METHODISCHE FLEXIBILITÄT

Ein Ansatz zur Lösung des Software-Dilemmas in der Statistik

Kurt RODLER*
Karl ZOLLES**

Forschungsbericht/
Research Memorandum No. 238

April 1987

* Assistent am Institut für Ökonometrie und Operations Research der Technischen Universität Wien

** Wissenschaftlicher Mitarbeiter der Abteilung Mathem. Methoden und Computerverfahren am Institut für Höhere Studien, Wien

Die in diesem Forschungsbericht getroffenen Aussagen liegen im Verantwortungsbereich der Autoren und sollen daher nicht als Aussagen des Instituts für Höhere Studien widergegeben werden.

I N H A L T

VORBEMERKUNG

Nach mehr als 20 Jahren der Entwicklung von Standardsoftware in Statistik und verwandten Gebieten muß festgestellt werden, daß sich auch dieses Software-Gebiet in einer Krise befindet. Breite Marktsegmente werden von großen, monolithischen und daher schlecht wartbaren und für den Anwender unveränderbaren Programmsystemen beherrscht, deren Wurzeln in die 60-er Jahre reichen. Als Ansatzpunkt zur Beseitigung der Diskrepanz zwischen Vorstellungen der Benützer auf der einen und der realisierten Software auf der anderen Seite bieten sich Konzepte, die auf eine Flexibilisierung der Systeme hinauslaufen.

Die vorliegende Arbeit beschäftigt sich vorwiegend mit den Aspekten der methodischen Flexibilität, die dem Benützer die benötigten Werkzeuge zur Erweiterung existierender bzw. zur Entwicklung neuer Methoden bereitstellt. Die bereits implementierten Ansätze werden an Hand von fünf statistischen Analysepaketen (TSP, RATS, SAS/IML, S, GAUSS) beschrieben.

ABSTRACT

Standard-Software for statistics and related fields has been developed for more than 20 years. Currently a broad segment of the market is covered by huge and monolithic program systems having their roots in the sixties. These programs are difficult to maintain and there are no facilities for changing and improving them by the user. One possible way to fight this software crisis and to remove the discrepancy between the wishes of the users and the current state of statistical software is the development and implementation of concepts that introduce a new level of flexibility.

This paper mainly deals with the aspects of methodological flexibility and the tools that must be supplied to enable the user to extend existing and develop new methods. The final chapter describes and compares implementations of methodological flexibility in five statistical Packages (TSP, RATS, SAS/IML, S, GAUSS).

1 Einleitung

Die Entwicklung der statistischen Anwendersoftware führte zu hochkomplexen Methodenpaketen, die eine umfassende Bearbeitung von Datensätzen versprechen. Kernpunkt dieser Systeme ist eine Batterie von Methoden, die sich immer durch ihre umfassende, auf neuesten Erkenntnissen der Statistik, der Informatik und der numerischen Mathematik beruhenden Konzeption, ihre einfache, dem täglichen Sprachgebrauch des angewandten Statistikers entnommene Verwendung (Benützerfreundlichkeit) und durch ihre selbsterklärende, die Bedürfnisse des Analytikers widerspiegelnde Ergebnispräsentation auszeichnen.

In der Praxis findet man jedoch eine Unzahl von Stand-Alone-Programmen, deren Existenz, wenn man obigen Versprechungen der Softwareentwickler glaubt, völlig unverständlich ist. Tatsächlich dürften die Ursachen im nicht kompletten Methodenspektrum bzw. dessen langsam bis schleppend voranschreitender Weiterentwicklung sowie in den unzureichenden (Benützer-)Schnittstellen zu suchen sein. Wenn man die Komplexität und den Aufbau der Methodenpakete bedenkt, so ist es unmittelbar einsichtig, warum eine Weiterentwicklung nur langsam und unter beträchtlichem Aufwand durchgeführt werden kann. Insbesondere die Grundkonzeption einer über ein fix vorgegebene Kommando- und Steuerprogramm gekoppelten Prozedurensammlung, realisiert in einer höheren Programmiersprache (meist FORTRAN), verlangt bei jeder Änderung den Abstieg in eben diese Prozedur- und Programmierebene. Dies ist allerdings praktisch Systemspezialisten vorbehalten, da nur diese die nötige Einsicht in die Gesamtstruktur besitzen. Der Anwender hat somit keine Chance in eine Methode und ihre Ein- und Ausgabe einzugreifen.

Für den angewandten Statistiker bedeutet dies:

- längere Wartezeiten bis zur Freigabe neuerer Programmversionen mit neuen und verbesserten Verfahren
- Entwicklung jener oben zitierten Stand-Alone-Programme, um die Lücken zu schließen
- Zuhilfenahme mehrerer Standardpakete - mit allen daraus resultierenden Problemen vor allem beim Datenmanagement - und wahr-

scheinlich eines Taschenrechners, um im Wechsel zwischen den Paketen die gewünschten Resultate zu erhalten
- erhöhten Arbeitsaufwand bei der Auswahl, Umsetzung und Interpretation der Ergebnisse.

