieses Konvergenzkriterium auch nach 16 Folgengliedern nicht erfüllt ist, wird der Wert der 16. Näherung zurückgegeben.

Kann nicht im CALC-Modus verwendet werden. Weitere Informationen über INTEGRAL und den CALC-Modus werden auf Seite 111 gegeben.

Auf den Seiten 109-111 finden Sie Informationen über die Schachtelung von INTEGRAL (Volumenintegration) und über die Wechselwirkungen zwischen INTEGRAL und [ATTN] und zwischen INTEGRAL und benutzerdefinierten Funktionen.

IVAR

Integrationsvariable

IVAR

Repräsentiert die Integrationsvariable in der die Funktion *F* definierenden Formel. Ist das letzte Argument von INTEGRAL.

Gibt die zuletzt von INTEGRAL benutzte Stützstelle zurück.

Kann im CALC-Modus verwendet werden.

IVALUE

Letztes Ergebnis von INTEGRAL

IVALUE

Gibt die letzte von INTEGRAL berechnete Näherung zurück. Wenn die Ausführung von INTEGRAL durch Drücken von [ATTN] oder auf eine andere Weise unterbrochen wurde, gibt IVALUE den Wert der momentanen Näherung zurück. Ansonsten gibt IVALUE den gleichen Wert zurück, der auch bei der letzten Ausführung von INTEGRAL zurückgegeben wurde.

IVALUE behält auch nach einem Ausschalten des HP-71 solange den momentanen Wert, bis INTEGRAL erneut ausgeführt wird.

Kann im CALC-Modus verwendet werden.

IBOUND

Fehlerabschätzung für INTEGRAL

IBOUND

Gibt eine Abschätzung des *absoluten* Fehlers für das zuletzt mit INTEGRAL berechnete Integral zurück.

- Ein positiver IBOUND-Wert deutet an, daß die Folge der Näherungen konvergiert hat.
- Ein negativer IBOUND-Wert deutet an, daß die Folge der Näherungen das Konvergenzkriterium nicht vollständig erfüllt hat; der von INTEGRAL zurückgegebene Wert kann außerhalb der Fehlertoleranz um den wahren Integralwert liegen.

Wie IVALUE erhält auch IBOUND (selbst nach einem Ausschalten des HP-71) solange den momentanen Wert, bis INTEGRAL erneut ausgeführt wird. Im Gegensatz zu IVALUE steht der Wert von IBOUND in keinerlei Beziehung zur momentanen Näherung, wenn die Ausführung von INTEGRAL unterbrochen wird.

Kann im CALC-Modus verwendet werden.

Gehen Sie zur Berechnung eines bestimmten Integrals wie folgt vor:

1. Schreiben Sie einen Ausdruck, der die zu integrierende Funktion repräsentiert.
2. Ersetzen Sie die Integrationsvariable des Ausdrucks durch das Schlüsselwort IVAR.
3. Verwenden Sie diesen Ausdruck als viertes Argument F von INTEGRAL.
4. Verwenden Sie die Integrationsuntergrenze als erstes Argument (A) und die Obergrenze als zweites Argument (B) von INTEGRAL.
5. Wählen Sie als drittes Argument E von INTEGRAL einen Wert, der ein Maß für den *relativen* Fehler in der Berechnung des Integranden darstellt. Jeder für E gewählte Wert wird auf den Bereich [1E−12,1] gerundet und sollte die Ungleichung

$$\frac{|\text{WAHRER INTEGRAND} - \text{BERECHNETER INTEGRAND}|}{|\text{BERECHNETER INTEGRAND}|} \leqslant E.$$

erfüllen. Da INTEGRAL den wahren Wert des zu bestimmenden Integrals nicht kennt, müssen Sie dieses Maß für den relativen Fehler angeben. Bei vielen rein mathematischen Funktionen (SIN, EXP, Polynome usw.) und endlichen Integrationsgrenzen kann die volle 12-stellige Genauigkeit des HP-71 ausgenutzt werden, so daß ein Wert für E in der Größenordnung von 1E−12 sinnvoll ist.

Zwischen INTEGRAL und IBOUND besteht der folgende Zusammenhang:

1. Basierend auf dem relativen Fehler E für die spezifizierte Funktion berechnet der Computer eine Fehlertoleranz für das Integral der Funktion. Wenn $f(X)$ die durch F approximierte wahre Funktion darstellt, sollte E so gewählt werden, daß

$$\frac{|F(X) - f(X)|}{|F(X)|} \leqslant E$$

für alle X im Integrationsintervall erfüllt ist. Der für E spezifizierte Wert wird gerundet, so daß gilt: $1E{-}12 \leqslant E \leqslant 1$.

Wenn F beispielsweise aus Meßdaten mit N signifikanten Stellen abgeleitet wird, sollte E auf 10^{-N} gesetzt werden.

2. Der Computer berechnet eine Folge von Approximationen I_k für das Integral der spezifizierten Funktion. Die Differenz zwischen zwei aufeinanderfolgenden Näherungen wird dann jeweils mit der Fehlertoleranz für das Integral verglichen.

3. Ein Wert für das Integral wird zurückgegeben, wenn gilt:

   - Die Folge der Approximationen I_k hat konvergiert. Zur Bestimmung der Konvergenz wird eine Folge von Näherungen J_k für das Integral der Funktion $E*|F|$ über dem gleichen Integrationsinterval benutzt. J_k beschreibt den bei der Berechnung von I_k auftretenden Fehler.

     INTEGRAL nimmt an, daß die Folge der Näherungen I_k gegen I_n konvergiert hat, wenn

$$|I_k - I_{k-1}| \leqslant J_k$$

     für $k = n - 1$ und $k = n$ gilt. INTEGRAL gibt dann den Wert von I_n zurück; IBOUND gibt einen positiven Wert als Fehlerabschätzung zurück.

   - Das Konvergenzkriterium ist auch nach der Berechnung der Näherungen I_1 bis I_{16} nicht erfüllt. INTEGRAL gibt dann I_{16} zurück; IBOUND liefert einen negativen Wert als Fehlerabschätzung.

Beipiele

Integration von $f(x) = x^2-2$ (INTEGRAL, IVAR)

Die folgenden sechs Beipiele verdeutlichen verschiedene Verwendungsmöglichkeiten von INTEGRAL und IVAR zur Integration der Funktion $x^2 - 2$ von 1 bis 2. Das Integrationsergebnis wird im ersten und sechsten Beispiel angegeben.

Beispiel 1:

Eingabe/Ergebnis

```
INTEGRAL(1,2,1E-11,IVAR^2-2)
END LINE
```

INTEGRAL kann sowohl über das Tastenfeld eingegeben als auch in einem Programm verwendet werden.

```
.333333333331
```

Beispiel 2:

```
10 DISP INTEGRAL(COS(0),LOG2(4),
   1E-11,IVAR^2-2)
```

Als Integrationsgrenzen können auch arithmetische Ausdrücke verwendet werden.

Beispiel 3:

```
10 DEF FNG=IVAR^2-2
20 DISP INTEGRAL(1,2,1E-11,FNG)
```

Das vierte Argument von INTEGRAL kann ein Ausdruck oder der Aufruf einer benutzerdefinierten Funktion sein.

Beispiel 4:

```
10 DEF FNF(X)=X^2-2
20 DISP INTEGRAL(1,2,1E-11,FNF(IVAR))
```

IVAR kann wie oben in einer benutzerdefinierten Funktion oder im vierten Argument von INTEGRAL verwendet werden.

Beispiel 5:

```
10 DEF FNH

20 FNH=IVAR^2-2
30 END DEF
40 DISP INTEGRAL(1,2,1E-11,FNH)
```

Die benutzerdefinierte Funktion kann ein- oder mehrzeilig sein.

Beispiel 6:

```
10 DEF FNJ(X)
20 FNJ=X^2-2
30 END DEF
40 DEF FNF(X)=2*X
50 DISP INTEGRAL(1,FNF(1),1E-11,
     FNJ(IVAR))

60 DISP IBOUND
```

Das erste, zweite und dritte Argument von INTEGRAL kann ebenfalls einen Aufruf einer benutzerdefinierten Funktion enthalten.

Eingabe/Ergebnis

RUN

```
 .333333333331
```

Das Integrationsergebnis.

```
 7.70341735781E-12
```

Eine Abschätzung für den absoluten Fehler des Integrationsergebnisses. Die Konvergenz der Integralnäherungen wird durch den positiven Wert bestätigt.

Integration von $f(x) = e^x - 2$ (INTEGRAL, IVAR, IVALUE)

Dieses Beispiel erläutert die Verwendung von IVALUE. Diese Funktion gibt die letzte von INTEGRAL berechnete Näherung zurück und wird selbst während der Ausführung von INTERGAL fortwährend aktualisiert. IVALUE gibt nach Abschluß der Integration den gleichen Wert wie INTEGRAL zurück.

Durch Anzeige von IVALUE können Sie während der Ausführung von INTEGRAL die Verbesserung der Integrationsnäherung beobachten. Diese Eigenschaft wird in dem nachstehenden Programm, das die Funktion e^x-2 von 1 bis 3 integriert, verdeutlicht. Als Fehlertoleranz wird 1E−12 vorgegeben.

```
10 Y=IVALUE
```

Y = Wert von IVALUE bei Beginn der Programmausführung. (Setzt voraus, daß INTEGRAL mindestens einmal ausgeführt wurde.)

```
20 DEF FNF(X)
30 IF IVALUE=Y THEN 50
```

Zeigt IVALUE nur bei einer Wertveränderung an.

```
40 DISP IVALUE @ Y=IVALUE
50 FNF=EXP(X)-2
60 END DEF
70 DISP INTEGRAL(1,3,.000000000001,
   FNF(IVAR))
```

Eingabe/Ergebnis

RUN

```
 10.7781121979
 13.683897213
 13.3653590516
 13.3671560314
 13.3672555263
 13.3672550945
 13.3672550947
 13.3672550947
```

Erster angezeigter Wert von IVALUE.

Letzter angezeigter Wert von IVALUE.
Wert des Integrals (INTEGRAL).

Integration von $f(x) = \exp(x^3+4x^2+x+1)$ (INTEGRAL, IVAR, IBOUND, IVALUE)

Zur Berechnung des Integrals von 0 bis 1 der Funktion

$$f(x) = \exp(x^3 + 4x^2 + x + 1)$$

können Sie das folgende Programm verwenden.

```
10 DEF F(X)=EXP(X^3+4*X^2+X+1)
20 INPUT E
30 DISP 'INTEGRIEREND'
40 X=INTEGRAL(0,1,E,F(IVAR))
50 BEEP
60 DISP 'INTEGRALWERT: '
70 DISP X
80 DISP 'GESCHAETZTER FEHLER: '
90 DISP IBOUND
```

Die benutzerdefinierte Funktion F.

Fragt nach dem relativen Fehler in F im Vergleich zu $f(x)$.

Tasten Sie das Programm ein, und starten Sie es durch Drücken von [RUN].

Eingabe/Ergebnis

[RUN]

```
? 
```

Eingabeaufforderung für den relativen Fehler der Funktion.

1E-5 [END LINE]

Obwohl der HP-71 die Funktion F bis auf eine Abweichung von 1 in der zwölften Stelle exakt berechnet, soll eine geringere Genauigkeit (hier eine Abweichung von 1 in der fünften Stelle) vorgegeben werden, um die Ausführungszeit zu verkürzen.

```
INTEGRIEREND
```

```
INTEGRALWERT:
 104.291097226
GESCHAETZTER FEHLER:
 1.04263904392E-3
```

Der Wert des Integrals ist 104.2911 $\pm$ 1.04×10^{-3}.

IVALUE [END LINE]

```
104.291097226
```

IVALUE gibt den Wert des zuletzt berechneten Integrals zurück.

Integration von $C(T) = a + bT$ (INTEGRAL, IVAR, IBOUND)

Sie können mittels INTEGRAL die Wärmemenge berechnen, die benötigt wird, um ein Gramm eines Gases bei konstantem Volumen von einer gegebenen Temperatur auf eine andere Temperatur zu erwärmen. Die benötigte Wärmemenge Q ergibt sich aus der Formel

$$Q = \int_{T1}^{T2} C(T)\, dT$$

wo $C(T)$ die spezifische Wärme des Gases als Funktion der Temperatur, $T1$ die Anfangs- und $T2$ die Endtemperatur ist.

In dem hier betrachteten Beipiel sei $C(T) = a + bT$, wo a und b experimentell zu $a = 1.023\text{E}^{-2}$ und $b = 2.384\text{E}^{-2}$ mit je 4 signifikanten Ziffern bestimmt sind. Der relative Fehler in $C(T)$ ergibt sich dann näherungsweise zu 5E−4. Das folgende Programm verlangt die Eingabe der Anfangs- und Endtemperatur in Kelvin und berechnet dann die zum Erhöhen der Gastemperatur von der Anfangstemperatur auf die Endtemperatur benötigte Wärmemenge.

```
10 DEF FNC(T)=.01023+.02384*T
20 INPUT 'ANFANGST., ENDT. (K)?';T1,T2
30 DISP 'INTEGRIEREND'
40 Q=INTEGRAL(T1,T2,.0005,FNC(IVAR))
50 DISP 'BENOETIGTE WAERMEMENGE:'
60 DISP Q;'+-';IBOUND
```

Benutzerdefinierte Funktion zur Berechnung der spezifischen Wärme.

Berechnung des Integrals.

Anzeige des Resultats und der Fehlerabschätzung.

Tasten Sie das Programm ein, und berechnen Sie die Wärmemenge, die benötigt wird, um das Gas von 300 K auf 310 K zu erhitzen.

Eingabe/Ergebnis

RUN

```
ANFANGST., ENDT. (K)?
```

300,310 END LINE

```
INTEGRIEREND
BENOETIGTE WAERMEMENGE:
 72.8143 +- .03640715
```

Weitere Informationen

Schachtelungsregeln und Volumenintegration

Wenn das vierte INTEGRAL-Argument *F* eine Formel spezifiziert, die eine weitere Auswertung von INTEGRAL bedingt, ist eine INTEGRAL Schachtelung gegeben. Bei INTEGRAL Schachtelungen sind maximal 5 Schachtelungsebenen zulässig. Programme mit zweifachen INTEGRAL Schachtelungen können beispielsweise zur Berechnung von Volumina verwendet werden.

Das nachstehende zur Integration von $f(x, y) = x^2 + 2y$ über dem Quadrat $0 < x < 1$, $0 < y < 1$ verwendete Programm verdeutlicht die Schachtelung von INTEGRAL. Das Programm berechnet das Doppelintegral

$$\int_0^1 \int_0^1 f(x, y) dy dx$$

```
10 DEF FNF(X,Y)=X^2+2*Y
20 DEF FNG(X)=INTEGRAL(0,1,1E-6,
   FNF(X,IVAR))
30 DISP INTEGRAL(0,1,1E-6,FNG(IVAR))
```

Definiert die zu integrierende Funktion.

Für jeden Wert von x wird über einen Streifen parallel zur y-Achse integriert.

Summiert alle Streifen parallel zur y-Achse auf.

Eingabe/Ergebnis

RUN

```
 1.33333333333
```

Das von INTEGRAL in Zeile 30 zurückgegebene Volumen.

IBOUND END LINE

```
 1.33317012712E-6
```

Obwohl das berechnete Ergebnis exakt ist, deutet IBOUND nur eine Genauigkeit von sechs Stellen an.

Das nachstehende Beispiel erläutert die Verwendung von INTEGRAL zur Auswertung des Integrals

$$\int_0^{\pi/2} \int_0^{y} \sin(x)\, dx\, dy$$

Eingabe/Ergebnis

```
RADIANS END LINE
INTEGRAL(0,PI/2,1E-3,
INTEGRAL(0,IVAR,1E-
3,SIN(IVAR)))
END LINE
```

Die erste IVAR repräsentiert die Intergrationsvariable des äußeren Integrals.

```
 .57080016668
```

IBOUND END LINE

```
 5.69950328155E-4
```

Das korrekte Ergebnis ist $\pi/2 - 1$ (ca. 0.5707963268).

Abbrechen von INTEGRAL mit [ATTN]

Wenn keines der Argumente von INTEGRAL einen Aufruf einer mehrzeiligen benutzerdefinierten Funktion enthält, kann die Operation von INTEGRAL bis zum Speichern von Zwischenergebnissen nicht mit [ATTN] abgebrochen werden. INTEGRAL arbeitet im einzelnen wie folgt: INTEGRAL gibt den momentanen Wert von IVALUE als angeblichen Wert des Integrals zurück und speichert diesen zugleich ab. Außerdem bewirkt INTEGRAL eine Vorzeichenumkehr bei dem Wert von IBOUND. Die Ausführung von INTEGRAL hält erst nach Abschluß dieser Operationen an.

Wenn INTEGRAL dagegen eine oder mehrere mehrzeilige benutzerdefinierte Funktionen als Argumente enthält (d.h., wenn die Berechnung von INTEGRAL die Ausführung mehrerer BASIC-Programmzeilen umfaßt), wird [ATTN] solange ignoriert, bis eine dieser benutzerdefinierten Funktionen aufgerufen wird. Die Ausführung hält dann an einer Zeile der benutzerdefinierten Funktion an. Dadurch sind Sie in der Lage, wichtige Werte wie den momentanen Wert von IVALUE zu untersuchen und anschließend (falls gewünscht) die Ausführung fortzusetzen.

Ein weiterer Vorteil in der Verwendung von mehrzeiligen benutzerdefinierten Funktionen als INTEGRAL Argumente besteht darin, daß die Umgebung von INTEGRAL bei Auftreten eines Fehlers in der benutzerdefinierten Funktion nicht zerstört wird. Damit stehen Ihnen die Korrektur- und Fortsetzungsmöglichkeiten des HP-71 vollständig zur Verfügung.

CALC-Modus

INTEGRAL kann im CALC-Modus weder direkt noch indirekt aufgerufen werden. Wenn Ihr momentaner File beispielsweise eine einzeilige benutzerdefinierte Funktion FNF enthält, deren Definition das Schlüsselwort INTEGRAL enthält, führt der Versuch, FNF im CALC-Modus aufzurufen, zu einer Fehlerbedingung.

Verwendung von benutzerdefinierten Funktionen

Die Funktion INTEGRAL kann nicht über das Tastenfeld ausgeführt werden, wenn das vierte Argument der INTEGRAL Funktion eine beliebige benutzerdefinierte Funktion auswertet. In diesem Fall muß INTEGRAL als Programmanweisung ausgeführt werden. Ebenso können benutzerdefinierte Funktionen, weder im BASIC- noch im CALC-Modus, über das Tastenfeld ausgeführt werden, wenn die Ausführung von INTEGRAL angehalten wurde.

Allgemeines zur numerischen Integration

Alle numerischen Integrationsverfahren beinhalten die Auswertung der zu integrierenden Funktion an einer Reihe von Stützstellen im Integrationsintervall. Das berechnete Integral ist dann einfach ein gewichtetes Mittel der Funktionswerte an diesen Stützstellen. Da ein bestimmtes Integral tatsächlich jedoch ein Mittel der Funktionswerte an *unendlich* vielen Stützstellen darstellt, kann ein numerisches Integrationsverfahren nur dann befriedigende Resultate liefern, wenn die gewählten Stützstellen das Verhalten der Funktion auf dem Gesamtintervall ausreichend gut beschreiben.

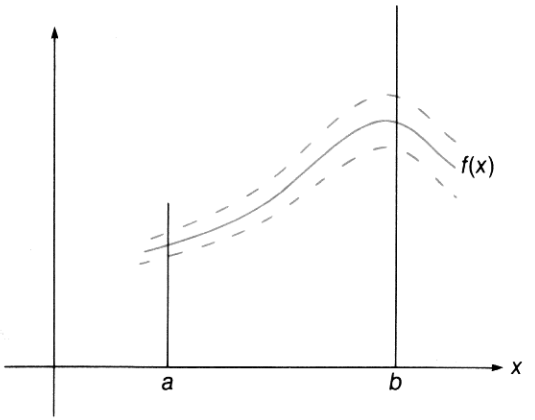
Wenn die Stützstellen dicht zusammen liegen, und die Funktion ihr Verhalten zwischen aufeinanderfolgenden Stützstellen nicht zu rasch ändert, liefert die numerische Integration in der Regel verläßliche Resultate. Andererseits liefert die numerische Integration häufig unzureichende Ergebnisse, wenn der Graph der Funktion auf einem Bereich stark variiert, der klein im Vergleich zum gesamten Integrationsgebiet ist. Andere Fehler, die das Ergebnis einer numerischen Integration beeinflussen können, sind die für jede Gleitkomma-Berechnung typischen Rundungsfehler und etwaige in der Routine zur Berechnung der zu integrierenden Funktion auftretende Fehler.

Behandlung numerischer Fehler

INTEGRAL benötigt die Vorgabe einer Fehlertoleranz E, um die Güte der zu integrierenden Funktion abschätzen zu können. Diese Fehlertoleranz sollte den *relativen* Fehler der benutzerdefinierten Funktion im Vergleich zu der "wahren", zu integrierenden Funktion beschreiben und wird dazu benutzt, um ein Band um die benutzerdefinierte Funktion zu legen, in dem die "wahre" Funktion liegen sollte. Wenn $f(x)$ die "wahre" Funktion und $F(x)$ die berechnete Funktion bezeichnet, sollte die Ungleichung

$$F(x) - \text{Fehler}(x) \leqslant f(x) \leqslant F(x) + \text{Fehler}(x)$$

für alle Punkte x im Integrationsintervall erfüllt sein. Fehler(x) ist hier die halbe Bandbreite an der Stelle x.



Es gilt dann

$$\int_a^b f(x)\,dx \approx \int_a^b F(x)\,dx \pm \int_a^b \text{Fehler}(x)\,dx$$

wobei das dritte Integral genau die Hälfte des Bandes beschreibt. Mit anderen Worten, die Integration der benutzerdefinierten Funktion anstelle der "wahren" Funktion kann zu keinem Fehler führen, der größer als die Hälfte der Fläche des Bandes um die benutzerdefinierte Funktion ist. INTEGRAL schätzt diesen Fehler während der Berechnung des Integrals ab; über IBOUND können Sie dann anschließend diesen Wert abrufen.

Auswahl der Fehlertoleranz

Die Genauigkeit der berechneten Funktion hängt von drei Faktoren ab:

- Güte der empirischen Konstanten der Funktion.
- Grad, mit dem das durch die Funktion dargestellte Modell die zugrundeliegende physikalische Situation beschreibt.
- Größe des bei der Berechnung anfallenden Rundungsfehlers.

Funktionen wie $\cos(x - \sin x)$ sind rein mathematische Funktionen; d.h. die Funktion selbst enthält keine empirischen Konstanten. Für derartige Funktionen können Sie beliebig kleine Fehlertoleranzen vorgeben, solange diese Funktion von der BASIC-Funktion trotz des unvermeidlichen Rundungsfehlers innerhalb dieser Toleranz berechnet wird. Da eine höhere Genauigkeit in der Regel durch längere Rechenzeiten erkauft wird, kann es unter Umständen sinnvoll sein, nicht die kleinstmögliche Fehlertoleranz zu wählen. Jede spezifizierte Fehlertoleranz wird auf einen Wert im Intervall [1E−12, 1] gerundet.

Wenn der Integrand eine physikalische Situation beschreibt, sind zusätzliche Überlegungen zu berücksichtigen. In jedem Fall sollten Sie abwägen, ob die für das berechnete Integral gewünschte Genauigkeit durch die Genauigkeit des Integranden gerechtfertigt ist. Wenn die Funktion beispielsweise empirische Konstanten enthält, die die tatsächlichen Konstanten nur auf drei Stellen annähern, ist es sinnlos Fehlertoleranzen kleiner als 1E−3 zu spezifizieren.

Des weiteren sollten Sie in Betracht ziehen, daß nahezu jede mit einer physikalischen Situation zusammenhängende Funktion bereits implizit einen Fehler beinhaltet, da die Funktion lediglich ein mathematisches Modell des eigentlichen Prozesses oder Ereignisses darstellt. Ein mathematisches Modell ist typischerweise eine Approximation, die die Auswirkungen aller nicht im Modell erfaßten Faktoren vernachlässigt.

Ein Beispiel für die unvollständige Erfassung eines physikalischen Prozesses durch ein mathematisches Modell, ist die Gleichung $s = s' - (.5)gt^2$, die die Höhe eines fallenden Körpers beschreibt, der aus einer ursprünglichen Höhe s' fallen gelassen wird. Hier wird die Abhängigkeit der Schwerkraftbeschleunigung g von der jeweiligen Höhe ignoriert. Mathematische Beschreibungen physikalischer Abläufe können lediglich Ergebnisse mit begrenzter Genauigkeit liefern. Wenn ein Integral mit einer Genauigkeit berechnet wird, die vom Modell nicht mehr unterstützt wird, ist es nicht gerechtfertigt, die (scheinbar) volle Genauigkeit des berechneten Werts zu benutzen. Die vorgegebene Fehlertoleranz sollte daher alle in der Funktion enthaltenen Ungenauigkeitsfaktoren berücksichtigen, da ansonsten mit einem hohen Rechenaufwand eine bedeutungslose Genauigkeit erkauft würde. Des weiteren ist die von `IBOUND` zurückgegebene Fehlerabschätzung möglicherweise nicht mehr signifikant.

Wenn eine Funktion $f(x)$ eine physikalischen Prozess beschreibt, ist der durch Rundung entstehende Fehler in der Regel sehr klein im Vergleich zu dem Modellfehler. Wenn $f(x)$ dagegen eine rein mathematische Funktion darstellt, ist ihre Genauigkeit lediglich durch den auftretenden Rundungsfehler beschränkt. Die exakte Bestimmung des bei der Berechnung einer derartigen Funktion entstehenden Fehlers ist im allgemeinen nur mit sehr komplizierten analytischen Methoden möglich. In der Praxis werden solche Effekte normalerweise durch Erfahrungswerte abgeschätzt.

Behandlung schwieriger Integrale

Integration auf Teilintervallen. Bei der Integration einer Funktion, die auf geringe Variationen im Argument mit substantiellen Schwankungen in den Funktionswerten reagiert, werden in der Regel wesentlich mehr Stützstellen als bei der Integration einer Funktion , die auf dem Integrationsintervall nur geringfügig variiert, benötigt. Der Grund hierfür liegt darin, daß das Verhalten der Funktion auf dem Gesamtintervall durch das Verhalten der Funktionswerte an den Stützstellen adäquat repräsentiert werden muß. Wenn eine Funktion in bestimmten Teilintervallen des Integrationsintervalls stärker variiert als in anderen, ist es sinnvoll, das Gesamtintervall zu unterteilen und die Funktion auf den Teilintervallen einzeln zu integrieren. Das Integral über das Gesamtintervall ist dann die Summe der Integrale über die Teilintervalle, und der Fehler des Integrals ist die Summe der Fehler der Integrale über die Teilintervalle.

Der von INTEGRAL verwendete Algorithmus entscheidet während der Ausführung, basierend auf dem Verhalten des Integranden auf einem bestimmten Intervall, wieviele Stützstellen benutzt werden sollen. Wenn nun das Integrationsintervall aufgeteilt wird, kann diese Stützstellenauswahl auf das Verhalten der Funktion auf dem betrachteten Teilintervall beschränkt werden. Dies führt im allgemeinen zu verbesserten Ausführungszeiten und einer erhöhten Genauigkeit.

Wenn Sie beispielsweise die Funktion $f(x) = (x^2 + 1E-12)^{1/2}$ von $x = -3$ bis $x = 5$ mit einer Fehlertoleranz von 1E−12 integrieren wollen, können Sie die dafür benötigte Rechenzeit wesentlich verkürzen, indem Sie das Intervall bei $x = 0$ unterteilen, wo die Funktion einen scharfen Knick hat. Da die Funktion über den beiden Teilintervallen $[-3, 0]$ und $[0, 5]$ sehr glatt verläuft, lassen sich die Integrale der Funktion über diesen Teilintervallen sehr einfach und schnell berechnen.

$$\int_{-3}^{5} f(x)\,dx = \int_{-3}^{0} f(x)\,dx + \int_{0}^{5} f(x)\,dx$$

Das folgende Programm berechnet die beiden Integrale über den Teilintervallen und kombiniert dann die Ergebnisse.

```
10 DEF FNF(X)=SQR(X*X+1E-12)
20 I=INTEGRAL(-3,0,1E-12,FNF(IVAR))
30 E=IBOUND
40 DISP 'WERT DES INTEGRALS:'
50 DISP I+INTEGRAL(0,5,1E-12,FNF(IVAR))
60 DISP 'GESCHAETZTER FEHLER:'
70 DISP E + IBOUND
```

Hier wird X*X anstelle X^2 verwendet, da X*X ein genaueres Resultat liefert. Diese Genauigkeitsbetrachtung gilt für jede *ganzzahlige* Potenz eines numerischen Ausdrucks.

Integration über das erste Teilintervall.

Zwischenspeicherung des Fehlers.

Summe des ersten und des zweiten Integrals.

Berechnung des Gesamtfehlers durch Summation der beiden Einzelfehler.

Tasten Sie das Programm ein, und starten Sie es durch Drücken von [RUN]. In der Anzeige erscheint dann das Ergebnis.

```
WERT DES INTEGRALS:
 17
GESCHAETZTER FEHLER:
 .000000000017
```

Bei Unterteilung des Intervalls werden hier zur Berechnung des Resultats nur wenige Sekunden benötigt; ohne diese Unterteilung kann die Ausführung des Programms eine beträchtliche Zeitspanne in Anspruch nehmen.

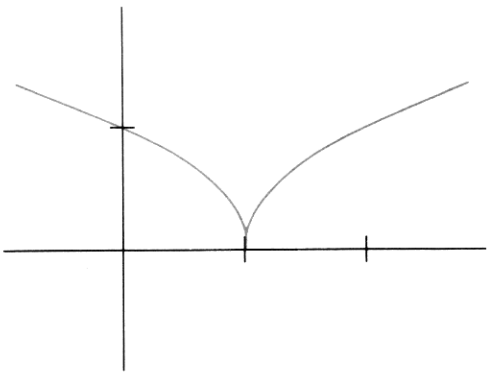
Die Unterteilung des Integrationsintervalls ist auch sinnvoll bei Funktionen mit Singularitäten im betrachteten Intervall. Die Singularität kann aus einem oder mehreren Punkten bestehen, an denen die Funktion nicht definiert ist oder einen Eckpunkt besitzt.

Beispielsweise sollte das Integral

$$\int_0^2 \frac{dx}{(x-1)^2} \qquad \text{in} \qquad \int_0^1 \frac{dx}{(x-1)^2} + \int_0^2 \frac{dx}{(x-1)^2}$$

zerlegt werden, um eine Auswertung der Funktion im Punkt $x = 1$ zu vermeiden, da die Funktion an dieser Stelle nicht definiert ist. Sie können die Funktion nun problemlos auf den beiden Teilintervallen integrieren, da $x = 1$ Endpunkt in jedem der beiden Teilintervalle ist und INTEGRAL keine Stützstellen in die Endpunkte des Integrationsintervalls legt.

Ähnlich problematisch ist normalerweise die Integration der Funktion $\sqrt{|x-1|}$, die bei $x = 1$ einen Eckpunkt besitzt.

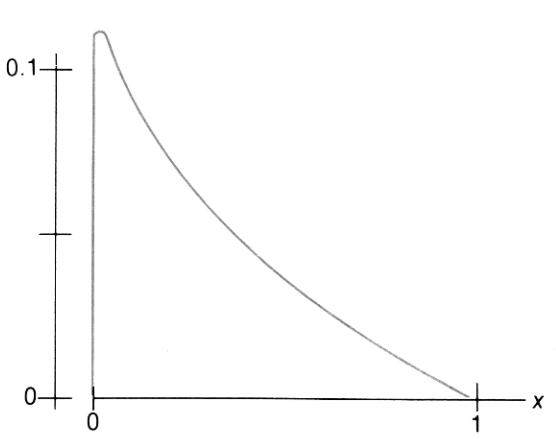


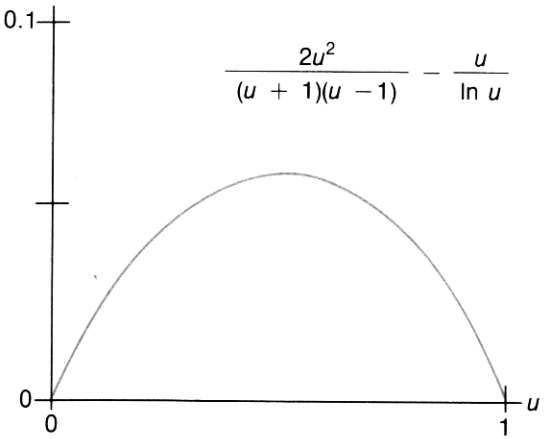
Wenn Sie diese Funktion von 0 bis 2 integrieren wollen, können Sie die Ausführungszeit verringern und die Genauigkeit erhöhen, indem Sie einzeln über die Teilintervalle [0, 1[und]1, 2] integrieren, auf denen die Funktion sehr glatt verläuft.

Variablentransformationen. Eine zweite Methode zur Behandlung schwieriger Integrale besteht in der Transformation der Variablen. Wenn die Variable eines bestimmten Integrals geeignet transformiert wird, ist das resultierende bestimmte Integral unter Umständen numerisch einfacher zu berechnen. Betrachten Sie zum Beispiel das Integral

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

Wie aus dem linken Graphen in der nächsten Abbildung zu ersehen ist, geht die Ableitung des Integranden gegen unendlich, wenn x gegen 0 strebt. Der Graph auf der rechten Seite zeigt, wie die Substitution $x = u^2$ ein wesentlich gutartigeres Verhalten bedingt.



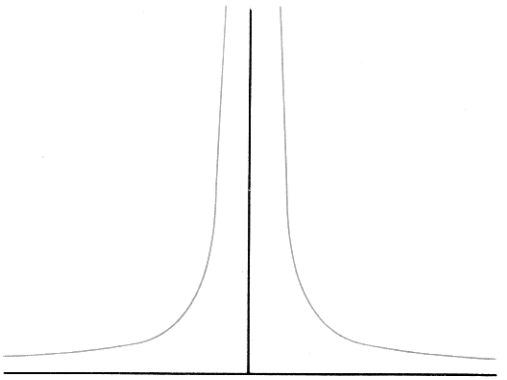


Sie können nun anstelle des ursprünglichen Integrals das aus dieser Substitution resultierende Integral

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

berechnen. (Ersetzen Sie nicht $(u + 1)(u - 1)$ durch $u^2 - 1$, da dieser Ausdruck für u gegen 1 die Hälfte seiner signifikanten Stellen durch Rundungsfehler verliert, was schließlich zu einem großen Endergebnis führt.)

Betrachten Sie die folgende Funktion als ein weiteres Beispiel für eine sinnvolle Anwendung der Substitution. Der Graph dieser Funktion besitzt einen langen Schweif, der sich weit aus dem Hauptkörper (der den Großteil der Fläche bildet) heraushebt.



Obwohl ein sehr dünner Schweif sich ohne große Genauigkeitsverluste abschneiden ließe, ist in diesem Fall der Schweif der Funktion zu breit, um bei der Berechnung von

$$\int_{-t}^{t} \frac{dx}{x^2 + 10^{-10}}$$

für große t ignoriert werden zu können. Im allgemeinen bildet die komprimierende Substitution $x = b \tan u$ die gesamte Zahlengerade in die entsprechenden Teilintervalle von $]-\pi/2, \pi/2[$ ab, wobei Teile der reellen Zahlengeraden in die entsprechenden Teilintervalle von $]-\pi/2, \pi/2[$ umgesetzt werden. Für $b = 1\text{E}{-5}$ erhält man die Substitution $x = 1\text{E}{-5} \tan u$, und das Integral wird zu

$\tan^{-1}(-t/b)$

$$10^5 \int_{\tan^{-1}(-t/b)}^{\tan^{-1}(t/b)} du$$

was sich problemlos für sehr große t berechnen läßt.

Die komprimierende Substitution ist auch das Standardverfahren zur Behandlung uneigentlicher Integrale. Beispielweise gilt:

$$\int_{-\infty}^{\infty} \frac{dx}{x^2 + 10^{-10}} = 10^5 \int_{-\pi/2}^{\pi/2} du$$

In einigen Fällen kann der Schweif einer Funktion auch ohne zu großen Genauigkeitsverlust abgeschnitten werden. Betrachten Sie dazu die Funktion $\exp(-x^2)$, die für $x > 34$ einen Bereichsunterlauf bedingt (d.h. in der Maschinenarithmetik das Ergebnis 0 erzeugt). Es gilt dann:

$$\int_0^{\infty} e^{-x^2} dx \approx \int_0^{34} e^{-x^2} dx$$

Daher können Sie bei der Behandlung uneigentlicher Integrale den Schweif abschneiden, wenn dieser nur unwesentlich zur Gesamtfläche beiträgt. In allen anderen Fällen sollte jedoch eine komprimierende Substitution verwendet werden.

Über den Algorithmus

Das Mathematik-Paket verwendet das Romberg-Verfahren zur Akkumulation eines Integralwerts. Die Effizienz des implementierten Verfahrens wird jedoch durch verschiedene Verfeinerungen gesteigert. Anstelle äquidistanter Stützstellen (d.h. Stützstellen, die jeweils den gleichen Abstand voneinander haben), die zu Resonanzen und damit bei periodischen Integranten zu verfälschten Ergebnissen führen können, benutzt INTEGRAL nicht äquidistante Stützstellen. Das verwendete Stützstellengitter wird am besten illustriert, indem man

$$x = \frac{3}{2}u - \frac{1}{2}u^3 \qquad \text{in} \qquad \int_a^b f(x)dx$$

substituiert und dann äquidistante Stützstellen für u verwendet. Neben der Resonanzunterdrückung hat diese Substitution noch zwei weitere Vorteile. Zunächst werden die Endpunkte des Integrationsintervalls nicht als Stützstellen verwendet; es sei denn, das Intervall ist so klein, daß ein Punkt im Intervallinnern durch Rundung zu einem Endpunkt wird. Dies führt dazu, daß die numerische Berechnung eines Integrals wie

$$\int_0^1 \frac{\sin x}{x}\, dx$$

nicht durch eine Division durch Null an einem Endpunkt unterbrochen wird. Des weiteren kann INTEGRAL auch zur Integration von Funktionen verwendet werden, deren Steigung in einem Endpunkt unendlich ist. Derartige Funktionen treten bei der Berechnung von Flächen auf, die von glatten geschlossenen Kurven wie $x^2 + f^2(x) = R$ begrenzt werden.

Zusätzlich verwendet INTEGRAL eine erhöhte Genauigkeit. Intern werden alle Summen mit 16 Stellen akkumuliert. Dies führt dazu, daß gegebenenfalls Tausende von Stützstellenwerten aufsummiert werden können, ohne daß durch Rundung mehr signifikante Stellen verloren gehen, als dies bereits in der Funktionsroutine geschieht.

Der durch INTEGRAL implementierte Iterationsprozess erzeugt eine Folge von Schätzwerten, die den "wahren" Wert des Integrals immer besser annähern. Des weiteren wird bei jeder Iteration auch die jeweilige Breite des Fehlerbandes abgeschätzt. Die Ausführung von INTEGRAL wird nur dann beendet, wenn drei aufeinanderfolgende Iterationen nur jeweils durch den berechneten Fehler voneinander verschieden sind, oder wenn dieses Abbruchkriterium auch nach 16 Iterationen nicht erfüllt ist.

In diesem Fall wurde die Funktion an 65535 Stellen ausgewertet. IBOUND gibt dann den berechneten Fehler mit einem negativen Vorzeichen zurück, um anzudeuten, daß der von INTEGRAL zurückgegebene Wert sich wahrscheinlich um mehr als die Fehlertoleranz vom tatsächlichen Wert des Integrals unterscheidet. In einer solchen Situation sollten Sie dann das Integrationsintervall in kleinere Teilintervalle zerlegen und die Funktion über jedem der Teilintervalle integrieren. Das Integral über dem ursprünglichen Intervall ist dann die Summe der Integrale über den Teilintervallen. Auf diese Weise kann die Funktion auf jedem Teilintervall an bis zu 65535 Stützstellen ausgewertet werden, was im allgemeinen zu einer höheren Genauigkeit bei der Berechnung des Integrals führt.

Zusammenfassend gilt, daß INTEGRAL für einen Vielzahl von Anwendungen auf rasche und bequeme Weise verläßliche Lösungen liefert. Die obigen theoretischen Betrachtungen beziehen sich auf allgemeine Probleme bei der numerischen Integration einer Funktion. Durch geeignete Anwendung der hier genannten Techniken lassen sich mittels INTEGRAL selbst schwierigste Integrale lösen.

Abschnitt 12

Bestimmen der Nullstellen eines Polynoms

Schlüsselwort

Das in diesem Abschnitt beschriebene Schlüsselwort PROOT erlaubt die Bestimmung *sämtlicher* Lösungen — sowohl der reellen als auch der komplexen — der Gleichung $P(x) = 0$, wo P ein gegebenes Polynom mit reellen Koeffizienten ist. Wenn P ein Polynom n-ten Grades ist, existieren n (nicht notwendigerweise verschiedene) Lösungen dieser Gleichung.