Dieser unerfreuliche Zustand wurde bereits wiederholt aufgezeigt (siehe u.a. Chambers, 1980; Chambers, 1982; De Jong, 1984; DeJong, 1986; Molenaar, 1984; Payne/Lane, 1986; Rodler/Hanappi, 1986) und folgernd die Forderung nach mehr Flexibilität in Standard-Statistik-Software erhoben. Einige der oben zitierten Arbeiten beinhalten auch Lösungvorschläge und verweisen auf existierende, flexible Softwareprodukte. So präsentieren Payne/Lane GENSTAT5, Chambers das System S als Lösungen des Flexibilitätsproblems. De Jong regt die Entwicklung von mehrstufiger Statistik-Software mit Implementierungs-, Methoden- und Anwenderebene sowie wohldefinierten Schnittstellen an.

Verstärkt werden die Forderungen auch von Seiten neuer Benützerkreise, die die Personal-Computer-Revolution und die daraus resultierende Unabhängigkeit von Rechenzentren, Systembetreuern etc. zurück zum Computer geführt hat. Ein Teil dieser "neuen" Benützergruppe ist in einem höheren Ausmaß methodenorientiert und verwendet Verfahren, die in den Statistikpaketen (noch) nicht standardmäßig enthalten sind.

Selbstverständlich wurde die Richtigkeit dieser Forderungen und die Notwendigkeit einer erhöhten Flexibilität auch von den Softwareentwicklern erkannt. Das Ergebnis sind einerseits mehr oder weniger gelungene Erweiterungen der existierenden Pakete andererseits Entwicklungen, die die Erweiterbarkeit des Systems durch den Benützer bereits in ihrer Grundkonzeption beinhalten. Einige dieser Realisierungen sollen in dieser Arbeit präsentiert werden.

2 Formen der Flexibilität in statistischer Software

Ausgehend von den obigen Erfahrungen können folgende drei Flexibilitätsbereiche festgestellt werden:

- Eingabe
- Ausgabe
- Statistische Methoden

Sie sollen in diesem Kapitel erläutert und der Ist-Zustand kurz dargestellt werden.

2.1 Flexibilität bei der Eingabe

Im Bereich der Eingabe sind zwei Gesichtspunkte hervorzuheben. Der erste betrifft die Benützerschnittstelle. Eine flexible Benützerschnittstelle ist einerseits an den Wissensstand und die Erfahrung der potentiellen Anwender anpaßbar. Sie sollte also Möglichkeiten bieten zwischen verschiedenen Oberflächen (z.B. Menü für den "casual user" und kurze Kommandosprache für den erfahrenen Benützer) zu wählen und zu wechseln. Andererseits ist eine veränderbare Schnittstelle wünschenswert, damit neue, vom Benützer entwickelte Methodenmodule möglichst nahtlos in die Systemsprache/Systemmenüs integriert werden können. Im Zusammenspiel mit dem ersten Punkt ergibt sich so eine an jeden Wissensstand und alle Erfordernisse adaptierbare Schnittstelle.

Der zweite Aspekt bezieht sich auf die einlesbaren Daten. Hier kann viel Zeit, Arbeit und Mühe eingespart werden, wenn die Dateneingabe weder auf bestimmte Datenstrukturen noch Datenformate eingeschränkt ist. Komplexe Datenstrukturen wie hierarchische oder relationale Datenbestände sollten ebenso verarbeitet werden können wie Datenbestände aus anderen Softwareprodukten (z.B. anderen Methodenpaketen, Datenbanken, div. Spreadsheetprogramme). Und dies möglichst ohne den Umweg über eigene Datenkonversions- und Aufbereitungsprogramme und über Betriebssystemfiles.

2.2 Flexibilität bei der Ausgabe

Auch hinsichtlich der Ausgabe eines Statistikpakets sind zwei Ansatzpunkte erwähnenswert. Der Umfang der Ergebnisse in Quantität und Qualität ist bereits lange Gegenstand von Diskussionen (Molenaar, 1984). Der Benützer sollte hier so triviale Dinge wie das Ausgabeformat (Anzahl der Dezimalstellen) ebenso steuern können wie eine qualitative Auswahl der ausgedruckten Statistiken oder geeignete graphische Aufbereitungen dieser Ergebnisse, denn oft liefert eine Graphik mehr Information als eine Vielzahl von doppelt genauen Dezimalzahlen (vgl. Tukey, 1977). Idealerweise nimmt auch hier das Programm Rücksicht auf den Wissensstand des Anwenders.