Um mittels PROOT die Lösungen der Gleichung $P(x) = 0$ zu bestimmen, wo

$$P(x) = a_n x^n + a_{n-1}x^{n-1} + \ldots + a_1 x + a_0$$

sind zunächst die Koeffizienten a_n, a_{n-1}, ..., a_0 in einem Feld mit insgesamt $n + 1$ Elementen zu speichern. Die Koeffizienten sind in dieser Reihenfolge abzulegen; d.h. der höchste Koeffizient zuerst und das konstante Glied zuletzt. Die Dimensionierung des Felds kann beliebig sein; das Mathematik-Paket verwendet lediglich die Gesamtanzahl der Elemente im Feld zur Bestimmung des Polynomgrads. Beispielsweise können die Felder

$$[6, 5, 4, 3, 2, 1], \begin{bmatrix} 6 & 5 & 4 \\ 3 & 2 & 1 \end{bmatrix}, \begin{bmatrix} 6 & 5 \\ 4 & 3 \\ 2 & 1 \end{bmatrix}, \text{ und } \begin{bmatrix} 6 \\ 5 \\ 4 \\ 3 \\ 2 \\ 1 \end{bmatrix}$$

alle zur Darstellung des Polynoms

$$6x^5 + 5x^4 + 4x^3 + 3x^2 + 2x + 1$$

verwendet werden. Das Feld, das die berechneten Lösungen aufnehmen soll, muß bei komplexen Nullstellen ebenfalls komplex sein. Wenn das Ergebnisfeld ein Vektor ist, wird der Vektor bei einem Polynom vom Grad N auf N Elemente umdimensioniert. Wenn das Ergebnisfeld eine Matrix ist, wird diese auf N Zeilen und eine Spalte umdimensioniert.

Der Grad des Polynoms, dessen Nullstellen berechnet werden sollen, ist lediglich durch die Größe des verfügbaren Speicherplatzes beschränkt.

PROOT

Nullstellen eines Polynoms

MAT **R**=PROOT(**P**)

wo **P** ein reeller Vektor oder eine reelle Matrix mit $N + 1$ Elementen ist, und wo N der Grad des Polynoms ist, dessen Nullstellen gesucht sind. **R** ist ein komplexer Vektor oder eine komplexe Matrix.

Dimensioniert **R** automatisch auf N Elemente um, wenn **R** ein Vektor ist. Dimensioniert **R** automatisch auf N Zeilen und eine Spalte um, wenn **R** eine Matrix ist. Weist **R** die (komplexen) Lösungen der Gleichung $P(x) = 0$ zu (wo P das Polynom n-ten Grades ist, dessen Koeffizienten in **P** abgelegt sind).

Die Operation kann durch zweimaliges Drücken von [ATTN] angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

Beispiel

Das folgende Beipiel findet alle Nullstellen des Polynoms

$$5Z^6 - 45Z^5 + 225Z^4 - 425Z^3 + 170Z^2 + 370Z - 500$$

```
OPTION BASE 1 [END LINE]
```

```
DIM A(7) [END LINE]
```

Erzeugt einen reellen Koeffizientenvektor.

```
COMPLEX B(10) [END LINE]
```

Erzeugt einen komplexen Vektor zur Aufnahme der Nullstellen.

```
MAT INPUT A [END LINE]
```

```
A(1)? ■
```

```
5,-45,225,-425,170,370,-500
[END LINE]
```

```
MAT B=PROOT(A) [END LINE]
```

Dimensioniert zuerst Vektor **B** auf sechs Elemente um, die erforderliche Mindestgröße zur Aufnahme der sechs (komplexen) Nullstellen des Polynoms 6-ten Grades. Berechnet anschließend alle Nullstellen und speichert diese in **B**.

MAT DISP B END LINE — Zeigt alle Nullstellen an.

```
(1,1)
(1,-1)
(-1,0)
(2,0)
(3,4)
(3,-4)
```

Weitere Informationen

Es gibt verschiedene Vorgehensweisen zur Überprüfung der Genauigkeit der berechneten Nullstellen. Eine Methode besteht darin, den Wert des Polynoms an einer vermeintlichen Nullstelle zu berechnen und diese Zahl dann mit Null zu vergleichen. Obwohl in der Theorie einfach und überzeugend, hat dieses Verfahren in der Praxis einige schwerwiegende Nachteile. Es kann sehr leicht vorkommen, daß die berechnete Nullstelle die beste maschineninterne Darstellung der wahren Nullstelle ist, aber da die Ableitung des Polynoms an dieser Stelle einen sehr großen Wert besitzt, ist der Wert des Polynoms an der berechneten Nullstelle ebenfalls sehr groß. Ein einfaches Beispiel für dieses Phänomens ist das Polynom $1E20x^2 - 2E20$. Die wahre Nullstelle ist in diesem Fall $\sqrt{2}$; die berechnete Nullstelle ist 1.41421356237, was der besten Maschinendarstellung von $\sqrt{2}$ entspricht. Der Wert des Polynoms für diese Näherung der Quadratwurzel von 2 ist jedoch $-1\,000\,000\,000$; ein *scheinbar* unakzeptabler Wert für eine Nullstelle.

Ein weiterer Nachteil dieses Verfahrens liegt darin, daß aufgrund der Genauigkeitsbeschränkungen in jeder numerischen Berechnung die Signifikanz in der Differenz zwischen dem Polynomwert und 0 durch Rundungsfehler bei der Auswertung des Polynoms vollständig eliminiert werden kann. Dies gilt insbesondere für Polynome sehr hohen Grades mit weit auseinanderliegenden Koeffizienten oder Nullstellen sehr hoher Ordnung.

Eine zweite Methode zur Abschätzung der Genauigkeit von berechneten Nullstellen besteht darin, daß man versucht, das Polynom aus den berechneten Nullstellen zu rekonstruieren. Wenn sich das rekonstruierte Polynom nicht zu stark vom ursprünglichen Polynom unterscheidet, werden die berechneten Nullstellen als ausreichend genau betrachtet. Diese Methode ist weniger sensitiv bezüglich den bei dem Polynomauswertungs-Verfahren beschriebenen Einflüssen. Andererseits liefert diese Methode natürlich keinerlei Aufschluß über die Genauigkeit einer einzelenen Nullstelle.

Das nachstehend gelistete Programm fordert zur Eingabe der Koeffizienten eines Polynoms auf und berechnet daraus die Nullstellen dieses Polynoms mit Hilfe des Schlüsselworts PROOT. Falls gewünscht, rekonstruiert das Programm die Koeffizienten anhand der berechneten Nullstellen. Zusätzlich ist das Programm in der Lage, den Wert des Polynoms an einer berechneten Nullstelle oder an einer beliebigen anderen reellen oder komplexen Stelle zu berechnen.

Die Zeilen 10 bis 200 bilden den Hauptteil des Programms zur Berechnung der Nullstellen eines gegebenen Polynoms mittels PROOT. Die Unterroutine in den Zeilen 210 bis 250 wertet das Polynom an einem beliebigen reellen oder komplexen Punkt über das Horner-Schema aus.

Die Unterroutine in den Zeilen 260 bis 410 rekonstruiert die Koeffizienten aus den berechneten Nullstellen. Dazu wird das Polynom (beginnend mit dem Polynom 1) mit den linearen Faktoren $(Z - R)$, wobei R eine berechnete reelle Nullstelle ist, oder mit dem quadratischen Ausdruck $Z^2 - 2 *$ REPT(R) + ABS$(R)^2$, wobei R eine berechnete komplexe Nullstelle ist, multipliziert. (Beachten Sie, daß auch CONJ(R) eine berechnete Nullstelle ist.)

```
10 OPTION BASE 0 @ INTEGER D,E @
   DIM U$[4] @ DELAY 1 @ WIDTH 96
```

```
20 INPUT "GRAD? ";D
```

D ist der Grad des Polynoms.

```
30 DIM P(D),C(D) @ COMPLEX R(D-1)
```

Die Matrix **P** wird die Koeffizienten des Polynoms in der zuvor angegebenen Reihenfolge enthalten, die Matrix **R** soll die berechneten Nullstellen aufnehmen und in der Matrix **C** werden die rekonstruierten Koeffizienten abgelegt.

```
40 DISP "KOEFFIZIENTENEINGABE" @
   MAT INPUT P
```

Eingabe der Koeffizienten. Der führende Koeffizient sollte ungleich 0 sein, um einen einwandfreien Programmablauf zu gewährleisten.

```
50 DISP "RECHNEND..."
```

```
60 MAT R=PROOT(P)
```

Berechnet die Nullstellen und speichert sie in **R**.

```
70 DISP "DIE NULLSTELLEN SIND:" @
   DELAY 8 @ MAT DISP R @ DELAY 1
```

Zeigt die gefundenen Nullstellen an. Drücken Sie zur Anzeige einer weiteren Nullstelle und zur Fortsetzung des Programms eine beliebige Taste.

```
80 U$=KEY$ @ INPUT
   "REKONSTRUKTION? (J/N)";U$
```

Falls gewünscht rekonstruiert das Programm die Koeffizienten aus den berechneten Nullstellen.

```
90 IF UPRC$(U$)="J" THEN GOSUB 260
   ELSE 110
```

Die in Zeile 260 beginnende Unterroutine führt die Rekonstruktion aus und speichert die rekonstruierten Koeffizienten in Matrix **C**.

```
100 DISP "DIE REKONSTRUIERTEN " @
    DISP "KOEFFIZIENTEN SIND: " @
    DELAY 8 @ MAT DISP C @ DELAY 1
```

Zeigt die rekonstruierten Koeffizienten an. Drücken Sie zur Anzeige einer weiteren Nullstelle und zur Fortsetzung des Programms eine beliebige Taste.

```
110 U$=KEY$ @ INPUT
    "AUSWERTUNG? (J/N) ";U$
```

Optionale Auswertung des Polynoms an einer Nullstelle oder einer beliebigen anderen Stelle.

```
120 IF UPRC$(U$)#"J" THEN 190
    ELSE COMPLEX Z
```

Der Wert des Polynoms wird in der komplexen Variable Z gespeichert.

```
130 INPUT "AN NULLSTELLE? (J/N) ";U$
```

```
140 IF UPRC$(U$)#"J" THEN INPUT "AN
    WELCHER STELLE ? ";Z @ GOTO 160
```

Die Stelle, an der das Polynom ausgewertet werden soll, kann sowohl reell als auch komplex sein.

```
150 DISP "WELCHE NULLSTELLE ?" @
    DISP USING '#,"(1...",K,")";D @
    INPUT E @ Z=R(E-1)
```

Geben Sie die Nummer der Nullstelle ein, an der Sie das Polynom auswerten wollen.

```
160 GOSUB 210 @ DISP "DER WERT DES "
    @ DISP "POLYNOMS IST" @ DELAY 8
    @ DISP Z @ DELAY 1
```

Die in Zeile 210 beginnende Unterroutine wertet das Polynom an der angegebenen Stelle aus. Dieser Wert wird dann angezeigt. Drücken Sie zur Fortsetzung eine beliebige Taste.

```
170 U$=KEY$ @ INPUT
    "NEUE AUSWERTUNG?(J/N)";U$
180 IF UPRC$(U$)="J" THEN 130
```

Falls gewünscht, wertet das Programm das Polynom an einer weiteren Stelle aus.

```
190 INPUT "NEUES POLYNOM? (J/N) ";U$
```

Das Programm kann für ein weiteres Polynom erneut gestartet werden.

```
200 IF UPRC$(U$)="J" THEN 20 ELSE STOP
210 COMPLEX B @ B=P(0)
```

Die Auswertung des Polynoms erfolgt mit Hilfe des Horner-Schemas.

```
220 FOR K=1 TO D
230 B=P(K)+Z*B
240 NEXT K
250 Z=B @ DESTROY B @ RETURN
260 DISP "RECHNEND..."
```

Mit dieser Zeile beginnt die Unterroutine zur Rekonstruktion der Koeffizienten. Durch die Akkumulation von Rundungsfehlern während der Rekonstruktion stimmen die rekonstruierten Koeffizienten möglicherweise selbst bei exakten Nullstellen nicht mit den ursprünglichen Koeffizienten überein.

```
270 MAT C=ZER @ C(D)=1
```

Erzeugt das Polynom 1 in **C**.

```
280 FOR L=1 TO D
```

Schleife zum Abruf der einzelnen Nullstellen.

```
290 IF IMPT(R(L-1))#0 THEN 340
300 FOR K=D-L TO D-1
310 C(K)=C(K+1)-C(K)*REPT(R(L-1))
320 NEXT K
330 C(D)=-C(D)*REPT(R(L-1)) @
    GOTO 400
```

Die Zeilen 300 bis 330 multiplizieren das in der Rekonstruktion befindliche Polynom mit einem linearen Faktor.

```
340 REAL B @ B=REPT(R(L-1))^2
    +IMPT(R(L-1))^2
```

Die Zeilen 340 bis 390 multiplizieren das in der Rekonstruktion befindliche Polynom mit einem quadratischen Faktor.

```
350 FOR K=D-L-1 TO D-2
```

```
360 C(K)=C(K+2)-2*REPT(R(L-1))
    *C(K+1)+B*C(K)
370 NEXT K
380 C(D-1)=-2*REPT(R(L-1))*C(K+1)
    +B*C(K)
390 C(D)=B*C(D) @ L=L+1
```

L wird inkrementiert, da das Polynom sowohl mit der komplexen Nullstelle als auch mit der konjugiert komplexen Nullstelle multipliziert wurde.

```
400 NEXT L
410 MAT C=(P(0))*C @ DESTROY B
    @ RETURN
```

Der führende Koeffizient des rekonstruierten Polynoms ist immer 1. Das rekonstruierte Polynom ist daher zu skalieren, wenn der ursprüngliche führende Koeffizient ungleich 1 war.

Beispiel. Es sollen nun die Nullstellen des Polynoms

$$x^6 + x^5 + x^4 + x^3 + x^2 + x + 1,$$

berechnet werden; anschließend soll die Güte der gefundenen Nullstellen durch Einsetzen überprüft werden.

Eingabe/Ergebnis

RUN

```
GRAD? ■
```

6 END LINE

```
KOEFFIZIENTENEINGABE
```

```
P(0)? ■
```

1,1,1,1,1,1,1 END LINE

```
RECHNEND...
```

```
DIE NULLSTELLEN SIND:
```

```
33956,-.974927912182)
```

Der Realteil der ersten Nullstelle wird während der Anzeige des Imaginärteils nach links aus der Anzeige hinausgeschoben.

g ◄

```
(-.222520933956,-.974
```

Der Realteil der ersten Nullstelle.

END LINE

```
933956,.974927912182)
```

Der Imaginärteil der zweiten Nullstelle.

g ◄

```
(-.222520933956,.9749
```

Der Realteil der zweiten Nullstelle.

Die verbleibenden Nullstellen können auf die gleiche Weise angezeigt werden:

Dritte Nullstelle: (-.900968867902,-.433883739118)

Vierte Nullstelle: (-.900968867902,.433883739118)

Fünfte Nullstelle: (.623489801859,.781831482468)

Sechste Nullstelle: (.623489801859,-.781831482468)

Die Programmausführung kann nach dem Anzeigen der letzten Nullstelle durch Drücken von END LINE fortgesetzt werden.

Eingabe/Ergebnis

```
REKONSTRUKTION?(J/N) ■
```

Jede Eingabe außer J oder j wird als "nein" interpretiert.

J END LINE

```
RECHNEND...
```

```
DIE REKONSTRUIERTEN
```

```
KOEFFIZIENTEN SIND:
```

```
 1
```

Der Koeffizient von x^6.

END LINE

```
 .99999999999
```

Der Koeffizient von x^5.

Die verbleibenden fünf Koeffizienten können auf die gleiche Weise angezeigt werden:

Koeffizient von x^4: 1

Koeffizient von x^3: .99999999998

Koeffizient von x^2: 1

Koeffizient von x^1: .99999999999

Koeffizient von x^0: 1

Die Programmausführung kann nach der Anzeige des letzten Koeffizienten durch Drücken von END LINE fortgesetzt werden.

Eingabe/Ergebnis

```
AUSWERTUNG? (J/N) ■
```

J END LINE

```
AN NULLSTELLE? (J/N) ■
```

J END LINE

```
WELCHE NULLSTELLE?
```

```
(1...6)? ■
```

```
1 END LINE
```

```
   DER WERT DES

   POLYNOMS IST:

    (0,-7.52E-13)

END LINE
```

Fortsetzung der Programmausführung.

```
   NEUE AUSWERTUNG?(J/N)

J END LINE

   AN NULLSTELLE? (J/N)

N END LINE

   AN WELCHER STELLE ?

(-.2,.9) END LINE

   DER WERT DES

   POLYNOMS IST:

    (.222523,.185814)

END LINE

   NEUE AUSWERTUNG?(J/N)

N END LINE

   NEUES POLYNOM? (J/N)

N END LINE
```

Beendet die Programmausführung.

Über den Algorithmus

Das Mathematik-Paket verwendet ein Iterationsverfahren, das sogenannte Laguerre-Verfahren, zur Bestimmung der Nullstellen eines Polynoms. Bei diesem Verfahren werden die Nullstellen einzeln nacheinander berechnet, indem zunächst eine Folge von Approximationen Z_1, Z_2, ... für eine Nullstelle erzeugt wird. Zum Aufbau dieser Folge wird die Formel $Z_{k+1} = Z_k + S_k$ benutzt, wobei S_k (der sogenannte Laguerre-Schritt) definiert ist als:

$$\frac{-N * P(Z_k)}{P'(Z_k) \pm [(N - 1)^2 * (P'(Z_k))^2 - N * (N - 1) * P(Z_k) * P''(Z_k)]^{1/2}}$$

Hier bezeichnen P, P' und P'' das Polynom und seine ersten beiden Ableitungen; n ist der Grad des Polynoms, und über das Vorzeichen des Nenners wird die Wertigkeit des Laguerre-Schritts verringert. Polynome oder deren Quotienten vom Grad 1 oder 2 werden direkt bzw. über die bekannte quadratische Formel aufgelöst.

Das Verfahren von Laguerre ist kubisch konvergent für einfache Nullstellen und linear konvergent für mehrfache Nullstellen.

Die Funktion PROOT ist in dem Sinne global, daß Sie weder einen Anfangs- oder Startwert noch ein Abbruchkriterium angeben müssen; d.h. der Algorithmus benötigt keine Vorabinformation über die Lage der Nullstellen. PROOT versucht immer, die Suche (Iteration) im Ursprung der komplexen Ebene zu beginnen. Es wird ein ringförmiges Gebiet bestimmt (mit Hilfe von fünf theoretischen Grenzwerten), das die betragsmäßig kleinste Nullstelle des (ursprünglichen oder Quotienten-) Polynoms enthält, und der erste Laguerre-Schritt wird zurückgewiesen, wenn er aus diesem Gebiet führen würde. In diesem Fall beginnt der Algorithmus eine spiralförmige Suche vom Inneren zum äußeren Rand des Ringgebiets, die solange fortgesetzt wird, bis ein akzeptabler Startwert gefunden ist.

Nach Beginn des Iterationsprozesses wird (mit Hilfe von zwei theoretischen Schranken) um jeden Iterationswert ein die Nullstelle enthaltender Kreis berechnet. Der Laguerre-Schritt wird permanent überprüft und modifiziert, wenn er aus diesem Kreis herausführen oder nicht zu einem kleineren Polynomwert führen würde. Die Nullstellen werden normalerweise in der Reihenfolge ansteigender Wertigkeit gefunden, was zu einer Abnahme des durch die Deflation bedingten Rundungsfehlereinflußes führt.

Die Auswertung des Polynoms und seiner Ableitungen mit einer reellen Iterationsvariablen Z_k entspricht genau dem Horner-Schema. Die Auswertung mit einer komplexen Iterationsvariablen ist ein modifiziertes Horner-Schema, bei dem etwa die Hälfte der Multiplikationen eingespart werden. Diese Modifikation nutzt die Symmetrie von konjugiert komplexen Lösungen der Horner'schen Rekursion aus.

PROOT bestimmt anhand einer sehr verfeinerten Technik, ob eine Approximation Z_k als Nullstelle akzeptiert werden kann. Bei jeder Auswertung des Polynoms an der Stelle Z_k wird gleichzeitig eine Schranke für den bei der Auswertung auftretenden Rundungsfehler berechnet. Ist der Wert des Polynoms kleiner als diese Schranke, wird Z_k als Nullstelle akzeptiert, wenn der Wert des Polynoms abnimmt und gleichzeitig die Größe des Laguerre-Schritts vernachlässigbar klein wird. Vor der

Auswertung des Polynoms an der Stelle Z_k wird der Imaginärteil von Z_k auf Null gesetzt, wenn dieser Wert klein im Vergleich zur Schrittweite ist. Dies verbessert die Geschwindigkeit des Algorithmus, da reellwertige Auswertungen weniger Zeit als komplexe Auswertungen beanspruchen. Wenn der Wert des Laguerre-Schritts vernachlässigbar klein geworden ist, aber der Wert des Polynoms immer noch nicht abnimmt, wird die Meldung `PROOT failure` angezeigt und die Berechnung abgebrochen. Dieser Fall sollte in der Praxis niemals auftreten.

Bei der Auswertung des Polynoms werden gleichzeitig die Koeffizienten des durch Division durch den zu Z_k gehörenden linearen oder quadratischen Faktor entstehenden Quotientenpolynoms berechnet. Wenn die Approximation Z_k die obigen Nullstellenkriterien erfüllt, wird dieses Quotientenpolynom zu dem (Rest-) Polynom, dessen Nullstellen noch zu bestimmen sind, und der gesamte Suchprozess wird wieder von Anfang an durchlaufen.

Mehrfache Nullstellen

Keine Routine zur Nullstellenbestimmung bei Polynomen — einschließlich `PROOT` — kann in konsistenter Weise Nullstellen hoher Ordnung mit beliebiger Genauigkeit berechnen. Als Faustregel für `PROOOT` gilt, daß mehrfache oder nahezu mehrfache Nullstellen mit einer Genauigkeit von ungefähr $12/K$ signifikanten Stellen, wo K die Ordnung der Nullstelle ist, berechnet werden.

Genauigkeit

`PROOT` verwendet als Genauigkeitskriterium, daß die Koeffizienten des aus den berechneten Nullstellen rekonstruierten Polynoms sehr dicht an den Koeffizienten des ursprünglichen Polynoms liegen sollten.

Die Leistungsstärke von `PROOT` bei isolierten Nullstellen wird beispielsweise durch das Polynom

$$P(Z) = \sum_{k=0}^{100} k$$

eindrucksvoll illustriert. Von den 200 reellen und imaginären Komponenten der Nullstellen werden über die Hälfte mit 12-stelliger Genauigkeit berechnet. Bei den restlichen Nullstellen ist der Fehler nicht größer als einige Einheiten in der 12-ten Stelle.

Bei der Anwendung von `PROOT` auf das Polynom $(Z + 1)^{20}$ (das -1 als Nullstelle der Ordnung 20 hat) werden die folgenden Nullstellen berechnet:

(−.997874038627,0)
(−.934656570635,0)
(−.947080146258,−.160105886062)
(−.947080146258,.160105886062)
(−.678701343788,−6.24034855342E−2)
(−.678701343788,6.24034855342E−2)

(−.815082852233,−.270565874916)
(−.815082852233,.270565874916)
(−.725960092383,−.178602450179)
(−.725960092383,.178602450179)
(−.934932478844,−.326980158732)
(−.934932478844,.326980158732)
(−1.06905713438,−.337946194292)
(−1.06905713438,.337946194292)
(−1.19977533452,−.295162714497)
(−1.19977533452,.295162714497)
(−1.30383056467,−.200016185042)
(−1.30383056467,.200016185042)
(−1.3593147483,7.00833934259E−2)
(−1.3593147483,−7.00833934259E−2)

Die berechneten Nullstellen sind aufgrund der hohen Ordung von −1 als Nullstelle ungenau. Nach der zuvor erwähnten Faustregel wäre keine oder vielleicht eine signifikante Stelle zu erwarten; die erste berechnete Nullstelle ist jedoch genauer. Wenn das Polynom aus den berechneten Nullstellen rekonstruiert wird, stimmen die (auf 12 Stellen gerundeten) Koeffizienten des rekonstruierten Polynoms sehr gut mit den Koeffizientendes ursprünglichen Polynoms überein.

Ursprüngliche Koeffizienten	Rekonstruierte Koeffizienten
1	1
20	20
190	190.000000001
1140	1140
4845	4845.00000003
15504	15504
38760	38760.0000003
77520	77520.0000007
125970	125970.000001
167960	167960.000002
184756	184756.000002
167960	167960.000003
125970	125970.000002
77520	77520.0000015
38760	38760.0000009
15504	15504.0000004
4845	4845.00000011
1140	1140.00000004
190	190.000000042
20	20.0000000344
1	1.00000001018

Geschwindigkeit von PROOT

Die Rechenzeitanforderungen von PROOT können der nachstehenden Tabelle entnommen werden. Die angegebenen Zeiten werden zur Berechnung *aller* Nullstellen des Polynoms

$$P(Z) = \sum_{k=0}^{N} k$$

für die unter Grad aufgeführten Werte von N benötigt.

Die angegebenen Zeiten sind ungefähre Zeiten.

Grad	Zeit (in Sek.)
3	3
5	6
10	22
15	42
20	142
30	168
50	568
70	1060
100	2101

Abschnitt 13

Finite Fouriertransformation

Schlüsselwort

Finite Fouriertransformationen stellen eine sehr weit verbreitete Methode zur Lösung einer Vielzahl von Problemstellungen aus den Bereichen der Ingenieurwissenschaften, Physik und Mathematik dar. Bekannte Anwendungen kommen beispielsweise in der Signalverarbeitung und bei der Lösung von Differentialgleichungen vor.

Für einen gegebenen Satz von N komplexen Datenpunkten Z_0, Z_1,..., Z_{N-1} liefert die finite Fouriertransformation einen zugehörigen Satz von N komplexen Werten W_0, W_1,..., W_{N-1}, so daß für $k = 0, 1, \ldots, N-1$ gilt:

$$Z_k = \sum_{j=0}^{N-1} W_j \left(\cos \frac{2\pi kj}{N} + i \sin \frac{2\pi kj}{N} \right)$$

Die W's repräsentieren dann die komplexen Amplituden der verschiedenen Frequenzkomponenten des durch die Datenpunkte dargestellten Signals und ergeben sich aus der Formel:

$$W_j = 1/N \sum_{k=0}^{N-1} Z_k \left(\cos \frac{2\pi kj}{N} - i \sin \frac{2\pi kj}{N} \right)$$

Diese Formel gilt für jede beliebige Anzahl von Datenpunkten. Das Mathematik-Paket verwendet den Algorithmus von Cooley-Tukey, der hier in der Maschinensprache des HP-71 codiert ist. Dies führt zu sehr guten Ergebnissen hinsichtlich Ausführungszeiten und Genauigkeit bei der Berechnung von finiten Fouriertransformierten; bedingt jedoch andererseits die Einschränkung, daß die Anzahl N der komplexen Datenpunkte eine positive ganzzahlige Potenz von 2 sein muß (etwa 2, 4, 8, 16, 32, 64, 128 usw.).

Um die finite Fouriertransformation anwenden zu können, sind zunächst die komplexen Datenpunkte $Z_0, \ldots, Z_{N-1}$ in aufeinanderfolgenden Elementen eines N-elementigen Felds abzulegen; Z_0 als erstes Element, Z_1 als zweites Element, usw. Das Format des Felds ist dabei unwichtig; es kommt nur darauf an, daß die Gesamtanzahl der Elemente in dem Feld mit der Anzahl der komplexen Datenpunkte übereinstimmt. So kann beispielsweise jedes der folgenden 8-elementigen Felder.

$$\begin{bmatrix} (1,2) \\ (3,4) \\ (5,6) \\ (7,8) \\ (9,10) \\ (11,12) \\ (13,14) \\ (15,16) \end{bmatrix}$$

$$\begin{bmatrix} (1,2) & (3,4) \\ (5,6) & (7,8) \\ (9,10) & (11,12) \\ (13,14) & (15,16) \end{bmatrix}$$

$$\begin{bmatrix} (1,2) & (3,4) & (5,6) & (7,8) \\ (9,10) & (11,12) & (13,14) & (15,16) \end{bmatrix}$$

$$\begin{bmatrix} (1,2) & (3,4) & (5,6) & (7,8) & (9,10) & (11,12) & (13,14) & (15,16) \end{bmatrix}$$

die Eingabedaten

$$\{(1,2),(3,4),(5,6),(7,8),(9,10),(11,12),(13,14),(15,16)\}$$

repräsentieren. Das Feld, in dem die Koeffizienten der Fouriertransformierten zu speichern sind, muß komplex sein. Wenn das Ergebnisfeld ein Vektor ist, wird dieser bei N Eingabedaten auf N Elemente umdimensioniert. Ist das Ergebnisfeld eine Matrix, wird diese auf N Zeilen und 1 Spalte umdimensioniert. Die Koeffizienten $W_0,\ldots,W_{N-1}$ der finiten Fouriertransformierten werden dann als komplexe Zahlen in aufeinanderfolgenden Elementen dieses komplexen N-elementigen Ergebnisfelds in der gleichen Reihenfolge wie die Datenpunkte zurückgegeben.

Außer der Einschränkung, daß N eine nichtnegative ganzzahlige Potenz von 2 sein muß, ist die Anzahl der Datenpunkte nur durch die Größe des verfügbaren Speicherbereichs begrenzt.

FOUR

Finite Fouriertransformierte

MAT **W**=FOUR(**Z**)

wo **Z** ein komplexer Vektor oder eine komplexe Matrix mit *N* Elementen, *N* (eine nichtnegative ganzahlige Potenz von 2) die Anzahl der komplexen Datenpunkte und **W** ein komplexer Vektor oder eine komplexe Matrix ist.

Wenn **W** ein Vektor ist, wird **W** implizit auf *N* Elemente umdimensioniert; wenn **W** eine Matrix ist, wird **W** implizit auf *N* Zeilen und eine Spalte umdimensioniert. Weist **W** die komplexen Koeffizienten der finiten Fouriertransformierten für die durch **Z** gegebenen Punkte zu.

Die Operation kann durch zweimaliges Drücken von [ATTN] angehalten werden.

Kann nicht im CALC-Modus verwendet werden.

Beispiel

In dem nachstehenden Beipiel wird die finite Fouriertransformation für den Datensatz ((1,2), (3,4), (5,6), (7,8), (9,10), (11,12), (13,14), (15,16)) berechnet.

```
10 OPTION BASE 1
20 COMPLEX SHORT A(8),B(1,2)

30 MAT INPUT A
40 MAT B=FOUR(A)
50 MAT DISP B
```

A enthält den Datensatz, **B** enthält nach einer Umdimensionierung die Fourierkoeffizienten.

[RUN]

```
A(1)? ■
```

```
(1,2),(3,4),(5,6),(7,8),(9,10),
(11,12),(13,14),(15,16)
END LINE
```

```
 (8,9)
 (-3.4142,1.4142)
 (-2,0)
 (-1.4142,-.58579)
 (-1,-1)
 (-.58579,-1.4142)
 (0,-2)
 (1.4142,-3.4142)
```

Weitere Informationen

Geschwindigkeit des Algorithmus

In nachstehenden Tabelle ist die von FOUR zur Zurückgabe der Transformierten benötigte Zeit in Abhängigkeit von der Anzahl der Datenpunkte angegeben.

Anzahl der Datenpunkte	Rechenzeit (Sekunden)
1	0.07
2	0.11
4	0.26
8	0.75
16	1.9
32	4.7
64	11
128	25
256	55
512	120
1024	260
2048	558

Zusammenhang zwischen finiten und kontinuierlichen Fouriertransformierten

Die finite (oder endliche) Fouriertransformierte wird in der Regel als Approximation der kontinuierlichen (oder unendlichen) Fouriertransformierten benutzt. Zur Begriffsklärung, in welchem Sinn dies eine Approximation darstellt, und zum Verständnis der Auswirkungen der verschiedenen Auswahlen, die bei der Verwendung dieser Approximation zu treffen sind, ist es hilfreich, den direkten Zusammenhang zwischen finiten und kontinuierlichen Fouriertransformationen zu betrachten.

Für eine komplexwertige Funktion $Z(x)$ ist die kontinuierliche Fouriertransformierte definiert als

$$W(f) = \int_{-\infty}^{\infty} Z(x) \exp(-2\pi i f x)\, dx$$

$Z_0, Z_1, \ldots, Z_{N-1}$ seien N komplexe Datenpunkte, die durch Abtasten der Funktion an N äquidistanten Stützstellen gebildet werden; d.h.

$$Z_k = Z(x_0 + k\Delta x) \text{ für } k = 0, 1, \ldots, N-1,$$

Die zu diesem Satz von Datenpunkten gehörende finite Fouriertransformierte $W_0, W_1, \ldots, W_{N-1}$ hängt mit der kontinuierlichen Fouriertransformierten $W(f)$ wie folgt zusammen. Für $k = 0, \ldots, N-1$ gilt

$$W_k = (r/N)\, \tilde{W}(k/N\Delta x) \text{ wo } r = \exp(-2\pi i x_0).$$

Hier ist $\tilde{W}$ eine Approxiomation der wahren kontinuierlichen Fouriertransformierten W. Um $\tilde{W}$ aus W abzuleiten, ist W auf zwei sehr verschiedene Weisen zu mitteln. Die dabei auftretende erste Gewichtung kann durch die Definition einer neuen Funktion $A(f)$ beschrieben werden, die einen Zwischenschritt zwischen W und $\tilde{W}$ bildet:

$$A(f) = \sum_{k=-\infty}^{\infty} W(f + k/\Delta x)$$

Dies besagt, daß der Wert A in einem Punkt f gleich der Summe der Werte von W an allen Punkten, die ganzzahlige Vielfache der von f ausgehenden begrenzenden Frequenz $1/\Delta x$ darstellen, ist. A besteht also aus unendlich vielen, in Abständen von $1/\Delta x$ Einheiten vom Ursprung weg angeordneten Wiederholungen des Frequenzbandes W einer bandbegrenzten Funktion. Dies ist eine Folge der Verwendung der *finiten* Fouriertransformation mit endlich großen Δx. Wenn nun Δx zu groß gewählt wird, kann dies bei der Fouriertransformierten zu *Frequenzbandüberlappungen* führen. Da die meisten der in praktischen Anwendungen vorkommenden Funktionen (und alle reellwertigen Funktionen) kontinuierliche Fouriertransformierte besitzen, die symmetrisch zum Ursprung sind, tritt für jede in W vorkommende Frequenz f_0 auch die Frequenz $-f_0$ in W auf. Aus diesem Grund sollten Sie Δx so wählen, daß $1/\Delta x$ größer als die doppelte Bandbreite der Funktion, d.h. zwei mal der Abstand zwischen der kleinsten und größten Frequenz, ist. Wenn die finite Fouriertransformierte alle, also auch die negativen Frequenzen ohne *Frequenzbandüberlappungen* enthalten soll, und Δf die größte in der Funktion enthaltene Frequenz ist, muß die Bedingung $\Delta f \Delta x < 1/2$ erfüllt sein.

Die zweite beim Übergang von W zu $\tilde{W}$ auftretende Gewichtung ist mehr lokaler Natur und führt zu einem Verlust in der Frequenzauflösung in $\tilde{W}$ im Vergleich zu W. Genauer gesagt gilt:

$$\tilde{W}(f) = (N\Delta x) \int_{-\infty}^{\infty} \text{sinc } (gN\Delta x)\ A(f - g)\ dg$$

$$\text{wo sinc } (a) = \begin{cases} 1 & \text{wenn } a = 0, \\ \dfrac{\sin (\pi a)}{\pi a} & \text{sonst} \end{cases}$$

Da sinc $(gN\Delta x)$ im wesentlichen aus einem höckerartigen Bereich um den Ursprung mit einer Breite, die umgekehrt proportional zu $N\Delta x$ ist, besteht, ist $\tilde{W}$ für kleine Werte von $N\Delta x$ mehr verschwommen (im Vergleich zu W). Dies ist kein besonderes Problem, solange W nicht einen großen Wert an einer Frequenz besitzt, die nicht ein Vielfaches der Grundfrequenz $N/\Delta x$ ist. In diesem Fall führen die "Seitenschwingungen" der sinc Funktion zu spürbaren Auswirkungen auf $\tilde{W}$. Dieser Effekt läßt sich etwas reduzieren,wenn die Datenpunkte Z_k vor der Bildung der finiten Fouriertransformierten mit einer glättenden (Fenster-) Funktion $G(k)$ multipliziert werden. Dies führt zu einer Gewichtungsfunktion, die kleinere Seitenschwingungen als sinc hat. Ein Beispiel für eine derartige Funktion ist die Hanning-Funktion $G(k) = (1/2)(1 - \cos (2\pi k/N))$.

Inverse finite Fouriertransformation

Viele Anwendungen der finiten Fouriertransformierten beinhalten die Berechnung der Transformierten für einen Satz von Datenpunkten, eine anschließende Manipulation der transformierten Werte (etwa Vergrößern oder Verkleinern der Amplituden) und schließlich eine Rücktransformation der Daten über die inverse finite Fouriertransformierte

$$Z_k = \sum_{j=0}^{N-1} W_j \left(\cos \frac{2\pi kj}{N} + i \sin \frac{2\pi kj}{N} \right)$$

Sie können mittels FOUR auch auf einfach Weise die inverse finite Fouriertransformierte bestimmen. Wenn W ein N-elementiges komplexes Feld ist, für das die inverse finite Fouriertransformierte bestimmt werden soll, können Sie wie folgt vorgehen:

1. Dimensionieren Sie W auf N Zeilen und eine Spalte um. (Wenn W ein Vektor oder ein Feld mit einer Spalte ist, ist keine Umdimensionierung notwendig.)
2. Transponieren Sie W (mit TRN). Dies liefert die konjugiert Komplexe von W, ohne daß dabei die Reihenfolge der Elemente geändert wird.
3. Bilden Sie die finite Fouriertransformierte des Ergebnisses.
4. Transponieren Sie das Ergebnis der finiten Fouriertransformation und multiplizieren Sie es mit N. Das Produkt stellt dann die inverse finite Fouriertransformierte des ursprünglichen Felds dar.

Beispiel. Dieses Beispiel illustriert diese Anwendung der finiten Fouriertransformation und die obige Prozedur zur Bestimmung der inversen finiten Fouriertransfomierten.

Gesucht sei die stationäre Lösung $Z(x)$ der inhomogenen Differentialgleichung

$$Z''(x) + 3Z'(x) + 12Z(x) = P(x)$$

wo $P(x)$ eine Funktion sei, für die Stichprobedaten vorliegen. Wenn $\tilde{Q}$ die (kontinuierliche) Fouriertransformierte einer beliebigen Funktion Q bezeichnet, führt die Bildung der Fouriertransformierten für die obige Gleichung zu

$$-f^2\tilde{Z}(f) + 3if\tilde{Z}(f) + 12\tilde{Z}(f) = \tilde{P}(f)$$

Diese Gleichung ist algebraisch lösbar:

$$\tilde{Z}(f) = \frac{\tilde{P}(f)}{(-f^2 + 12) + 3if}$$

Mit einer hinreichend guten Approximation für $\tilde{P}$ läßt sich die rechte Seite dieser Gleichung problemlos berechnen. Aus diesem Ergebnis kann dann die Lösung der ursprünglichen Gleichung über die inverse Fouriertransformierte bestimmt werden.

Aus Gründen der Vereinfachung sei hier unterstellt, daß $P(x)$ durch entsprechende Skalierung der Gleichung eine Einheitsperiode besitzt, und daß die höchste Frequenzkomponente von P (ungefähr) dem 30-fachen der Grundfrequenz enspricht. In diesem Fall werden dann Frequenzbandüberlappungen vermieden, wenn P 64 Mal pro Periode abgetastet wird.

Um die Eingabe von 64 komplexen Datenpunkten als Meßdaten für P zu vermeiden, verwendet das nachstehende Programm eine relativ einfache Funktion für P. Diese Werte könnten natürlich auch über jede andere Quelle vorgegeben werden.

```
10 OPTION BASE 1 @ DESTROY ALL
20 COMPLEX P(64),Q(64,1),Z(1,64)
30 COMPLEX T
40 DISP "RECHNEND; BITTE WARTEN"
50 RADIANS
```

P soll die Datenpunkte, die die Meßwerte für P repräsentieren, aufnehmen. Q dient der Speicherung von $\tilde{P}$ und $\tilde{P}/(-f^2 + 3if + 12)$. Z wird schließlich die Lösung der Differentialgleichung enthalten.