Als nach außen offenes System wird ein Softwareprodukt auch bezüglich der Datenausgabe vielseitig sein. Das Rückspeichern in Datenbanken oder Systemdateien anderer Produkte und die Erstellung bestimmter Dateiformate für nichtflexible Systeme sind hier anzuführen.

2.3 Methodische Flexibilität

Methodische Flexibilität kann als Erweiterung und Wartung bestehender Verfahren oder als völlige Neuentwicklung von statistischen Methoden gesehen werden. Unter Wartung und Erweiterung soll neben Korrekturen der Einbau von zusätzlichen Statistiken und Tests, aber auch eine verbesserte Prüfung von Annahmen und Voraussetzungen eines statistischen Verfahrens, und daraus folgernd ein unter Umständen geändertes Systemverhalten (Warnungen, Fehlermeldungen, Berechnung von Statistiken, etc.) verstanden werden. Dazu werden aber sehr system-/methodennahe Eingriffe benötigt, die bei den konventionell implementierten Systemen mit ausprogrammierten Methoden im Allgemeinen nicht möglich sind. Falls diese Systeme eine Erweiterung im obigen Sinn erlauben und die dazu notwendigen Operationen bereitstellen, so kann dies bestenfalls durch Vor- oder Nachschalten entsprechender Prozeduren oder Makros, niemals jedoch durch echte Eingriffe in das Programm realisiert werden.

Die vollständige (Neu-)Entwicklung von Methoden ermöglicht dem Benützer das Methodenspektrum seines bevorzugten Methodenpakets nach seinen Bedürfnissen zu erweitern und damit die zwangläufig vorhandene Lücken zu schließen. Die Vorteile sind offensichtlich:

- Verkürzung der Spanne zwischen theoretischer Methodenentwicklung und praktischer Methodenimplementierung und -verwendung
- Verbleib in der bekannten, bevorzugten Systemumgebung
- keine zusätzlichen Datenmanipulationen
- Anwendung von problemspezifischen Verfahren, die wegen des geringen Bedarfs von den Systementwicklern nicht implementiert werden

Die Anforderungen, die die methodische Flexibilität an das zugrundeliegende Methodenpaket, aber auch an den Anwender, der diese Möglichkeit nutzen will, stellt, sind ungleich höher als bei den anderen Flexibilitätsbereichen. Als Voraussetzungen für die Realisierung einer akzeptablen Methodenimplementierungsumgebung innerhalb eines statistischen Analysesystems sind folgende Punkte zu nennen:

- Existenz von numerischen und alphanumerische Datenstrukturen (Skalare, Vektoren, Matrizen); von Vorteil kann auch der Datentyp des Pointers zum Definieren von komplexen Datenstrukturen (Records, Verbunde) sein
- einfacher Zugriff auf die Daten in der systemeigenen Datenorganisation
- Zugriff auf die Zwischen- und Endergebnisse anderer Verfahren
- Prozessor für Matrixarithmetik, dessen Schnittstelle zum Benützer der in den statistischen und mathematischen Textbüchern verwendeten Notation[1] folgt; über die grundlegenden Matrixoperationen wie Addition, Subtraktion, Multiplikation, Transponierung, Invertierung, usw. sollten auch weitergehende mathematische und statistische Operationen und Funktionen (z.B. Faktorisierungsverfahren, Eigenwert- und Eigenvektorenberechnung, Kovarianz- und Korrelationsmatrixberechnung) enthalten sein

[1] Die extravagante und gewöhnungsbedürftige Syntax von APL ist auch der Grund, warum nicht weiter auf diese mathematische Programmiersprache eingegangen wird.

- Programmierkonstrukte (Schleifen, Bedingungen und Prozedurdefinitionsanweisungen) zum Verknüpfen der arithmetischen Anweisungen sowie für eine brauchbare Fehlerbehandlung
- Mechanismen zum Schutz oftmals verwendeter Prozeduren vor unbefugten Eingriffen

Der Kernpunkt dieser Umgebung ist die Matrixarithmetik mit zugehöriger Schnittstelle. Durch diese ist eine Entwicklung in jenen Gedankengängen und Konventionen möglich, die auch die statistisch-methodische Literatur verwendet, und der Methodenimplementer wird weder auf die Ebene jener (höheren) Programiersprache gezwungen, in der die Systementwicklung durchgeführt wird, noch benötigt er besondere Kenntnisse über die interne Realisierung.

Für eine wirklich komfortable und effiziente Methodenimplementierungsumgebung wären noch eine Reihe von Punkten, die in modernen Programmentwicklungsumgebungen bereits Eingang gefunden haben, wie syntaxgetriebener Full-Screen-Editor, symbolischer Debugger, usw. anzuführen. Beim momentanen Entwicklungsstand statistischer Analysesoftware ist eine Verwirklichung dieser Forderungen aber in nächster Zeit nicht zu erwarten, und es soll daher nicht näher auf sie eingegangen werden.