T ist ein komplexer Skalar für die komplexe Division.

```
60 FOR I=1 TO 64
70 R=PI*I/32
80 P(I)=( 6000*COS(3*R)*SIN(7.5*R)*
     COS(5.5*R) , 4000*COS(13*R)+
     3500*SIN(11*R) )
90 NEXT I
100 MAT Q=FOUR(P)
110 FOR F=-31 TO 32
120 J=MOD(F,64)+1
130 T=(-F^2+12,3*F)
140 Q(J,1)=Q(J,1)/T
150 NEXT F
160 MAT Q=TRN(Q)
170 MAT Z=FOUR(Q)
180 MAT Z=TRN(Z)
190 MAT Z=(64)*Z
200 COMPLEX Z(64,1)
210 DISP "LOESUNG:"
220 MAT DISP USING
     "C(MDDD.D,MDDD.D)";Z
```

Routine zur Belegung von P mit Abtastwerten der komplexwertigen Funktion, die durch die rechte Seite der Zeile 80 repräsentiert wird. Dabei wird eine Auswertung von 64 äquidistanten Punkten unterstellt.

Q repräsentiert nun $\tilde{P}$.

F stellt die Frequenzvariable dar und tastet den gesamten Bereich aller positiven und negativen Frequenzen ab, deren Auftreten in $\tilde{P}$ zu erwarten ist.

J ist die Nummer der Zeile im Feld Q, die die Amplitude der Frequenz F enthält.

T ist der Nenner des komplexen Quotienten.

Q enthält nun $\tilde{P}/(-f^2 + 3if + 12)$.

Routine zur Belegung von Z mit den Werten der inversen finiten Fouriertransformierten von Q. Die komplex Konjugierte von Q wird hier durch Transposition gebildet.

Hier wird die Konjugation ebenfalls durch Transposition gebildet.

Die angezeigten Werte stellen die komplexen Funktionswerte der stationären Lösung der Differentialgleichung für 64 äquidistante Punkte in einer Periode dar.

Sinus/Cosinus-Fourierreihen

Für rein reelle Datenpunkte Z_k kann anstelle der finiten Fouriertransformierten eine Fourierreihen-Transformierte benutzt werden. Hier wird für einen Satz von $2N$ (reellen) Datenpunkten $Z_0, Z_1, \ldots, Z_{2N-1}$ ein Satz von $2N + 1$ reellen Konstanten $A_0, A_1, \ldots, A_N, B_1, \ldots, B_N$ berechnet, die die Gleichungen

$$Z_k = \frac{A_0 2}{2} + \sum_{j=1}^{N} A_j \cos \frac{2\pi jk}{2N} + B_j \sin \frac{2\pi jk}{2N}$$

erfüllen.

Wenn $W_0, W_1, \ldots, W_{2N-1}$ die komplexen Koeffizienten der finiten Fouriertransformierten für die reellen Datenpunkte $Z_0, \ldots, Z_{2N-1}$ darstellen, ergeben sich die Koeffizienten der Fourierreihe aus

$$A_j = 2\text{REPT}(W_j) \qquad \text{für } j = 0, \ldots, N-1,$$

$$A_N = \text{REPT}(W_N)$$

$$B_j = -2\text{IMPT}(W_j) \qquad \text{für } j = 1, \ldots, N.$$

Anhang A

Benutzerinformation

Einsetzen und Entfernen des Moduls

Sie können das Mathematik-Modul in jeden der vier ROM-Einschubschächte auf der Vorderseite des Computers einsetzen.

VORSICHT

- Achten Sie darauf, daß der HP-71 (durch Drücken von [f] [OFF]) ausgeschaltet ist, bevor Sie irgendein Applikations-Modul einsetzen oder entfernen.
- Wenn Sie ein Modul entfernt haben, um das Mathematik-Modul einsetzen zu können, sollten Sie zum Zurücksetzen interner Zeiger den Computer vor dem Einsetzen des Mathematik-Moduls ein- und ausschalten.
- Stecken Sie keine Finger, Werkzeuge oder sonstige Fremdobjekte in die Einschubschächte des Computers. Die Nichtbeachtung dieser Vorsichtsmaßnahme kann zu geringfügigen elektrischen Schlägen und Störungen von Herzschrittmacherfunktionen führen. Des weiteren könnten die Kontakte in den Einschubschächten sowie die internen Schaltkreise des Computers beschädigt werden.
- Sollte das Modul beim Einsetzen klemmen, könnten Sie es verkehrt herum halten. Der Versuch, das Modul mit Gewalt in den Einschubschacht zu drücken, kann zu einer Beschädigung des Computers oder des Moduls führen.
- Behandeln Sie nichteingesetzte Einsteck-Module besonders vorsichtig. Führen Sie keine Gegenstände in die Kontaktbuchsen des Moduls ein. Verschließen Sie des weiteren nichtbenutzte Einschubschächte immer mit Modulattrappen. Die Nichtbeachtung dieser Vorsichtsmaßnahmen kann zu einer Beschädigung des Moduls oder des Computers führen.

Gewährleistung

Hewlett-Packard gewährleistet, daß das Mathematik-Modul in Bezug auf elektronische Bauteile und mechanischen Aufbau, jedoch nicht im Bezug auf die Software frei von Material- und Verarbeitungsschäden ist, und verpflichtet sich, etwaige fehlerhafte Teile kostenlos instandzusetzen oder auszutauschen, wenn das Modul — direkt oder über einen HP-Vertragshändler — an Hewlett-Packard eingesandt wird. Die Gewährleistung beträgt 12 Monate ab Verkaufsdatum.

Weitergehende Ansprüche, insbesondere auf Ersatz von Folgeschäden, können nicht geltend gemacht werden. Schäden, die auf unsachgemäße Veränderungen des Moduls durch Dritte zurückzuführen sind, werden von dieser Gewährleistung nicht umfaßt.

Diese Gewährleistung gilt nur in Verbindung mit entweder

- dem von einem Hewlett-Packard Vertragshändler ausgestellten Kaufbeleg oder
- der Originalrechnung von Hewlett-Packard.

Die Ansprüche des Käufers aus dem Kaufvertrag bleiben von dieser Gewährleistungsregelung unberührt.

Änderungsverpflichtung

Sämtliche Produkte werden auf der Grundlage der technischen Daten bei der Herstellung verkauft. Hewlett-Packard übernimmt keine Verpflichtungen, einmal verkaufte Produkte zu modifizieren oder auf den neuesten Stand zu bringen.

Gewährleistungsinformation

Wenn Sie bezüglich dieser Gewährleistung Fragen haben, setzen Sie sich bitte mit einem Hewlett-Packard Vertragshändler in Verbindung. Falls dies nicht möglich ist, schreiben Sie an:

- In Europa:

Hewlett-Packard S.A.
150, route du Nant-d'Avril
P.O. Box
CH-1217 Meyrin 2 (Genf)
Schweiz
Telefon: (022) 83 81 11

Hinweis: Senden Sie keine Geräte zur Reparatur an diese Adresse.

- In den U.S.A.:

Hewlett-Packard Company
Portable Computer Division
1000 N.E. Circle Blvd.
Corvallis, OR 97330
U.S.A.
Telefon: (503) 758-1010

- In allen anderen Ländern:

Hewlett-Packard Intercontinental
3495 Deer Creek Rd.
Palo Alto, CA 94304
U.S.A.
Telefon: (415) 857-1501

Hinweis: Senden Sie keine Geräte zur Reparatur an diese Adresse.

Service

Serviceniederlassungen

Hewlett-Packard unterhält weltweit Serviceniederlassungen. Sie können Ihr Gerät jederzeit von einer Hewlett-Packard Serviceniederlassung reparieren lassen, sei es mit oder ohne Gewährleistung. Nach Ablauf der einjährigen Gewährleistungsfrist werden Ihnen die Reparaturkosten berechnet.

Servicezentrale in den Vereinigten Staaten

Die Servicezentrale für batteriebetriebene Computerprodukte von Hewlett-Packard in den U.S.A. befindet sich in Corvallis, Oregon:

Hewlett-Packard Company
Service Department
P.O. Box 999
Corvallis, OR 97339, U.S.A.
oder
1030 N.E. Circle Blvd.
Corvallis, OR 97330, U.S.A.
Telefon: (503) 757-2000

Serviceniederlassungen in Europa

Die folgende Aufstellung zeigt die Serviceniederlassungen in Europa. Setzen Sie sich in nicht aufgelisteten Ländern mit dem Händler in Verbindung, bei dem Sie Ihr Gerät erworben haben.

BELGIEN
HEWLETT-PACKARD BELGIUM SA/NV
Woluwedal 100
B-1200 BRÜSSEL
Tel. (02) 762 32 00

DÄNEMARK
HEWLETT-PACKARD A/S
Datavej 52
DK-3460 BIRKERØD (Kopenhagen)
Tel. (02) 81 66 40

DEUTSCHLAND
HEWLETT-PACKARD GmbH
Berner Strasse 117
Postfach 560 140
D-6000 FRANKFURT 56
Tel. (0611) 50 04 1

FINNLAND
HEWLETT-PACKARD OY
Revontulentie 7
SF-02100 ESPOO 10 (Helsinki)
Tel. (90) 455 02 11

FRANKREICH
HEWLETT-PACKARD FRANCE
Avenue des Tropiques
Z.I. de Courtaboeuf
F-91947 LES ULIS CEDEX
Tel. (6) 907 78 25

GROSSBRITANNIEN
HEWLETT-PACKARD Ltd
King Street Lane
GB-Winnersh, Wokingham
BERKSHIRE RG11 5AR
Tel. (0734) 784 774

ITALIEN
HEWLETT-PACKARD ITALIANA S.P.A.
Casella postale 3645 (Milano)
Via G. Di Vittorio, 9
I-20063 CERNUSCO SUL NAVIGLIO
Tel. (2) 90 36 91

NIEDERLANDE
HEWLETT-PACKARD NEDERLAND B.V.
Van Heuven Goedhartlaan 121
N-1181 KK AMSTELVEEN (Amsterdam)
P.O. Box 667
Tel. (020) 47 20 21

NORWEGEN
HEWLETT-PACKARD NORGE A/S
P.O. Box 34
Øesterndalen 18
N-1345 ØESTERAAS (Oslo)
Tel: (2) 17 11 80

ÖSTERREICH
HEWLETT-PACKARD Ges.m.b.H.
Lieblgasse 1
P.O. Box 72
A-1222 WIEN
Tel. (0222) 23 65 11 0

OSTEUROPA
Bitte wenden Sie sich an die unter Österreich angegebene Adresse.

SCHWEDEN
HEWLETT-PACKARD SVERIGE AB
Skalholtsgatan 9, Kista
Box 19
S-16393 SPANGA (Stockholm)
Tel. (08) 750 20 00

SCHWEIZ
HEWLETT-PACKARD (SCHWEIZ) AG
Allmend 2
CH-8967 WIDEN
Tel. (057) 31 21 11

SPANIEN
HEWLETT-PACKARD ESPAÑOLA S.A.
Crta.de la Coruna
Las Rozas
E-MADIRD 16
Tel. (1) 458 2600

Internationale Serviceinformation

Nicht alle Hewlett-Packard Serviceniederlassungen bieten Reparatur für alle HP-Produkte an. Wenn Sie Ihr Gerät bei einem HP-Vertragshändler erwoben haben, können Sie allerdings sicher sein, daß in dem Land, in dem das Gerät gekauft wurde, auch Service angeboten wird.

Wenn Sie sich außerhalb des Landes befinden, in dem Sie das Gerät gekauft haben, sollten Sie sich mit der örtlichen Hewlett-Packard Serviceniederlassung in Verbindung setzen und, falls die Reparatur dort nicht möglich ist, das Gerät an die unter "Servicezentrale in den Vereinigten Staaten" angegebene Adresse schicken. Von dort können Sie auch eine Liste der Serviceniederlassungen in anderen Ländern erhalten.

Sämtliche mit dem Versand verbundene Kosten gehen zu Ihren Lasten.

Reparaturkosten

Hewlett-Packard erhebt für Reparaturen, die außerhalb der Gewährleistungsfrist liegen, Gebühren nach einem festgesetzten Satz. In den Reparaturkosten sind Arbeitszeit und Materialien eingeschlossen. In der Bundesrepublik Deutschland wird auf den Rechnungsbetrag Mehrwertsteuer erhoben und auf der Rechnung getrennt ausgewiesen.

Die festgesetzten Reparatursätze gelten nicht für durch Gewalteinwirkung oder Mißbrauch beschädigte Produkte. In solchen Fällen werden die Reparaturkosten auf der Grundlage von Arbeitszeit- und Materialaufwand individuell festgelegt.

Gewährleistung auf Servicearbeiten

Auf alle Reparaturen außerhalb der Gewährleistungsfrist wird eine Garantie auf Material und Verarbeitung für einen Zeitraum von 90 Tagen ab dem Reparaturdatum gegeben.

Versandanweisungen

Wenn Ihr Gerät repariert werden muß, senden Sie es bitte mit den folgenden Unterlagen ein:

- Eine Beschreibung der Störung.
- Einen Kassenbeleg oder ein anderer Verkaufsnachweis, falls die einjährige Gewährleistungsfrist noch nicht abgelaufen ist.

Das Produkt, eine kurze Fehlerbeschreibung und gegebenenfalls der Beleg des Kaufdatums sollten zur Vermeidung von Versandschäden in der Originalpackung oder einer anderen angemessenen Schutzverpackung eingesandt werden. Versandschäden sind in der Jahresgarantie nicht eingeschlossen. Das verpackte Gerät sollte an die nächstliegende Hewlett-Packard Serviceniederlassung gesandt werden. Lassen Sie sich dazu von Ihrem Händler beraten. (Wenn Sie sich nicht in dem Land aufhalten, in dem Sie Ihr Gerät erworben haben, lesen Sie bitte den vorangegangenen Teilabschnitt "Internationale Serviceinformation").

Sonstiges

Hewlett-Packard bietet keine Serviceverträge an. Ausführung und Design des von Computerprodukten sind Eigentum von Hewlett-Packard; Servicehandbücher sind nicht für Kunden verfügbar. Sollten Sie weitere Fragen bezüglich Reparaturen haben, wenden Sie sich bitte an Ihre nächste Hewlett-Packard Serviceniederlassung.

Händler- und Produktinformation

Einen Bezugsquellennachweis über den Fachhandel, sowie Produkt- und Preisinformationen erhalten Sie in der Bundesrepublik Deutschland über:

Hewlett-Packard GmbH
Vertriebszentrale/Werbeabteilung
Berner Straße 117
Postfach 560 140
D-6000 Frankfurt/M 56

Anhang B

Speicheranforderungen

Das Mathematik-Modul belegt 43.5 Bytes des Schreib/Lese-Speichers (RAM) für den eigenen Bedarf. Zusätzlich fordern die einzelnen Routinen vorübergehend geringe Mengen an Speicherplatz für Overhead-Zwecke an. Wesentlich mehr Speicherplatz wird bei der Deklaration von komplexen Variablen und Feldern und beim Vergrößern von Feldern (während Umdimensionierungen) belegt. Der von den einzelnen Operationen des Mathematik-Pakets benötigte Speicherplatz kann der nachstehenden Tabelle entnommen werden.

Schlüsselwort	Speicheranforderung der Operation
Einfache Variable	
COMPLEX	25.5 Bytes
COMPLEX SHORT	18.5 Bytes
Feld	
COMPLEX	16*(Dimension 1 − OPTION BASE + 1) *(Dimension 2 − OPTION BASE + 1) + 9.5
COMPLEX SHORT	9*(Dimension 1 − OPTION BASE + 1) *(Dimension 2 − OPTION BASE + 1) + 9.5
DET(**A**)	$2N(4N+1)$ Bytes, wo **A** eine $N \times N$ Matrix ist.
MAT PRINT USING MAT DISP USING	14 Bytes
MAT INPUT	40 Bytes
MAT **A=A*A** MAT **A=A*B** MAT **A=B*A**	Belegt nur dann zusätzlichen Speicherplatz, wenn ein Operandenfeld auch als Ergebnisfeld spezifiziert wird. Wenn das Produkt (d.h. die umdimensionierte Matrix **A**) eine $M \times N$ Matrix ist (bei Vektoren ist $N = 1$), wird der folgende Speicherplatz belegt: 3*MN* Bytes, wenn **A** vom Typ INTEGER ist. 4.5*MN* Bytes, wenn **A** vom Typ SHORT ist. 8*MN* Bytes, wenn **A** vom Tvp REAL ist. 9*MN* Bytes, wenn **A** vom Typ COMPLEX SHORT ist. 16*MN* Bytes, wenn **A** vom Typ COMPLEX ist.
MAT **A=TRN(A)*A** MAT **A=TRN(A)*B**	Belegt nur dann zusätzlichen Speicherplatz, wenn ein Operandenfeld auch als Ergebnisfeld spezifiziert wird.

Schlüsselwort	Speicheranforderung der Operation
MAT **A**=TRN(**B**)*A	Wenn das Produkt (d.h. die umdimensionierte Matrix **A**) eine $M \times N$ Matrix ist (bei Vektoren ist $N = 1$), wird der folgende Speicherplatz belegt: 3*MN* Bytes, wenn **A** vom Typ INTEGER ist. 4.5*MN* Bytes, wenn **A** vom Typ SHORT ist. 8*MN* Bytes, wenn **A** vom Typ REAL ist. 9*MN* Bytes, wenn **A** vom Typ COMPLEX SHORT ist. 16*MN* Bytes, wenn **A** vom Typ COMPLEX ist.
MAT **B**=INV(**A**)	**A** sei eine $N \times N$ Matrix. Wenn **A** vom Datentyp REAL, SHORT oder INTEGER und **B** vom Datentyp REAL ist: 4*N* Bytes. Wenn **A** vom Datentyp REAL, SHORT oder INTEGER und **B** *nicht* vom Datentyp REAL ist: $4N(2N + 1)$ Bytes. Wenn **A** vom Datentyp COMPLEX oder COMPLEX SHORT ist: $8N(4N + 1)$ Bytes.
MAT **C**=SYS(**A**,**B**)	**A** sei eine $N \times N$ Matrix und **B** eine $N \times P$ Matrix (für Vektoren ist $P = 1$). Wenn **A** vom Datentyp REAL, SHORT oder INTEGER und **B** vom Datentyp REAL, SHORT oder INTEGER ist: $4N(2N + 4P + 1)$ Bytes. Wenn **A** vom Datentyp REAL, SHORT oder INTEGER und **B** vom Datentyp COMPLEX oder COMPLEX SHORT ist: $4N(2N + 8P + 1)$ Bytes. Wenn **A** vom Datentyp COMPLEX oder COMPLEX SHORT ist: $8N(4N + 4P + 1)$ Bytes.
MAT **A**=TRN(**A**)	Wenn **A** eine $M \times N$ Matrix und vom Datentyp INTEGER ist: $MN/2$ Bytes. Wenn Operanden- und Ergebnismatrix verschieden sind oder wenn **A** nicht vom Datentyp INTEGER ist, wird kein zusätzlicher Speicherplatz belegt.
MAT **B**=PROOT(**A**)	Wenn **A** ein Polynom *N*-ten Grades repräsentiert. $21N + 261$ Bytes.
MAT **B**=FOUR(**A**)	**A** enthalte *N* Elemente. Wenn **B** vom Datentyp COMPLEX SHORT ist: 16*N* Bytes. Wenn **B** vom Datentyp COMPLEX ist, wird kein zusätzlicher Speicherplatz belegt.
FNROOT	112.5 Bytes, wenn FNROOT nicht geschachtelt ist. Zusätzlich für jede Schachtelungsebene 96.5 Bytes.
INTEGRAL	208.5 Bytes, wenn INTEGRAL nicht geschachtelt ist. Zusätzlich für jede Schachtelungsebene 192.5 Bytes.

Anhang C

Fehlerbedingungen

Das Mathematik-Paket gibt zwei Arten von Fehlermeldungen zurück:

- Fehlermeldungen des Mathematik-Pakets. Diese Fehlermeldungen sind durch die LEX-Identifikationsnummer 2 gekennzeichnet und werden in der ersten Tabelle erläutert.
- Fehlermeldungen des HP-71, die vom Mathematik-Paket zurückgegeben werden. Diese Fehlermeldungen sind durch die LEX-Identifikationsnummer 0 gekennzeichnet und werden in der zweiten Tabelle erläutert.

Fehlermeldungen des Mathematik-Pakets

Nummer	Fehlermeldung und Fehlerbedingung
1	`#DIMS` • `DOT(`**A**`,`**B**`)`: **A** oder **B** ist eine Matrix. • `DET(`**A**`)`, `MAT` **B**`=INV(`**A**`)`, `MAT` **B**`=TRN(`**A**`)`, `MAT` **A**`=IDN`, `MAT` **X**`=SYS(`**A**`,`**Y**`)`: **A** oder **B** ist ein Vektor. • `MAT` **A**`=IDN(`*i*`)`: Es wurde nur ein Umdimensionierungsindex angegeben. • `MAT` **A**`=`*Operation*`(`*Operandenfeld(er)*`)`: Die Anzahl der Indizes von **A** entspricht nicht der vom Ergebnis der Operation benötigten Anzahl von Indizes.
2	`Not Square` • `DET(`**A**`)`, `MAT` **A**`=IDN`, `MAT` **B**`=INV(`**A**`)`, `MAT` **X**`=SYS(`**A**`,`**B**`)`: Die Anzahl der Spalten der Matrix **A** entspricht nicht der Anzahl der Zeilen. • `MAT` **A**`=IDN(`*i*`,`*j*`)`: $i \neq j$.