Problematischer erweisen sich die Anforderungen, die an den menschlichen Faktor zu stellen sind. Nicht jeder Anwender, den eine Methodenlücke plagt, wird die nötigen Fähigkeiten besitzen, diese zu schließen. Als Voraussetzungen lassen sich jene Fähigkeiten anführen, die das Bild des "Computational Statistician" definieren:

- profunde statistische Methodenkenntnisse, die über das Wissen über die Anwendung von Methoden hinausgehen
- Kenntnisse der numerischen Mathematik
- gute Kenntnisse über Organisation und Aufbau von Statistikpaketen, insbesondere des verwendeten
- grundlegende Programmierkenntnisse

Die Problematik liegt weniger in obigen Forderungen als in der Sicherstellung, daß befähigte Personen und nicht nur Personen, die sich berufen fühlen, das Werkzeug der methodischen Flexibilität

benützen. Denn man muß sich bewußt sein, daß Flexibilität in statistischen Analysesystemen natürlich auch ein Tor in Richtung ungenauer und falscher Methodenimplementierung öffnet. Ein unverantwortliches Verwenden dieses Werkzeuges kann sehr rasch alle Statistik-Software und die Statistik als Wissenschaft in Mißkredit bringen.

2.4 Der Ist-Zustand in statistischer Standard-Software

Bevor der Ist-Zustand der Standardstatistikpakete bezüglich ihrer Flexibilität kurz näher beleuchtet wird, soll nicht unerwähnt bleiben, daß die großen, bekannten und weit verbreiteten Produkte auch ihre Vorzüge besitzen. Besonders angeführt sei hier die Stabilität und die Sicherheit der statistischen und numerischen Algorithmen sowie der gesamten Systemumgebung mit Datenorganisation, Ein- und Ausgabe. Diese Zuverlässigkeit resultiert aus der langandauernden Entwicklung in einem Team versierter Statistiker, Mathematiker und Informatiker, der daraus folgenden Erfahrung, der weiten Verbreitung und den umfassenden Testphasen, denen die Produkte vor ihrer Freigabe unterworfen werden. Dennoch kann ein so hochkomplexes System, wie es ein allgemeines, umfassendes Statistikpaket ist, nicht fehlerfrei sein. Doch ist hier sofort ein weiteres Plus anzuführen: die ständige Wartung durch einen qualifizierten Stab, der mit einiger Zuverlässigkeit auftretende und gemeldete Fehler bis zur nächsten Version korrigiert.

Betrachtet man nun den Ist-Zustand der Flexibilität in Statistikprogrammen, so ist im Bereich der Dateneingabe einiges auch in den gängigen "Dinosauriern" verwirklicht, z.B.:

- die Verarbeitung von anderen als rechteckigen Datenstrukturen
- das Einlesen fremder Systemdateien anderer Statistikpakete und sogenannter integrierter Pakete auf PC's, und zwar entweder direkt oder mittels vorschaltbarer Konversionsprogramme
- Zugriff auf Daten einiger gängiger Datenbanksysteme

Unverändert rigide und unflexibel präsentieren sich in der Regel aber die Kommandoebenen und die Kommandosprachen, wo bereits freiformatierte Eingabe als großer Fortschritt bejubelt wird. Benüt-

zerschnittstellen, die auf den Wissenstand des Anwenders eingehen oder die erweitert und umdefiniert werden können, sind abgesehen von Ausnahmen (GENSTAT5) praktisch unbekannt.

Etwas erfreulicher stellt sich die Ausgabeseite dar. Weit verbreitet sind

- Steuerungsmöglichkeiten von Outputquantität und statistischen Kenngrößen über Optionen oder Schlüsselworte
- Listen- und Tabellengeneratoren
- Erstellung diverser Ausgabe-Dateiformaten
- Datenbank-Interfaces
- integrierte oder gekoppelte Graphikmöglichkeiten

Zu erwähnen bleibt, daß vor allem die letzen drei Punkte durch die Entwicklungen auf dem PC-Markt intensiviert wurden und zum Teil auf diesen beschränkt sind. Eine weiter Ausweitung dieser Fähigkeiten, insbesondere der graphischen Aufbereitungsmöglichkeiten sowohl auf PC's als auch auf Großrechnern, wobei bei der Graphik die Portabilität von Graphik-Software und die Inkompatibilität der graphischen Ausgabegeräte Probleme aufwerfen, wäre wünschenswert.

Auf dem Gebiet der methodischen Flexibilität, auf die sich die restliche Arbeit beschränkt, liegen bei den bekannten und verbreiteten Statistikpaketen nur wenige akzeptable Realisierungen vor. Große statistische Standardpakete wie SPSS und BMDP bieten keinerlei Möglichkeiten aus dem vorgegebenen Methodenkorsett auszubrechen. Andere Entwickler, wie jene von SAS, GENSTAT und RATS, um nur einige zu nennen, haben die Notwendigkeit entsprechender Werkzeuge erkannt und bieten mehr oder weniger gelungene Realisierungen an.

3 Realisierungsansätze

Payne/Lane (1986) stellen in ihrer Arbeit drei Konzepte, die bei der Entwicklung von GENSTAT5 berücksichtigt wurden, für die Erweiterung des Methodenspektrums eines Statistikpaketes vor:

- Hinzufügen von neuem Quellen-Code
- Programmiermöglichkeit in der Kommandosprache
- Zusammenfassung von "Programmen" in der Kommandosprache zu Prozeduren.

Eine kurze Betrachtung dieser weist die erste als die problematischste und unkomfortabelste Möglichkeit aus. Voraussetzung ist, daß im System eine Dummy-Anweisung mit zugehöriger Dummy-Routine enthalten sein muß, die durch den neuen Methodenmodul ersetzt wird. Diese Vorgangsweise zwingt den Statistiker auf die Implementierungsebene und beinhaltet eine Reihe von Nachteilen:

- profunde Programmierkenntnisse
- Systemkenntnisse auf Implementierungebene (z. B.: woher kommen die Daten und wohin sind Ergebnisse zu speichern)
- Neuerstellung eines exekutierbaren Programmes (Linken, Binden) ist erforderlich
- eignet sich praktisch nur für Neuentwicklung von kompletten Methodenmodulen und nicht zum Abändern existierender Module.

Der zweite Ansatz, Programmieren in der Kommandosprache, kann den Gedankengängen eines Anwenders schon wesentlich näher kommen. Voraussetzung dazu sind die in Kap. 2.3 angeführten Punkte (Datenstrukturen, Programmierkonstrukte, Zugang zu den Zwischen- und Endergebnissen anderer Verfahren und Matrixarithmetik). Auch hier ist an und für sich kein Zugang zu den bestehenden, implemetierten Methodenmodulen möglich, und dieser Ansatz unterstützt ebenfalls vorwiegend die Neuentwicklung. Es ergeben sich jedoch interessante Erweiterungsmöglichkeiten durch Vor- und Nachschalten von neuen Prüf- und Teststatistiken.

Naheliegend ist die Zusammenfassung der in der Kommandosprache erstellten Programme zu Prozeduren und das Speichern dieser in einer benützereigenen Methodenbibliothek. Der Aufruf sollte sich natür-

lich möglichst nahtlos in die Benützeroberfläche einfügen (Kap. 2.1). Ansonsten gilt das im vorhergehenden Absatz Gesagte. Vorteil ist, daß die so definierten Methoden auch Anwendern ohne spezifisches Methodenwissen problemlos zur Verfügung gestellt werden können.

Die bisher dargestellten Ansätze konsequent weitergedacht, drängt sich folgende Frage auf:

> Braucht man überhaupt ausprogrammierte und damit unflexible Methoden, oder ist es denkbar, ein ganzes statistisches Analysepaket in einer Methodensprache aufbauend auf ebenfalls in dieser Sprache geschriebenen Basismodulen zu entwickeln?

Dieses Paket könnte, wenn die Syntax der Methodensprache der Denkweise der methodisch versierten Statistiker entspricht in seiner Gesamtheit - Methoden- und auch Basismodule - den Anwendern zur Einsichtnahme, Verbesserung und Erweiterung zur Verfügung gestellt werden; der höchstmögliche Grad an methodischer Flexibilität wäre erreicht. Auch für die anderen Flexibilitätsbereiche könnten unter Anwendung des gleichen Prinzips, vor allem durch Bereitstellung geeigneter Eingabe-/Ausgabe-Funktionen wesentliche Verbesserungen erreicht werden. Programmentwicklungen in diese Richtung gibt es bereits seit einiger Zeit (vgl. z.B.: Momirovic, 1984; Puntanen, 1982). Auch zwei Statistikpakete, die in letzter Zeit Verbreitung gefunden haben, beruhen auf diesen Überlegungen: S und GAUSS.

Je flexibler nun ein Statistikprodukt konzipiert ist, um so größer ist damit aber auch die Gefahr, daß inkompetente und unbefugte Personen Veränderungen durchführen, deren Reichweite nicht abgeschätzt werden kann. Besonders bedenklich ist Flexibilität, wenn mehrere Personen an dem System partizipieren. Daher müssen auch und besonders für flexible Software geeignete Schutzmechanismen gefordert werden. Diese werden den möglichen Eingriffen der Anwender sicher einige Restriktionen auferlegen. Sie in dieser Arbeit zu fordern, mag im ersten Moment als Widerspruch und Rückschritt erscheinen. Die praktischen Erfahrungen bei Software-Entwicklung, Software-Wartung und Software-Verwendung zeigen aber die Unverzichtbarkeit solcher Mechanismen.