Nummer	Fehlermeldung und Fehlerbedingung
3	Conformability • MAT **A**=**B**+**C**, MAT **A**=**B**−**C**: **B** und **C** sind nicht vereinbar bezüglich Additionen (d.h. die Anzahl der Spalten oder die Anzahl der Zeilen stimmt nicht überein). • MAT **A**=**B***C: **B** und **C** sind nicht vereinbar bezüglich Multiplikationen (d.h. **B** ist ein Vektor oder die Anzahl der Spalten von **B** stimmt nicht mit der Anzahl der Zeilen von **C** überein). • MAT **A**=TRN(**B**)*C: **B** und **C** sind nicht vereinbar bezüglich transponierter Multiplikationen (d.h. **B** ist ein Vektor oder die Anzahl der Zeilen von **B** entspricht nicht der Anzahl der Zeilen von **C**). • MAT **X**=SYS(**A**,**B**): Obwohl **A** eine quadratische Matrix ist, sind **A** und **B** nicht vereinbar bezüglich Multiplikationen. • DOT(**A**,**B**): **A** und **B** sind Vektoren; jedoch entspricht die Anzahl der Elemente in **A** nicht der Anzahl der Elemente in **B**.
4	Parameter Redim • Das Ergebnisfeld einer MAT Anweisung ist ein Unterprogrammparameter. Die MAT Anweisung erfordert eine Umdimensionierung, bei der die Anzahl der Feldelemente geändert würde.
5	Nesting Error • Mehr als 5 Ebenen in FNROOT oder INTEGRAL Schachtelungen.
6	Kybd FN in FNROOT/INTEGRAL • Versuch der Ausführung von FNROOT oder INTEGRAL über das Tastenfeld im BASIC-Modus, wobei die Funktion, deren Nullstelle oder Integral zu berechnen ist, eine benutzerdefinierte Funktion ist. • Versuch der Ausführung einer benutzerdefinierten Funktion über das Tastenfeld, wenn die Ausführung von FNROOT oder INTEGRAL während der Berechnung der Funktion, deren Nullstelle oder Integral zu berechnen ist, angehalten wurde.
7	Function Interrupted • Die Ausführung von DET(**A**), CNORM(**A**), RNORM(**A**), FNORM(**A**) oder DOT(**A**,**B**) wurde durch zweimaliges Drücken von [ATTN] unterbrochen.
8	Bad Array Size • MAT **B**=FOUR(**A**): Die Anzahl der Elemente von **A** ist nicht eine positive ganzzahlige Potenz von 2. • MAT **B**=PROOT(**A**): **A** besteht aus genau einem Element (und repräsentiert damit ein Polynom vom Grad 0).
9	PROOT Failure • PROOT kann keine Nullstelle des spezifizierten Polynoms finden.

Nummer	Fehlermeldung und Fehlerbedingung
10	GAMMA=Inf • GAMMA(X): X ist eine ganze Zahl kleiner oder gleich 0.
11	ATANH(+-1) • ATANH(1) oder ATANH(-1)
Keine Fehler-Nummer	**Initialisierung** • Die Overhead-Speicheranforderungen des Mathematik-Pakets können wegen unzureichendem Speicherplatz nicht befriedigt werden. Das Mathematik-Paket belegt 43.5 Bytes des Systemspeichers für den eigenen Bedarf. Dieser Speicherplatz muß beim Einsetzen des Moduls verfügbar sein.

Fehlermeldungen des HP-71

Nummer	Fehlermeldung und Fehlerbedingung
11	Invalid Arg • BVAL(*B$*,*R*), BSTR$(*X*,*R*): Der ganzahlig gerundete Wert von *R* ist ungleich 2, 8 oder 16. • BVAL(*B$*,*R*): *B$* ist keine zulässige Stringdarstellung einer Zahl zur Basis *R*. • BSTR$(*X*,*R*): Der ganzahlig gerundete Wert von *X* liegt nicht im Interval [0,1E12). • BVAL(*B$*,*R*): Das Dezimaläquivalent von *B$* ist größer als 999 999 999 999. • LBND(**A**,*N*), UBND(**A**,*N*): Der ganzahlig gerundete Wert von *N* ist weder 1 noch 2. • In einer MAT CON, IDN, ZER, COMPLEX oder COMPLEX SHORT Anweisung wurde ein unzulässiger Index verwendet.
24	Insufficient Memory • Unzureichender Speicherplatz. In Anhang B sind die Speicherplatzanforderungen der Operationen des Mathematik-Pakets gelistet.
31	Data Type • Ein (reeller oder komplexer) Skalar wurde anstatt eines an dieser Stelle benötigten Felds verwendet. Entsprechendes gilt für den umgekehrten Fall. • Ein Skalar oder Feld vom Typ COMPLEX wurde anstatt eines an dieser Stelle benötigten Skalars oder Felds vom Typ REAL verwendet. Entsprechendes gilt für den umgekehrten Fall.

Nummer	Fehlermeldung und Fehlerbedingung
32	No Data • Versuch der Ausführung von DETL vor der ersten Ausführung von MAT . . . INV (mit reellwertigem Argument) oder von MAT . . . SYS (mit reellwertigem ersten Argument). • Versuch der Ausführung von FVALUE oder FGUESS vor der ersten Ausführung von FNROOT. • Versuch der Ausführung von IVALUE oder IBOUND, bevor INTEGRAL die Funktion, deren Integral zu berechnen ist, zum ersten Mal ausgewertet hat. • Versuch der Ausführung von FVAR, ohne daß FNROOT momentan eine Funktion, deren Nullstelle gesucht ist, auswertet. • Versuch der Ausführung von IVAR, ohne daß INTEGRAL momentan eine Funktion, deren Integral gesucht ist, auswertet.
46	Invalid USING • Formatierung eines reellen Ausdrucks mit einem komplexen Formatstring. Entsprechendes gilt für den umgekehrten Fall.
79	Illegal Context • Es wurde versucht, INTEGRAL oder FNROOT im CALC-Modus indirekt auszuführen.
80	Invalid Parameter • Ein als Antwort auf eine MAT INPUT Eingabeaufforderung eingegebener Ausdruck enthält einen Aufruf einer benutzerdefinierten Funktion.

Anhang D

Wirkung von [ATTN]

Die Wirkung der [ATTN] Taste während der Ausführung der nachstehend aufgeführten Schlüsselworte wurde bereits auf der angegebenen Seite erläutert.

Die Ausführung der in diesem Anhang gelisteten Schlüsselworte kann durch ein- oder zweimaliges Drücken von [ATTN] abgebrochen werden.

Feldausgabeanweisungen

Die Ausführung der Feldausgabeanweisungen des Mathematik-Pakets (MAT DISP/PRINT [USING]) kann jederzeit durch *einmaliges* Drücken von [ATTN] angehalten werden.

Weitere MAT Anweisungen

Die Ausführung der nachstehenden MAT Anweisungen kann jederzeit durch *zweimaliges* Drücken von [ATTN] angehalten werden.

MAT *Ergebnis* = [−] *Operand*

MAT *Ergebnis* = *Operand* +/−/* *Operand*

MAT *Ergebnis* = (*Skalar*) [* *Operand*]

MAT *Ergebnis* = INV(*Operand*)

MAT *Ergebnis* = SYS(*Operand* , *Operand*)

MAT *Ergebnis* = TRN(*Operand*)[* *Operand*]

MAT *Ergebnis* = FOUR(*Operand*)

MAT *Ergebnis* = PROOT(*Operand*)

Es sei unterstellt, Sie wollen ein langes Programm abbrechen, das eine MAT INV Anweisung enthält, und drücken dazu einmal [ATTN]. Die Programmausführung hält jedoch nicht an (d.h. die Statusanzeige **SUSP** erscheint nicht in der Anzeige). Dies deutet an, daß das Programm momentan die Anweisung MAT INV ausführt und Sie erhalten somit die Möglichkeit, zu entscheiden, ob Sie das Ergebnis der Ausführung von MAT INV abwarten oder die Programmausführung und damit auch die Ausführung der Anweisung durch ein weiteres Drücken von [ATTN] sofort abbrechen wollen. Diese Verwendung der Taste [ATTN] ermöglicht einen stufenweisen Programmabbruch.

Wenn Sie während der Ausführung einer MAT INV Anweisung die Taste [ATTN] nur einmal drücken, wird das Programm erst nach Abschluß der Ausführung der Anweisung abgebrochen.

Skalare Matrixfunktionen

Die Ausführung der nachstehenden skalaren Matrixfunktionen kann jederzeit durch *zweimaliges* Drücken von [ATTN] abgehalten werden.

DET(*Operand*)

DOT(*Operand* , *Operand*)

FNORM(*Operand*)

CNORM(*Operand*)

RNORM(*Operand*)

Die oben beschriebenen Vorteile des zweimaligen Drückens von [ATTN] treffen auch auf diese Funktionen zu. Da die Ausführung einer Funktion nur durch eine Fehlerbedingung unterbrochen wird, wird bei den obigen Funktionen nach zweimaligem Drücken von [ATTN] die Fehlermeldung Function Interrupted angezeigt.

Mathematische Ausnahmen und IEEE-Vorschlag

Einleitung

Dieser Anhang erläutert die Realisierung des IEEE-Vorschlags zur Behandlung von mathematischen Ausnahmen durch die Funktionen und Operationen des Mathematik-Pakets. Dies beinhaltet Berechnungen mit NaN und Inf Argumenten, das Setzen von mathematischen Ausnahmeflags, die Behandlung von bereichsüberschreitenden Argumenten, Fehlermeldungen oder Warnungen und Vorgabewerte für IVL und DVZ Ausnahmen. Der IEEE-Vorschlag zur Behandlung von mathematischen Ausnahmen wird im *HP-71 Referenzhandbuch* erläutert. Die Funktionen des Mathematik-Pakets setzen, wenn nötig, die Ausnahmeflags IVL, DVZ, OVF, UNF und INX und geben in Abhängigkeit von der für diese Flags geltenden Auffangwerte (der jeweiligen TRAP Einstellung) Fehlermeldungen oder Warnungen (zusammen mit den mit Vorgabewerten berechneten Ergebnissen) zurück. Die Definitionen bzw. Berechnungsformeln für einige der hier beschriebenen Funktionen finden Sie in den entsprechenden Abschnitten dieses Handbuchs.

Die in den Abschnitten 2 und 3 dieses Handbuchs beschriebenen Schlüsselworte und die nachstehend genannten Schlüsselworte setzen keine Ausnahmeflags: TYPE, - (Negation von komplexen Zahlen), CONJ, CON, IDN, ZER, MAT DISP/PRINT [USING], LBND, UBND, DETL, FVAR, FVALUE, FGUESS, IVAR, IVALUE und IBOUND. Die Ausnahmeflags INX, OVF und UNF werden vom Mathematik-Paket unter Umständen gesetzt, wenn Zahlenwerte zur Umwandlung in einen anderen Zahlentyp gerundet werden müssen (etwa bei der Zuweisung von (MAXREAL,MAXREAL) an eine COMPLEX SHORT Variable oder bei der Ausführung von MAT **A**=**B**, wobei **A** vom Typ INTEGER ist und **B** Elemente enthält, die größer als 99999 sind).

Die Anweisungen MAT **A**=**B**, MAT **A**=-**B**, MAT **A**=TRN(**B**) und MAT **A**=(*X*) setzen zusätzlich zu den beim Runden auftretenden Ausnahmen nur den Ausnahmeflag IVL (bei gleichzeitiger Anzeige der Meldung Signaled Op) und auch nur dann, wenn **A** vom Typ INTEGER ist und entweder **B** ein aktives NaN enthält oder *X* ein aktives NaN ist. Der Grund hierfür besteht darin, daß INTEGER-Variablen nur passive NaN's enthalten können. Entsprechendes gilt für MAT INPUT.

Die in den nachstehenden Tabellen für jedes Schlüsselwort gegebenen Fälle werden in der Reihenfolge von oben nach unten ausgewertet.

Hinweis: In diesem Anhang repräsentiert das Symbol ★ ein beliebiges Argument.

Reelle Skalarfunktionen

Diese Funktionen wurden in Abschnitt 4 dieses Handbuchs beschrieben. Jedes Argument mit einem aktiven NaN setzt den IVL-Flag und zeigt die Meldung Signaled Op an. Bei TRAP(IVL) = 2 wird NaN in den passiven Zustand versetzt und die Operation kann fortgesetzt werden. Außer bei der Funktion NaN$ gibt jedes passive NaN-Argument als Ergebnis NaN zurück, ohne daß dabei Ausnahmeflags gesetzt werden. (Die Funktionen IROUND und NaN$ setzen außer bei aktiven NaN-Argumenten keine Ausnahmeflags.)

Reeller Sinus Hyperbolicus (SINH(*X*))

Argument	Ergebnis
±Inf	±Inf; keine Ausnahmeflags gesetzt.
±0	±0; keine Ausnahmeflags gesetzt.
☆	INX gesetzt; UNF, OVF gesetzt, falls nötig.

Reeller Cosinus Hyperbolicus (COSH(*X*))

Argument	Ergebnis
±Inf	|Inf|; keine Ausnahmeflags gesetzt.
±0	1; keine Ausnahmeflags gesetzt.
☆	INX gesetzt; OVF gesetzt, falls nötig

Reeller Tangens Hyperbolicus (TANH(*X*))

Argument	Ergebnis
±Inf	SGN(±Inf); keine Ausnahmeflags gesetzt.
±0	±0
☆	INX gesetzt; UNF gesetzt, falls nötig.

Reeller Arcus Sinus Hyperbolicus (ASINH(*X*))

Argument	Ergebnis
±Inf	±Inf; keine Ausnahmeflags gesetzt.
±0	±0; keine Ausnahmeflags gesetzt.
☆	INX gesetzt; UNF gesetzt, falls nötig.

Reeller Arcus Cosinus Hyperbolicus (ACOSH(*X*))

Argument	Ergebnis
Inf	Inf; keine Ausnahmeflags gesetzt.
$X < 1$	IVL gesetzt; NaN als Ergebnis; Meldung: Invalid Arg.
1	0; keine Ausnahmeflags gesetzt.
☆	INX gesetzt.

Reeller Arcus Tangens Hyperbolicus (ATANH(*X*))

Argument	Ergebnis
$\|X\| > 1$	IVL gesetzt; NaN als Ergebnis; Meldung: Invalid Arg.
$\|X\| = 1$	DVZ gesetzt; Meldung: ATANH(+-1). SGN(*X*) × Inf als Ergebnis, falls TRAP(DVZ) = 2. SGN(*X*) × MAXREAL als Ergebnis und INX gesetzt, falls TRAP(DVZ) = 1.
±0	±0; keine Ausnahmeflags gesetzt.
☆	INX gesetzt; UNF gesetzt, falls nötig.

Logarithmus zur Basis 2 (LOG2(*X*))

Argument	Ergebnis
Inf	Inf; keine Ausnahmeflags gesetzt.
$X < 0$	IVL gesetzt; NaN als Ergebnis; Meldung: LOG(neg).
±0	DVZ gesetzt; Meldung: LOG(0). −Inf als Ergebnis, falls TRAP(DVZ) = 2. −MAXREAL als Ergebnis mit INX gesetzt, falls TRAP(DVZ) = 1.
1	0; keine Ausnahmeflags gesetzt.
☆	INX gesetzt.

Gammafunktion (GAMMA(*X*))

Argument	Ergebnis
Inf	Inf; keine Ausnahmeflags gesetzt.
±0	DVZ gesetzt; Meldung: GAMMA=INF. CLASS(*X*) × Inf als Ergebnis, falls TRAP(DVZ) = 2. CLASS(*X*) × MAXREAL als Ergebnis mit INX gesetzt, falls TRAP(DVZ) = 1.
$X < 0$ und ganzzahlig	DVZ gesetzt; Meldung: GAMMA=INF. −Inf als Ergebnis, falls TRAP(DVZ) = 2. −MAXREAL als Ergebnis und INX gesetzt, falls TRAP(DVZ) = 1.
☆	INX wird für alle *X*, die nicht in {1, 2, ..., 18} enthalten sind, gesetzt; UNF, OVF gesetzt, falls nötig.

Nächste Maschinenzahl (NEIGHBOR(*X*, *Y*))

Argumente		Ergebnis
X	*Y*	
X = *Y*	*X* = *Y*	*X*; UNF, INX gesetzt, falls TRAP(UNF) ≠ 2 und 0 < |*X*| < EPS.
MAXREAL	Inf	Inf; keine Ausnahmeflags gesetzt.
−MAXREAL	−Inf	−Inf; keine Ausnahmeflags gesetzt.
±Inf	★	SGN(*X*) × MAXREAL; keine Ausnahmeflags gesetzt.
±0	★	SGN(*Y*) × MINREAL; UNF, INX gesetzt, falls TRAP(UNF) ≠ 2.
MINREAL	±0	0; keine Ausnahmeflags gesetzt.
−MINREAL	±0	−0; keine Ausnahmeflags gesetzt.
★	★	UNF, INX gesetzt, falls |NEIGHBOR(X,Y)| < EPS und TRAP(UNF) ≠ 2.

Skalierung mit Zehnerpotenzen (SCALE10(*X*, *N*))

Argumente		Ergebnis
X	*N*	
★	nicht ganzzahlig	IVL gesetzt; NaN als Ergebnis; Meldung: Invalid Arg.
±Inf	−Inf	IVL gesetzt; NaN als Ergebnis; Meldung: Inf*0.
0	Inf	IVL gesetzt; NaN als Ergebnis; Meldung: Inf*0.
±Inf	★	±Inf; keine Ausnahmeflags gesetzt.
★	−Inf	SGN(*X*) × 0; keine Ausnahmeflags gesetzt.
★	Inf	SGN(*X*) × Inf; keine Ausnahmeflags gesetzt.
★	★	INX, OVF, UNF gesetzt, falls nötig.

Komplexe Funktionen und Operationen

Komplexe Funktionen und Operationen werden in Abschnitt 5 dieses Handbuchs behandelt. Nachfolgend wird für die auf komplexe Argumente erweiterten Funktionen des HP-71 und des Mathematik-Pakets (+, −, *, /, ^, LOG, EXP, SIN, COS, TAN, SINH, COSH, TANH, SQRT, SGN, ABS, =, <, >, ?, and #) nur der komplexe Fall diskutiert. Die Berechnung der Funktionen POLAR, RECT, ARG und PROJ mit einem reellen Argument X entspricht der Berechnung der Funktionen mit dem komplexen Argument $(X, 0)$.

Ein aktives NaN als Argument (NaN kann sowohl im Real- als auch im Imaginärteil eines komplexen Arguments stehen) setzt den Flag IVL und gibt die Meldung Signaled Op zurück; bei TRAP(IVL) = 2 das ursprünglich aktive NaN zu einem passiven NaN und die Operation kann fortgesetzt werden. Im folgenden wird nur auf passive NaN's Bezug genommen.

Die nachstehenden Begriffe werden im folgenden verwendet:

- *Komplex* bezeichnet Daten vom Typ COMPLEX oder COMPLEX SHORT.
- *Reell* bezeichnet Daten vom Typ REAL, SHORT oder INTEGER.
- CNaN ist eine beliebige komplexe Zahl, die mindestens in einer Komponente NaN enthält.
- CInf ist eine beliebige komplexe Zahl mit Betrag Inf; d.h. mindestens eine Komponente der Zahl ist ±Inf.
- *CZERO* ist eine beliebige komplexe Zahl mit Betrag 0.
- *Arg(Z)* bezeichnet das Argument von *Z*, d.h. *Arg(Z)* ist das mathematisch exakte Äquivalent der Funktion ARG(Z) des Mathematik-Pakets.
- $|Z|$ bezeichnet den Betrag von *Z*.
- Die komplexen Variablen *Z* und *W* werden auch als (x, y) und (u, v) angegeben.

+, − (Addition und Subtraktion)

Für reelles *a* und komplexes *Z* gilt $a \pm Z = (a \pm x, \pm y)$. Ist sowohl *Z* als auch *W* komplex, gilt $Z \pm W = (x \pm u, y \pm v)$. Der Flag IVL wird gesetzt und die Meldung Inf-Inf angezeigt, wenn eine der Komponenten durch Inf − Inf berechnet wird; der entsprechenden Komponente wird NaN zugewiesen. Ansonsten wird in Abhängigkeit von der Art der Fehlerbedingung für jede Ergebniskomponente der Flag INX, OVF oder UNF gesetzt.