Eine denkbarer Lösungsansatz ist eine Aufteilung des Softwarepakets in einzelne Bereiche oder Ebenen,die über wohldefinierte Schnittstellen kommunizieren und miteinander verbunden sind. Änderungen bleiben dem für die Ebene zuständigen Personenkreis vorbehalten. Dieses Konzept kann mit dem der modernen und sicheren Programmiersprache wie MODULA 2 verglichen werden. Designvorstellungen dieser Art (z.B. De Jong, 1986) laufen darauf hinaus, daß lediglich bestimmte Benützergruppen auf die einzelnen Ebenen zugreifen dürfen und können:

- methodisch nichtversierten Anwendern sind nur "kosmetische" Veränderungen seiner Benützeroberfläche gestattet
- methodisch geschulten Benützern ist Zugang zu den Methodenmodulen für Erweiterungen und Ergänzungen in der Methodensprache gewährt
- die Methodiker programmieren die statistischen Basismodule (ebenfalls in der Methodensprache und in Zusammenarbeit mit Numerikern)
- die Sprachkonstrukte und Datenstrukturen, aus denen die Methodensprache besteht, können nur von der Implementierungsgruppe angesprochen werden, die die dahinter stehenden, in einer herkömmlichen (höheren) Programmiersprache geschriebenen Programme und Prozeduren warten.

4 State-of-the-Art

Die statistischen Softwareprodukte, die die Forderung nach methodischer Flexibilität aufgegriffen haben und weitreichende Möglichkeiten zur Manipulation von Matrizen sowie zugehörige Kontrollkonstrukte in ihrer Kommandosprache bieten, lassen sich grob in drei Gruppen klassifizieren:

- Systeme mit aufgesetzten Matrixbefehlen, d.h. es gibt keine Zuweisungen, Operatoren etc. für Matrizen, sondern nur zusätzliche Befehle der Kommandosprache (z.B. TSP, SORITEC)
- Systeme mit Matrixsubsystemen, es werden durch zusätzliche Anweisungen Operationen als Matrixoperationen interpretiert (z.B. Präfix MATRIX in RATS, Aufruf des Moduls IML in SAS)
- Systeme mit konzeptioneller, methodischer Flexibilität und einem umfangreichen Repertoire an Matrix- und Vektoroperationen sowie einer angeschlossenen Sammlung von statistischen Anwenderprozeduren (z.B. GAUSS, S, GENSTAT5)

Um den Rahmen dieser Arbeit nicht zu sprengen, beschränkt sie sich auf eine kurze Gegenüberstellung folgender Analysepakete[2]

-	**TSP** 4.0	(Hall, 1983)
-	**RATS** 2.0	(Doan/Litterman, 1986)
-	**SAS/IML** Vers. 6	(SAS, 1985)
-	**S**	(Becker/Chambers, 1984)
-	**GAUSS** 1.49b	(Edlefsen/Jones, 1986)

Die Darstellungen in diesem Kapitel beziehen sich auf die angegebene Programmdokumentationen und etwaige sonstige Programmbeschreibungen und Veröffentlichungen sowie Softwareverzeichnisse (ESRC, 1986) bis Ende 1986. Einige der Aussagen können daher bereits oder innerhalb kurzer Zeit überholt sein "as statistical software is a perishable item, like meat or fish" (Hamer, 1981).

Bezüglich der Verfügbarkeit und der Preise läßt sich ein sehr heterogenes Bild feststellen (Tabelle 1). Bei den dargestellten Paketen handelt es sich einerseits um Mainframe-Produkte, die auch

[2] Das zitierte Paket GENSTAT5 kann wegen nicht eingelangtem Informationsmaterial nicht behandelt werden.

auf PC's portiert wurden (TSP, RATS, SAS), andererseits um reine PC- bzw. Mini-Computer-Entwicklung (S, GAUSS). Die Preisspanne reicht von 300 US$ bis zum mehr als Zehnfachen.

Tabelle 1: Verfügbarkeit und Preise

	TSP	SAS/IML	RATS	S	GAUSS
Personal Comp. (MS-DOS)	ja	ja	ja	-	ja
Minicomputer (UNIX)	ja	-	-	ja	-[3]
Mainframes	ja	ja[4]	ja[5]	-	-
Preis	300-2700 $	5250 DM[6]	1200 DM	3000 $	300 $

Die Systeme erfüllen natürlich die in Kapitel 2.3 aufgestellten Anforderungen. Lediglich die Forderung nach Schutzmechanismen wurde noch in keinem Produkt aufgegriffen. Zentrale Bedeutung kommt den zur Verfügung stehenden Matrixmanipulations- und Matrixtransformationsmöglichkeiten und der Schnittstelle des Matrixinterpreters zum Benützer, der Syntax zu. Tabelle 2 enthält eine Zusammenfassung wichtiger Matrixoperationen. Dieser Überblick ermöglicht dem Leser einen Vergleich sowohl der bereitgestellten Operationen wie auch deren Verwendung.

Auf spezifische Positiva und Negativa der einzelnen Statistikpakete bei der Erfüllung der Flexibilitätanforderungen wird in den folgenden Unterkapiteln eingegangen. Zur besseren Illustration der Benützerschnittstelle wird die Realisierung eines Moduls zur Berechnung der Parameter einer OLS-Regression als Beispiel angefügt.

[3] geplant für Rechner mit 8088/8086-Prozessor unter XENIX
[4] nur für Systeme mit virtueller 32-Bit-Architektur
[5] ohne Grafik
[6] inklusive dem benötigten Basis-SAS

Tabelle 2: Matrixoperationen in flexiblen statistischen Analysesystemen

	TSP	RATS	SAS/IML	S	GAUSS
Addition	MADD B C A	MATRIX A=B+C	A=B+C	A <- B+C	A=B+C
Multiplikation	MMULT B C A	MATRIX A=B*C	A=B*C	A <- B%*C	A=B*C
Elem. Multiplikation	-	EWISE A(I,J)=B(I,J)*C(I,J)	A=B#C	A <- B*C	A=B.*C
Kronecker-Produkt	-	MATRIX A=KRONEKER(B,C)	A=B@C	A <- ?kroneker(B,C)	A=B.*.C
Log./vergl. Operatoren	-	-	ja	ja	ja
Spaltenkonkatenierung	MMAKE A B C	MAKE A Zeit # B C	A=B||C	A <- cbind(B,C)	A=B~C
Zeilenkonkatenierung	-	-	A=B//C	A <- rbind(B,C)	A=B|C
Submatrizen	UNMAKE A B C	OVERLAY, Schleifen	A=B(r,c)	A <- b[r,c]	A=SUBMAT(B,r,c) A=B[r,c]
Inversion	INV B A	MATRIX A=INV(B)	A=INV(B)	A <- solve(B)	A=INV(B)
Verallg. Inversion	-	-	A=GINV(B)	-	A=PINV(B)
Transponierung	MATRAN B A	MATRIX A=TR(B)	A=B'	A <- t(B)	A=B'
Determinante	INV B A DET	MATRIX A=DET(B)	A=DET(B)	-	A=DET(B)
Lösung eines linearen Gleichungssystems	-	-	x=SOLVE(A,b)	x <- solve(A,b)	x=SOLPD(b,A)
Matrixfaktorisierung					
Cholesky	YFACT B A	A=DECOMP(B)	A=ROOT(B)	A <- chol(B)	A=CHOL(B)
LU	-	-	-	-	A=CROUTP(B)
QR	-	-	-	-	CALL QR(B,"A",fl)
Sing. Value Dekomp.	-	-	CALL SVD(U,Q,V,B)	A <- svd(B)	A=SVD(B,flag)
Eigenwerte, -vektoren	YLDFACT B EW A	EIGEN B EW EV	CALL EIGEN(EW,EV,B)	A <- eigen(B)	EW=EIGSYM(B,EV) EW=EIG(B,EV)
Spezielle Matrizen					
Einheitsmatrix	-	A=IDEN(n)	A=I(n)	A <- diag(n)	A=EYE(n)
Eins-Matrix	-	A=CONST(1)	A=J(r,c,1)	A <- matrix(1,r,c)	A=ONES(r,c)
Momentmatrix X'X	MSQUARE X A	-	-	A <- crossprod(X)	A=MOMENT(X)
Utilities (Summen, Minima, Maxima, ...)	-	ungenügend	reichlich	reichlich	reichlich
Verteilungsfunktionen	einige	wenige	reichlich	reichlich	reichlich