* (Multiplikation)

Für reelles *a* und komplexes *Z* gilt $a \times Z = Z \times a = (ax, ay)$. Der Flag IVL wird gesetzt und die Meldung Inf*0 angezeigt, wenn eine der Komponenten durch die Multiplikation $(\pm \text{Inf}) \times (\pm 0)$ berechnet wird; der entsprechenden Komponente wird NaN zugewiesen. Ansonsten wird in Abhängigkeit von der Art der Fehlerbedingung für jede Ergebniskomponente der Flag INX, OVF oder UNF gesetzt.

Ist sowohl Z als auch W komplex, dann ist $Z \times W$ über die nachstehende Tabelle gegeben.

Multiplikation zweier komplexer Werte ($Z * W$)

Argumente		Ergebnis
Z	*W*	
CNaN	☆	(NaN, NaN); keine Ausnahmeflags gesetzt.
☆	CNaN	(NaN, NaN); keine Ausnahmeflags gesetzt.
CInf	CZERO	IVL gesetzt; (NaN, NaN) als Ergebnis; Meldung: Inf*0.
CZERO	CInf	IVL gesetzt; (NaN, NaN) als Ergebnis; Meldung: Inf*0.
CInf	☆	RECT(Inf, Arg(*Z*) + Arg(*W*)); keine Ausnahmeflags gesetzt.
☆	CInf	RECT(Inf, Arg(*Z*) + Arg(*W*)); keine Ausnahmeflags gesetzt.
☆	☆	(*xu* − *yv*, *xv* + *yu*); gegebenenfalls wird für jede Ergebniskomponente INX, OVF, UNF gesetzt.

/ (Division)

Für reelles a und komplexes Z gilt $Z/a = (x/a, y/a)$. Der Flag IVL wird gesetzt und die Meldung 0/0 angezeigt, wenn eine der Komponenten durch die Division $(\pm 0)/(\pm 0)$ berechnet wird; in die entsprechende Ergebniskomponente wird NaN eingetragen. Der Flag IVL wird gesetzt und die Meldung Inf/Inf angezeigt, wenn eine der Komponenten durch die Division $(\pm \text{Inf}/\pm \text{Inf})$ berechnet wird; in die entsprechende Ergebniskomponente wird NaN eingetragen. Der Flag DVZ wird gesetzt und die Meldung /Zero angezeigt, wenn eine der Komponenten durch die Division $T/(\pm 0)$ berechnet wird, wobei T weder NaN noch $\pm$Inf oder ± 0 sein darf; in die entsprechende Komponente wird Inf mit dem entsprechenden Vorzeichen als Ergebnis eingetragen, falls TRAP(DVZ) = 2 gesetzt ist; wenn TRAP(DVZ) = 1 gesetzt ist, dann wird MAXREAL mit dem entsprechenden Vorzeichen in die entsprechende Komponente eingetragen und der Flag INX gesetzt. Ansonsten wird in Abhängigkeit von der Fehlerbedingung für jede Ergebniskomponente der Flag INX, OVF oder UNF gesetzt.

Für komplexes Z gelten die folgenden Definitionen: Für Z = CZERO ist $1/Z$ als (CLASS(x) × Inf, −SGN(y)) definiert. Für Z = CInf ist $1/Z$ als (SGN(x) × 0, −SGN(y) × 0) festgelegt.

a/Z ist bei reellem a und komplexem Z über die nachstehende Tabelle definiert.

Division einer reellen Zahl durch eine komplexe Zahl (*a* / *Z*)

Argumente		Ergebnis
a	**Z**	
NaN	☆	(NaN,NaN); keine Ausnahmeflags gesetzt
☆	CNaN	(NaN,NaN); keine Ausnahmeflags gesetzt
±Inf	CInf	IVL gesetzt; (NaN,NaN) als Ergebnis; Meldung: Inf/Inf
±0	CZERO	IVL gesetzt; (NaN,NaN) als Ergebnis; Meldung: 0/0.
±Inf	CZERO	SGN(*a*) × (1/*Z*) (Multiplikation reell × komplex); keine Ausnahmeflags gesetzt
☆	CZERO	DVZ gesetzt; Meldung: /Zero. *a* × (1/*Z*) (Multiplikation reell × komplex); ein Ergebnis wird nur bei TRAP(DVZ) = 2 zurückgegeben. *a* × (1/*Z*) (Multiplikation reell × komplex); bei TRAP(DVZ) = 1 wird ±Inf in einer Ergebniskomponente durch ±MAXREAL ersetzt und der Flag INX gesetzt.
☆	CInf	*a* × (1/*Z*) (Multiplikation reell × komplex); keine Ausnahmeflags gesetzt
±Inf	☆	*a* × CONJ(*Z*) (Multiplikation reell × komplex); IVL wird gesetzt und die Meldung Inf*0 angezeigt, wenn eine der Komponenten durch die Multiplikation (±Inf) × (±0) berechnet wird; in die entsprechende Ergebniskomponente wird NaN eingetragen. In allen anderen Fällen werden keine Ausnahmeflags gesetzt.
☆	☆	($a/|Z|^2$) × CONJ(*Z*) (Multiplikation reell × komplex); INX, OVF, UNF werden in Abhängigkeit von der Fehlerbedingung für jede Ergebniskomponente gesetzt.

Wenn sowohl Z als auch W komplex ist, dann ist W/Z über die nachstehende Tabelle gegeben.

Division zweier komplexer Zahlen (W/Z)

Argumente		Ergebnis
W	*Z*	
CNaN	⋆	(NaN,NaN); keine Ausnahmeflags gesetzt.
⋆	CNaN	(NaN,NaN); keine Ausnahmeflags gesetzt.
CZERO	CZERO	IVL gesetzt; (NaN,NaN) als Ergebnis; Meldung: 0/0.
CInf	CInf	IVL gesetzt; (NaN,NaN) als Ergebnis; Meldung: Inf/Inf.
CInf	CZERO	$W \times (1/Z)$ (Multiplikation komplex × komplex); keine Ausnahmeflags gesetzt.
⋆	CZERO	DVZ gesetzt; Meldung: /Zero $W \times (1/Z)$ (Multiplikation komplex × komplex); ein Ergebnis wird nur bei TRAP(DVZ) = 2 zurückgegeben. $W \times (1/Z)$ (Multiplikation komplex × komplex); wenn TRAP(DVZ) = 1, dann wird ±Inf in einer Ergebniskomponente durch ±MAXREAL ersetzt und INX gesetzt.
⋆	CInf	$W \times (1/Z)$ (Multiplikation komplex × komplex); keine Ausnahmeflags gesetzt.
⋆	⋆	$(W \times \text{CONJ}(Z)) / \lvert Z \rvert^2$ (Multiplikation komplex × komplex und Division komplex/reell); gegebenenfalls wird für jede Ergebniskomponente INX, OVF oder UNF gesetzt.

Die nachstehenden Tabellen definieren für die angegebenen Funktionen den Funktionswert $f(Z)$ für komplexe Argumente Z.

Komplexer Sinus (SIN(*Z*))

Argument	Ergebnis
CNaN	(NaN,NaN); keine Ausnahmeflags gesetzt.
(±Inf, ⋆)	IVL gesetzt; (NaN,NaN) als Ergebnis; Meldung: Invalid Arg.
(⋆, ±Inf)	RECT((Inf, Arg((sin(*x*), SGN(*y*)cos(*x*))))); keine Ausnahmeflags gesetzt.
⋆	Gegebenenfalls wird für jede Ergebniskomponente INX, OVF oder UNF gesetzt.

Komplexer Sinus Hyperbolicus (SINH(*Z*))

Argument	Ergebnis
CNaN	(NaN,NaN); keine Ausnahmeflags gesetzt.
(⋆, ±Inf)	IVL gesetzt; (NaN,NaN) als Ergebnis; Meldung: Invalid Arg.
(±Inf, ⋆)	RECT((Inf, Arg((SGN(*x*)cos(*y*), sin(*y*))))); keine Ausnahmeflags gesetzt.
⋆	Gegebenenfalls wird für jede Ergebniskomponente INX, OVF oder UNF gesetzt.

Komplexer Cosinus (COS(*Z*))

Argument	Ergebnis
CNaN	(NaN,NaN); keine Ausnahmeflags gesetzt.
(±Inf, ⋆)	IVL gesetzt; (NaN,NaN) als Ergebnis; Meldung: Invalid Arg.
(⋆, ±Inf)	RECT((Inf, Arg((cos(*x*), −SGN(*y*)sin(*x*))))); keine Ausnahmeflags gesetzt.
⋆	Gegebenenfalls wird für jede Ergebniskomponente INX, OVF oder UNF gesetzt.

Komplexer Cosinus Hyperbolicus (COSH(*Z*))

Argument	Ergebnis
CNaN	(NaN,NaN); keine Ausnahmeflags gesetzt.
(⋆, ±Inf)	IVL gesetzt; (NaN,NaN) als Ergebnis; Meldung: Invalid Arg.
(±Inf, ⋆)	RECT((Inf, Arg((cos(*y*), SGN(*x*)sin(*y*))))); keine Ausnahmeflags gesetzt.
⋆	Gegebenenfalls wird für jede Ergebniskomponente INX, OVF oder UNF gesetzt.

Komplexer Tangens (TAN(*Z*))

Argument	Ergebnis
CNaN	(NaN,NaN); keine Ausnahmeflags gesetzt.
(±Inf, ±Inf)	(0, SGN(*y*)); keine Ausnahmeflags gesetzt.
(±Inf, ⋆)	IVL gesetzt; (NaN,NaN) als Ergebnis; Meldung: Invalid Arg.
(⋆, ±Inf)	(SGN(sin(*x*) cos(*x*))*0, SGN(*y*)); keine Ausnahmeflags gesetzt.
⋆	Gegebenenfalls wird für jede Ergebniskomponente INX, OVF oder UNF gesetzt.

Komplexer Tangens Hyperbolicus (TANH(*Z*))

Argument	Ergebnis
CNaN	(NaN,NaN); keine Ausnahmeflags gesetzt.
(±Inf, ±Inf)	(SGN(*x*),−0); keine Ausnahmeflags gesetzt.
(⋆, ±Inf)	IVL gesetzt; (NaN,NaN) als Ergebnis; Meldung: Invalid Arg.
(±Inf, ⋆)	(SGN(*x*),SGN(sin(*y*) cos(*y*))*0); keine Ausnahmeflags gesetzt.
⋆	Gegebenenfalls wird für jede Ergebniskomponente INX, OVF oder UNF gesetzt.

Betrag einer komplexen Zahl (ABS(Z))

Argument	Ergebnis
CNaN	NaN; keine Ausnahmeflags gesetzt.
CInf	Inf; keine Ausnahmeflags gesetzt.
☆	INX, OVF, UNF gesetzt, falls nötig.

Argument (ARG(Z))

Argument	Ergebnis
CNaN	NaN; keine Ausnahmeflags gesetzt.
(Inf,Inf)	45 Grad bzw. $\pi/4$ Radiant; INX gesetzt, falls im Radiant-Modus.
(-Inf,Inf)	135 Grad bzw. $3\pi/4$ Radiant; INX gesetzt, falls im Radiant-Modus.
(Inf,-Inf)	-45 Grad bzw. $-\pi/4$ Radiant; INX gesetzt, falls im Radiant-Modus.
(-Inf,-Inf)	-135 Grad bzw. $-3\pi/4$ Radiant; INX gesetzt, falls im Radiant-Modus.
☆	ANGLE(*x*,*y*); INX oder UNF gesetzt, falls nötig.

Projektion auf ∞ (PROJ(Z))

Argument	Ergebnis
CNaN	(NaN,NaN); keine Ausnahmeflags gesetzt.
CInf	(Inf,0); keine Ausnahmeflags gesetzt.
☆	*Z*; für jede Komponente, deren Betrag zwischen 0 und EPS liegt wird UNF und INX gesetzt, falls TRAP(UNF) ≠ 2.

Einheitsvektor (SGN(Z))

Argument	Ergebnis
CNaN	(NaN,NaN); keine Ausnahmeflags gesetzt.
CZERO	*Z*; keine Ausnahmeflags gesetzt.
(±Inf, ±Inf)	RECT((1, Arg(*Z*))); INX gesetzt.
(±Inf, ☆)	(SGN(*x*),SGN(*y*)*0); keine Ausnahmeflags gesetzt.
(☆, ±Inf)	(SGN(*x*)*0,SGN(*y*)); keine Ausnahmeflags gesetzt.
☆	Gegebenenfalls wird für jede Ergebniskomponente INX oder UNF gesetzt.

Quadratwurzel (SQRT(*Z*))

Argument	Ergebnis
CNaN	(NaN,NaN); keine Ausnahmeflags gesetzt.
CInf	RECT((Inf, Arg(*Z*)/2)); keine Ausnahmeflags gesetzt.
✱	Gegebenenfalls wird für jede Ergebniskomponente INX oder UNF gesetzt.

Rechtecks/Polarumwandlung (POLAR(*Z*))

Argument	Ergebnis
✱	(ABS(*Z*),ARG(*Z*)); gegebenenfalls wird für jede Ergebniskomponente INX, OVF oder UNF gesetzt.

Polar/Rechtecksumwandlung (RECT(*Z*))

Argument	Ergebnis
CNaN	(NaN,NaN); keine Ausnahmeflags gesetzt.
(±Inf, ±Inf)	(SGN(*x*)* Inf,0); keine Ausnahmeflags gesetzt.
(±0, ±Inf)	(*x*, *x*); keine Ausnahmeflags gesetzt.
(✱, ±Inf)	IVL gesetzt; (NaN,NaN) als Ergebnis; Meldung: Invalid Arg.
(±Inf, ✱)	(a cos(*y*),b sin(*y*)); keine Ausnahmeflags gesetzt; $a = \begin{cases} x \text{ falls } \cos(y) \neq 0 \\ \text{SGN}(x) \text{ falls } \cos(y) = 0 \end{cases}$ und $b = \begin{cases} x \text{ falls } \sin(y) \neq 0 \\ \text{SGN}(x) \text{ falls } \sin(y) = 0 \end{cases}$
✱	(*x* cos(*y*),*x* sin(*y*)); gegebenenfalls wird für jede Ergebniskomponente INX oder UNF gesetzt.

Natürlicher Logarithmus (LOG(Z))

Argument	Ergebnis
CNaN	(NaN,NaN); keine Ausnahmeflags gesetzt.
CZERO	DVZ gesetzt; Meldung: LOG(0). (-Inf, ARG(Z)) als Ergebnis, falls TRAP(DVZ) = 2. (-MAXREAL,ARG(Z)) als Ergebnis und INX gesetzt, falls TRAP(DVZ) = 1.
CInf	(Inf,ARG(Z)); gegebenenfalls wird für den Imaginärteil des Ergebnisses INX oder UNF gesetzt.
☆	Gegebenenfalls wird für jede Ergebniskomponente INX oder UNF gesetzt.

Exponentiation (EXP(Z))

Argument	Ergebnis
CNaN	(NaN,NaN); keine Ausnahmeflags gesetzt.
(-Inf, ±Inf)	(0,0); keine Ausnahmeflags gesetzt.
(Inf, ±Inf)	(Inf,0); keine Ausnahmeflags gesetzt.
(☆, ±Inf)	IVL gesetzt; (NaN,NaN) als Ergebnis; Meldung: Invalid Arg.
(-Inf, ☆)	(0 × cos(*y*),0 × sin(*y*)); INX wird für jede Ergebniskomponente gesetzt, falls nötig.
(Inf, ☆)	RECT(Z); keine Ausnahmeflags gesetzt.
☆	Gegebenenfalls wird für jede Ergebniskomponente INX, OVF oder UNF gesetzt.

Verhältnisoperatoren

Jede numerische Vergleichsoperation mit komplexen Operanden, die die Operatoren < oder > ohne ? bzw. # enthält, bedingt das Setzen des IVL-Flags und die Anzeige der Meldung Unordered. Wenn TRAP(IVL) auf 2 gesetzt ist, dann wird 0 oder 1 nur dann zurückgegeben, wenn der Vergleichsoperator = vorhanden ist. D.h. $Z <= W$, $Z >= W$ und $Z <> W$ sind nur dann wahr, wenn $x = u$ und $y = v$ ist; $Z < W$, $Z > W$ und $Z <> W$ sind immer falsch.

^ (Exponentiation)

Vor der Berechnung von $W \hat{} Z$ werden die nachstehenden Vorbereitungsmaßnahmen getroffen:

1. Ein reelles W oder Z wird für die Berechnung in eine komplexe Zahl mit Imaginärteil 0 umgewandelt.
2. Wenn die Variable W bzw. Z den Wert CNaN enthält, wird das Ergebnis (NaN,NaN) zurückgegeben und kein Ausnahmeflag gesetzt.
3. W und Z werden anschließend zur Berechnung in eine kanonische Darstellung umgeformt. Dies geschieht wie folgt: Wenn eine Komponente der komplexen Zahl ±Inf, die andere Komponente jedoch endlich ist, wird der endliche Teil in der kanonischen Form durch ±0 ersetzt (d.h. das Vorzeichen wird erhalten). In allen anderen Fällen wird die vorliegende Form der komplexen Zahl als die kanonische Form betrachtet. (0,Inf) ist beispielsweise die kanonische Form von (6.7,Inf); (-Inf,-0) ist die kanonische Form von (-Inf,-MAXREAL). Im folgenden wird unterstellt, daß W und Z bereits in der kanonischen Darstellung vorliegen.

$W \hat{} Z$ wird für W = CZERO durch die nachstehende Tabelle gegeben.

Exponentiation ($W \hat{} Z$): W = CZERO

Argument Z	Ergebnis
$x > 0$	(SGN(*u*^*x*),0); keine Ausnahmeflags gesetzt.
$x < 0$	DVZ gesetzt; Meldung: 0^Neg. (SGN(*u*^*x*)*Inf,0) als Ergebnis, falls TRAP(DVZ) = 2. (SGN(*u*^*x*)*MAXREAL,0) als Ergebnis und INX gesetzt, falls TRAP(DVZ) = 1.
$x = 0$ und $y = 0$	Keine Ausnahmeflags gesetzt; Meldung: 0^0; als Ergebnis wird der Vorgabewert (1,0) zurückgegeben, sofern TRAP(IVL) ≠ 0.
$x = 0$ und $y \neq 0$	IVL gesetzt; (NaN,NaN) als Ergebnis; Meldung: Invalid Arg.

$W \hat{} Z$ wird für $y \neq 0$ durch die nachstehende Tabelle gegeben.

Exponentiation ($W \hat{} Z$): $y \neq 0$

Argumente		Ergebnis
W	Z	
(1, ±0)	CInf	IVL gesetzt; (NaN,NaN) als Ergebnis; Meldung: 1^Inf.
✭	✭	EXP(Z*LOG(W)) (Multiplikation komplex × komplex). Wenn Z*LOG(W) gleich (±0, ±Inf) ist, dann ist dieser Wert nicht im Definitionsbereich von EXP enthalten und IVL wird gesetzt, (NaN,NaN) zurückgegeben und die Meldung Invalid Arg angezeigt. Ansonsten wird gegebenenfalls für jede Ergebniskomponente INX, OVF oder UNF gesetzt.

$W \hat{} Z$ wird für $y = 0$ und $v \neq 0$ durch die nachstehende Tabelle definiert.

Exponentiation ($W \hat{} Z$): $y = 0$ and $v \neq 0$

Argumente		Ergebnis
W	***Z***	
$\|W\| = 1$	CInf	IVL gesetzt; (NaN,NaN) als Ergebnis; Meldung: Invalid Arg.
CInf	CZERO	Keine Ausnahmeflags gesetzt; Anzeige der Meldung Inf^0; falls TRAP(IVL) ≠ 0 wird der Vorgabewert (1,0) als Ergebnis zurückgegeben.
☆	☆	EXP(*x**LOG(*W*)) (Multiplikation reell × komplex); gegebenenfalls wird für jede Ergebniskomponente INX, OVF oder UNF gesetzt.

$W \hat{} Z$ wird für $y = 0$ und $v = 0$ durch die nachstehende Tabelle definiert.

Exponentiation ($W \hat{} Z$): $y = 0$ und $v \neq 0$

Argumente		Ergebnis
W	***Z***	
$u = \pm$Inf	$x = 0$	Keine Ausnahmeflags gesetzt; Anzeige der Meldung Inf^0; falls TRAP(IVL) ≠ 0 wird der Vorgabewert (1,0) als Ergebnis zurückgegeben.
u = ±1	CInf	IVL gesetzt; (NaN,NaN) als Ergebnis; Meldung: 1^Inf.
☆	CInf	(|*u*|^*x*,0); keine Ausnahmeflags gesetzt.
☆	☆	EXP(*x**LOG(*W*)) (Multiplikation reell × komplex); gegebenenfalls wird für jede Ergebniskomponente INX, OVF oder UNF gesetzt.

Matrizenfunktionen und -operationen

Matrizenfunktionen und -operationen werden in den Abschnitten 7, 8 und 9 dieses Handbuchs beschrieben. Die im letzten Teilabschnitt eingeführten Definitionen für CZERO, CInf, komplex, usw. werden auch in diesem Absatz verwendet.

CNORM(A), RNORM(A)

Wenn **A** eine $M \times N$ Matrix ist (bei Vektoren ist $N = 1$), dann gilt:

$$\texttt{CNORM(}\mathbf{A}\texttt{)} = \max_{1 \leqslant j \leqslant N} \sum_{i=1}^{M} |a_{ij}| \qquad \texttt{RNORM(}\mathbf{A}\texttt{)} = \max_{1 \leqslant j \leqslant M} \sum_{j=1}^{N} |a_{ij}|$$

Wenn der Real- oder Imaginärteil eines Elements von **A** ein aktives NaN enthält, so bedingt dies ein Setzen des Flags IVL und die Anzeige der Meldung Signaled Op. Bei TRAP(IVL) = 2 wird als Ergebnis ein passives NaN zurückgegeben, und es werden keine weiteren Elemente verarbeitet.

Wenn der Real- oder Imaginärteil eines Elements von **A** eine passives NaN enthält, so bedingt dies ein Setzen des IVL-Flags und die Anzeige der Meldung Unordered; als Ergebnis wird NaN zurückgegeben. In allen anderen Fällen wird gegebenenfalls INX, OVF oder UNF gesetzt.

FNORM(A)

Wenn **A** eine $M \times N$ Matrix ist (bei Vektoren ist $N = 1$), dann gilt:

$$\texttt{FNORM(}\mathbf{A}\texttt{)} = \left(\sum_{i=1}^{N} \sum_{j=1}^{N} |a_{ij}|^2 \right)$$

Wenn der Real- oder Imaginärteil eines Elements von **A** ein aktives NaN enthält, so bedingt dies ein Setzen des IVL-Flags und die Anzeige der Meldung Signaled Op. Bei TRAP(IVL) = 2 wird als Ergebnis ein passives NaN zurückgegeben, und es werden keine weiteren Elemente verarbeitet.

Passive NaN's werden weitergereicht, ohne daß ein Ausnahmeflag gesetzt wird. In allen anderen Fällen wird gegebenenfalls INX, OVF oder UNF gesetzt.

DOT(A,B)

Wenn sowohl **A** als auch **B** ein N-elementiger Vektor ist, dann gilt:

$$\texttt{DOT(}\mathbf{A}\texttt{,}\mathbf{B}\texttt{)} = \sum_{i=1}^{N} \overline{a_i} b_i$$

(Wenn einer der beiden Vektoren komplex ist, gelten die zuvor gegebenen Definitionen für komplexe Addition und Multiplikation.) Wenn der Real- oder Imaginärteil eines Elements von **A** oder **B** ein aktives NaN enthält, so bedingt dies ein Setzen des IVL-Flags und die Anzeige der Meldung Signaled Op. Wenn bei der Berechnung des obigen Ausdrucks ± 0 oder CZERO mit $\pm$Inf oder CInf multipliziert wird, dann wird der IVL-Flag ebenfalls gesetzt und die Meldung Inf*0 angezeigt. Schließlich wird der IVL-Flag gesetzt und die Meldung Inf-Inf angezeigt, wenn in dem obigen Ausdruck eine Inf − Inf entsprechende Addition durchgeführt wird.

Wenn nur eine den IVL-Flag setzende Ausnahmebedingung auftritt, dann wird die der Ausnahme entsprechende Meldung angezeigt. Bei Auftreten von mehrereren IVL-Ausnahmebedingungen hängt die angezeigte Meldung von der Reihenfolge des Auftretens und vom Typ der Ausnahmebedingungen ab. Bei TRAP(IVL) = 2 wird als Ergebnis im reellen Fall NaN oder im komplexen Fall ein komplexer Wert mit einer oder zwei NaN-Komponenten zurückgegeben. Passive NaN's werden weitergereicht, ohne daß Ausnahmeflags gesetzt werden. In allen anderen Fällen gegebenenfalls INX, OVF oder UNF gesetzt.

MAT C=A*B

Wenn **A** eine $M \times N$ Matrix und **B** eine $N \times P$ Matrix ist (bei Vektoren ist $P=1$), dann gilt:

$$c_{ij} = \sum_{k=1}^{N} a_{ik} b_{ij}$$

(Wenn einer der beiden Vektoren komplex ist, gelten die zuvor gegebenen Definitionen für komplexe Addition und Multiplikation.) Da sich jedes Ergebniselement aus einem Punktprodukt berechnet, entspricht die hier geltende Ausnahmebehandlung derjenigen bei der Anwendung von DOT(**A**,**B**) für jedes einzelne Ergebniselement.

MAT C=TRN(A)*B

Wenn **A** eine $M \times N$ Matrix und **B** eine $M \times P$ Matrix ist (bei Vektoren ist $P=1$), dann gilt:

$$c_{ij} = \sum_{k=1}^{M} \overline{a_{ki}} b_{kj}$$

(Wenn entweder **A** oder **B** komplex ist, gelten die zuvor gegebenen Definitionen für komplexe Addition und Multiplikation.)

Da jedes Ergebniselement über ein Punktprodukt berechnet wird, entspricht die hier geltende Ausnahmebehandlung derjenigen bei der Anwendung von DOT(**A**,**B**) für jedes einzelne Ergebniselement.

MAT C=A±B

Alle Elemente von **C** werden einzeln über

$$c_{ij} = a_{ij} \pm b_{ij}$$

berechnet. (Wenn entweder **A** oder **B** komplex ist, dann gelten die zuvor gegebenen Definitionen für komplexe Addition und Multiplikation.)

Wenn ein Element von **A** oder **B** (oder bei komplexen Matrizen **A** oder **B** der Real- oder Imaginärteil eines Elements) ein aktives NaN enthält, wird der IVL-Flag gesetzt und die Meldung Signaled Op angezeigt. Bei TRAP(IVL) = 2 wird das ursprünglich aktive NaN in einem Element oder einer Elementkomponente zu einem passiven NaN und die Operation wird fortgesetzt. Passive NaN's werden weitergereicht, ohne daß Ausnahmeflags gesetzt werden.

Der IVL-Flag wird gesetzt und die Meldung Inf-Inf angezeigt, wenn eine Addition oder Subtraktion (oder eine komponentenweise Addition oder Subtraktion) Inf − Inf entspricht; NaN wird in das entsprechende Ergebniselement oder in die entsprechende Ergebniskomponente eingetragen. In allen anderen Fällen wird für das entsprechende Ergebniselement oder die entsprechende Ergebniskomponente gegebenenfalls INX, OVF oder UNF gesetzt.

MAT B=(s)*A

Alle Elemente von **B** werden einzeln über

$$b_{ij} = sa_{ij}$$

berechnet. (Wenn entweder s oder **A** komplex ist, gilt die zuvor gegebene Definition für komplexe Multiplikationen.) Wenn s (oder bei komplexem s der Real- oder Imaginärteil) ein aktives NaN enthält, wird der IVL-Flag gesetzt und die Meldung Signaled Op angezeigt; entsprechendes gilt, wenn ein Element von **A** (bzw. bei komplexem **A** der Real- oder Imaginärteil eines Elements) ein aktives NaN enthält. In beiden Fällen werden bei TRAP(IVL) = 2 ursprünglich aktive NaN's zu passiven NaN's, und die Operation wird fortgesetzt. Passive NaN's werden weitergereicht, ohne daß Ausnahmeflags gesetzt werden.

Wenn bei der Berechnung eines Ergebniselements ±0 oder CZERO mit ±Inf oder CInf multipliziert wird, dann wird der IVL-Flag gesetzt und die Meldung Inf*0 angezeigt. Das entsprechende Ergebniselement ist bei TRAP(IVL) = 2 entweder der Wert NaN oder ein komplexer Wert mit einer oder zwei NaN-Komponenten. In allen anderen Fällen wird für das entsprechende Ergebniselement oder die entsprechende Ergebniskomponente gegebenenfalls INX, OVF oder UNF gesetzt.

DET(A), MAT C=INV(A), MAT C=SYS(A,B)

Die Ausnahmebehandlung dieser drei Operationen ist wegen deren komplizierten Grundalgorithmen äußerst schwierig. Daher wird hier nur eine Zusammenfassung gegeben.

Wenn ein Element von **A** oder **B** (oder bei komplexen Matrixelementen der Real- oder der Imaginärteil) ein aktives NaN enthält, wird der IVL-Flag gesetzt und die Meldung Signaled Op angezeigt. Bei TRAP(IVL) = 2 wird das ursprünglich aktive NaN in einem Element oder einer Elementkomponente zu einem passiven NaN und die Operation wird vorgesetzt.

Gegebenenfalls wird für jedes Ergebniselement OVF, UNF oder INX gesetzt. Diese Flags können auch während des Ablaufs der Berechnung gesetzt werden. (OVL wird speziell bei Auftreten einer (maschinen-) singulären Matrix **A** gesetzt.) Ebenso kann der IVL-Flag gesetzt werden, wobei die entsprechenden Meldungen (Inf*0, Inf-Inf und/oder Inf/Inf) angezeigt werden. Diese Meldungen werden nur bei Auftreten von ±Inf in **A** oder **B** oder bei einer Bereichsüberschreitung an einem Zwischenschritt angezeigt. Im letzteren Fall kann das Anzeigen einer Meldung durch Setzen von TRAP(OVF) = 1 vor Beginn der Berechnung unterdrückt werden.

Weitere Funktionen des Mathematik-Pakets

PROOT

Bei der Ausführung von PROOT untersucht der Algorithmus das Koeffizientenfeld zunächst auf das Auftreten der Werte NaN und Inf sowie auf führende und nachlaufende Nullen.

Zuerst wird auf NaN's abgeprüft. Wenn nur ein Element des Koeffizientenfelds den Wert NaN enthält, wird jedem Element des Ergebnisfelds der Wert (NaN,NaN) zugewiesen, und die Ausführung von PROOT wird beendet, ohne daß Ausnahmeflags gesetzt werden. (Insbesondere setzen Koeffizienten mit einem aktiven NaN nicht den IVL-Flag.)

Der als nächstes untersuchte Sonderfall ist das Auftreten von Inf's im Koeffizientenfeld. Wenn nur einer der Koeffizienten ±Inf ist, werden alle endlichen Koeffizienten auf Null gesetzt, und die Berechnung wird mit der Abfrage auf führende und nachlaufende Nullen fortgesetzt.

Führende Nullen werden als nächstes behandelt. Für jeden führenden Nullkoeffizient wird eine Nullstelle im Punkt (Inf,Inf) im Ergebnisfeld abgelegt, ohne daß dabei Ausnahmeflags gesetzt werden. Anschließend wird der nächste Koeffizient als führender Koeffizient angenommen und der Prozeß von vorne durchlaufen. Bei jedem Ablegen einer Nullstelle im Ergebnisfeld wird der Grad des Polynoms heruntergesetzt, und die Ausführung von PROOT ist beendet, wenn der Grad des Polynoms 0 ist.

Anschließend werden nachlaufende Nullen behandelt. Für jeden nachlaufenden Nullkoeffizienten wird im Ergebnisfeld eine Nullstelle im Punkt (0,0) abgelegt. Ausnahmeflags werden dabei nicht gesetzt. Der vorletzte Koeffizient wird zum letzten Koeffizient und der Prozeß von vorne durchlaufen. Wie bei führenden Nullkoeffizienten wird auch hier für jede im Ergebnisfeld abgelegte Nullstelle der Grad des Polynoms reduziert, und die Ausführung von PROOT ist beendet, sobald der Grad des Polynoms 0 ist.

Wenn alle diese Spezialfälle abgearbeitet sind, ist der Grad des Polynoms positiv und die (verbleibenden) Koeffizienten sind entweder sämtlich endlich oder der erste und der letzte Koeffizent

des (Rest-) Polynoms sind beide ±Inf. Im ersten Fall werden die Nullstellen des (Rest-) Polynoms berechnet. Im zweiten Fall sind mindestens zwei der Koeffizienten des ursprünglichen Polynoms ±Inf und eine Zerlegung des Polynoms ist sinnlos; der Algorithmus legt dann *D* Nullstellen im Punkt (NaN,NaN) im Ergebnisfeld ab (wo *D* der Grad des Restpolynoms ist) und beendet die Ausführung von PROOT. Jede dieser Nullstellen bedingt ein Setzen des IVL-Flags und die Anzeige der Meldung Invalid Arg.

Nach Ausschluß der oben genannten Spezialfälle wird gegebenenfalls für jedes Element des Ergebnisfelds OVF oder UNF gesetzt; der Flag INX wird immer gesetzt.

FOUR

Wie bei PROOT werden auch bei FOUR zuerst die Spezialfälle (NaN und Inf in den Komponenten von Datenfeldelementen) behandelt.

Zuerst wird auf NaN's abgeprüft: Wenn sich unter den Komponenten der Datenfeldelemente der Wert NaN befindet, dann wird jedem Element des Ergebnisfelds der Wert (NaN,NaN) zugewiesen. Die Ausführung von FOUR wird beendet, und es werden keine Ausnahmeflags gesetzt. (Der IVL-Flag wird durch aktive NaN-Komponenten nicht gesetzt.)

Anschließend fragt der Algorithmus auf Inf ab. Wenn sich unter den Komponenten der Datenfeldelemente der Wert ±Inf befindet, dann wird jedem Element des Ergebnisfelds der Wert (Inf,Inf) zugewiesen. Die Ausführung von FOUR wird beendet, und es werden keine Ausnahmeflags gesetzt.

Nach Ausschluß der oben genannten Spezialfälle wird gegebenenfalls für jedes Element des Ergebnisfelds OVF oder UNF gesetzt; der Flag INX wird immer gesetzt. Dies trifft nur zu, wenn das Datenfeld ungleich Null ist.

FNROOT und INTEGRAL

Wenn bei der Auswertung der Argumente von INTEGRAL oder FNROOT ein (aktives oder passives) NaN auftritt, wird die Fehlermeldung Invalid Arg angezeigt. Dieser Fehler hält die Ausführung der Operation an. Es wird kein Ausnahmeflag gesetzt.

Allgemein wird jedes ±Inf, das bei der Berechnung eines Arguments von FNRROT oder INTEGRAL auftritt, zur Fortsetzung der Berechnung von FNROOT oder INTEGRAL in den Wert ±MAXREAL umgewandelt. Gegebenenfalls wird für das Ergebnis INX, OVF oder UNF gesetzt.

Sie sollten daran denken, daß FNROOT mit Hilfe des Werts von TRAP(UNF) entscheidet, ob eine Nullstelle im denormalisierten Zahlenbereich gesucht werden soll oder nicht. Dieser Bereich wird nur durchsucht, wenn bei Beginn der Ausführung von FNROOT der Auffangwert TRAP(UNF) auf 2 gesetzt ist.

Schlüsselwortindex

Verkaufsniederlassungen

Hewlett-Packard GmbH:
6000 Frankfurt 56, Bernerstraße 117, Postfach 560140, Tel. (0611) 50 04-1
7030 Böblingen, Herrenbergerstraße 110, Tel. (07031) 14-0
1000 Berlin 30, Keithstraße 2-4, Tel. (030) 24 90 86
4000 Düsseldorf 11, Emanuel-Leutze-Straße 1, Tel. (0211) 59 71-1
2000 Hamburg 60, Kapstadtring 5, Tel. (040) 6 38 04-1
3000 Hannover 91, Heidering 37-39, Tel. (0511) 57 06-0
6800 Mannheim, Roßlauer Weg 2-4, Tel. (0621) 70 05-0
7910 Neu-Ulm, Messerschmittstraße 7, Tel. (0731) 70 24-1
8500 Nürnberg, Neumeyerstraße 90, Tel. (0911) 52 20 83/87
8028 Taufkirchen, Eschenstraße 5, Tel. (089) 61 17-1
7517 Waldbronn 2, Hewlett-Packard-Straße, Tel. (07243) 602-1

Hewlett-Packard (Schweiz) AG:
CH-8967 Widen, Allmend 2, Tel. (057) 31 21 11
CH-4058 Basel, Clarastraße 12, (061) 33 59 20
CH-1217 Meyrin 2, rue du Bois-du-Lan 7, Tel. (022) 83 81 11

Hewlett-Packard Ges.m.b.H für Österreich/sozialistische Staaten:
A-1222 Wien, Lieblgasse 1, Tel. (0222) 23 65 11-0
A-8052 Graz, Grottenhofstraße 94, Tel. (0316) 2 15 66

Hewlett-Packard S.A. Europa-Zentrale:
CH-1217 Meyrin 2, route du Nant d'Avril 150

Serviceniederlassungen

Hewlett-Packard GmbH:
6000 Frankfurt 56, Bernerstraße 117, Postfach 560140, Tel. (0611) 50 04-1

Hewlett-Packard (Schweiz) AG:
CH-8967 Widen, Allmend 2, Tel. (057) 31 21 11

Hewlett-Packard Ges.m.b.H für Österreich/sozialistische Staaten:
A-1222 Wien, Lieblgasse 1, Tel. (0222) 23 65 11-0

HEWLETT PACKARD

Bestellnummer
82480-90003 German

Printed in Singapore 4/84
82480-90012

HP-71
Benutzerdokumentation
Addendum

Dieses Addendum enthält zusätzliche Informationen zur Verwendung von zwei HP-71 Schlüsselworten.

Verwendung von ON . . . GOTO und ON . . . GOSUB

***HP-71 Benutzerhandbuch*, Seite 181.** In den beiden nachstehend beschriebenen Situationen bedingt die Ausführung von ON . . . GOTO und ON . . . GOSUB Anweisungen keine Programmverzweigung. Stattdessen kann jede dieser Anweisungen die Wirkung einer ON . . . RESTORE Anweisung haben. In jedem Fall wird der Speicherinhalt nicht verändert.

Situation 1. Verwenden Sie nicht ON . . . GOTO/GOSUB, solange eine der Rundungseinstellungen OPTION ROUND POS oder OPTION ROUND NEG aktiv ist.

Um dieses Problem zu vermeiden, sollten Sie eine OPTION ROUND NEAR oder OPTION ROUND ZERO Anweisung in jedes Programm einfügen, das ON . . . GOTO/GOSUB Konstruktionen enthält. Stellen Sie sicher, daß bei der Ausführung von ON . . . GOTO/GOSUB die OPTION ROUND NEAR/ZERO Einstellung aktiv ist. Dadurch wird verhindert, daß eine gegebenenfalls aktive OPTION ROUND POS/NEG Einstellung die korrekte Ausführung einer ON . . . GOTO/GOSUB Verzweigung beeinträchtigt; unabhängig davon, ob die OPTION ROUND POS/NEG Einstellung durch eine entsprechende Anweisung in Ihrem Programm erzeugt oder durch den Permanentspeicher erhalten wurde.

Situation 2. Der in einer ON . . . GOTO/GOSUB Anweisung verwendete numerische Ausdruck (ON *numerischer Ausdruck* GOTO/GOSUB) darf keine Operatoren außer +, −, *, / und DIV sowie keine Funktionen (einschließlich benutzerdefinierter Funktionen, trigonometrischer Funktionen, SQRT, usw.) enthalten.

Um dieses Problem zu umgehen, sollte der numerische Ausdruck nur einfache Variablen enthalten und nur die Operatoren +, −, *, / und DIV verwenden. Wenn Sie für den Ausdruck eine Funktion oder einen anderen Operator benötigen, sollten Sie den Wert des Ausdrucks zuvor einer einfachen Variablen zuweisen und dann diese einfache Variable als Pointer in der ON . . . GOTO/GOSUB Anweisung verwenden.

Beispiele: Nachstehend finden Sie Beispiele für den falschen und den richtigen Gebrauch eines Ausdrucks als Pointer in einer ON . . . GOTO/GOSUB Anweisung, wenn der Ausdruck eine Funktion und einen von +, −, *, / oder DIV verschiedenen Operator enthält.

Falsch:

```
ON FNJ(T)^SIN(M) GOSUB 500,600,700
```

Diese ON . . . GOSUB Anweisung bedingt *keine* Programmverzweigung.

Richtig:

```
A = FNJ(T)^SIN(M)
ON A GOSUB 500,600,700
```

Diese `ON...GOSUB` Konstruktion *bedingt* eine Programmverzweigung; die Anweisung wird korrekt ausgeführt.

HEWLETT PACKARD

Portable Computer Division
1000 N.E.Circle Blvd., Corvallis, OR 97330