4.1 TSP

Der Time-Series Processor ist eines der weitverbreitetsten ökonometrischen Programmpakete und beinhaltet seit langem die Voraussetzungen für methodische Flexibilität. Der TSP-Benützer kann eigene Methodenmodule in der TSP-Kommandosprache erstellen. Die Kommandosprache enthält die dazu benötigten Bedingungs-, Schleifen- und Prozedurdefinitionsanweisungen, die Möglichkeit in den Prozeduren Verfahren des Paketes zu verwenden, den Zugang zu den Ergebnissen vorangegangener Methodenaufrufe und natürlich Transformationsanweisungen für Skalare, Zeitreihen und Matrizen. TSP ist ein batch-orientiertes Programm, dessen Hauptgewicht auf der Bearbeitung von Zeitreihen liegt und das ein breites Spektrum von linearen und nichtlinearen Schätz- und Simulationsverfahren beinhaltet. Dem gegenüber ist die Mächtigkeit der Programmiermöglichkeit, insbesondere der Matrizenmanipulation, wie sie auch Tabelle 2 ausweist, als bescheiden zu bezeichnen. Dies gilt sowohl für den Umfang der Matrixmanipulationsanweisungen - wichtige Operationen wie elementweise Multiplikation, logische Vergleiche und Verknüpfungen, Zusammensetzen und Aufteilen, sowie diverse Hilfsoperationen sind nicht oder nur sehr rudimentär verfügbar - als auch deren Realisierung. Im Gegensatz zu den anderen hier behandelten Systemen kennt TSP keine algebraische Spezifikation mit Zuweisungen, die der gebräuchlichen mathematischen Notation nahe kommt. Jede Matrixoperation ist vielmehr als Funktionsaufruf ausgeführt und entspricht praktisch einer TSP-Anweisung. Diese Vorgangsweise der aufgesetzten Matrizenbefehle wurde wohl wegen ihrer umkomplizierten Implementierbarkeit gewählt, entspricht aber nicht mehr dem heutigen Standard. TSP sieht in seinem Systemkonzept auch keine Behandlung von Missing Values vor.

Beispiel OLS-Regression:

a) Prozedurdefinition:

```
PROC myreg y x nrow ncol beta;
MSQUARE x xx;
INV xx xxinv;
MMULT (TRANS) x y xy;
MMULT xxinv xy beta;
MMULT x beta yhat;
MSUB y yhat u
INPROD 1 1 u u ressq;
```

```
SET df = nrow-ncol;
MDIV y y ones;
INPROD 1 1 ones y ymean;  SET ymean = ymean/nrow;
MSUB y ymeand ymeand;
MMULT ymean ones ymeand;
INPROD 1 1 ymeand ymeand tssq;
SET rsquare = 1-ressq/tssq;
PRINT beta;
.
Ausgabe weiterer Ergebnisse
.
ENDP myreg;
```

b) Prozeduraufruf:

```
LOAD (FILE='longley.dat',NOPRINT) d1 d2 d3 d4 d5 d6 d7;
SET nobs = @NOB;  SET nvar = 7;  GENR d0 = 1.0;
MMAKE y d7;  MMAKE x d0 d1 d2 d3 d4 d5 d6;
MYREG y x nobs nvar b;
```

4.2 RATS

Auch RATS ist ein primär auf die Bearbeitung und Analyse von Zeitreihen ausgerichtetes und an ökonometrischen Methoden reiches Programmsystem (Regression Analysis of Time-Series), das aber darüber hinaus auch einige Verfahren zur Analyse von Querschnitts- und Paneldaten enthält. RATS, auf Mainframe-, Mini- und Personal-Computer verfügbar, unterstützt sowohl die interaktive, wie auch die Batch-Verarbeitung. Die große Zahl von Datentypen, die das System aufweist, der gute Zugriff auf Ergebnisse von aufgerufenen Statistikprozeduren, das Vorhandensein der benötigten Programmierkonstrukte (Schleifen, Bedingungen, Prozedurdefinition) und arithmetischen Operatoren für Skalar- und Matrixarithmetik könnten ein flexibles Datenanalyseprogramm ergeben. Besonders hervorzuheben sind die Konzepte der Prozedurdefinition und der Prozedurverwendung, die eine fast nahtlose Integration der neuentwickelten Module in die RATS-Kommandosprache erlauben.

Negativ ist jedoch anzumerken, daß das Programm als Ganzes nicht sehr benützerfreundlich ist. Dies gilt auch für die gebotene Programmierumgebung, in der explizite Speicherplatzzuweisungen, Deklarationen der Variablen und verschiedene, vom Datentyp abhängige Anweisungen für Kalkulationen vorgesehen sind, womit es einer konventionellen Programmiersprache sehr nahekommt. Verstärkt wird dieser Effekt auch durch den in einigen Bereichen unzufriedenstel-

lenden Umfang der Matrixmanipulationsmöglichkeiten, wodurch der Anwender frühzeitig gezwungen wird, seine Bedürfnisse auszuprogrammieren. In diesem Zusammenhang sind die geringe Zahl an Dekompositionsalgorithmen, die fehlenden allgemeinen Utilities, vor allem aber die unzureichenden Möglichkeiten des Verknüpfens und Aufspaltens von Matrizen (Konkatenierung und Bilden von Submatrizen) zu nennen.

Das Manual, das sich eher als Tutorial denn als Reference-Manual sieht, bedarf einiger Zeitinvestition und einiges intellektuellen Aufwands bis RATS anspruchsvoll und flexibel verwendet werden kann. Auch der Griff zur Reset- bzw. Ein/Austaste mit nachherigem erneutem Hochfahren des Systems wird dem Benützer des öfteren nicht erspart bleiben, da RATS zwar eine flexible Datenanalyseumgebung ist, aber ein ausgereiftes Exception-Handling fehlt. Die flexible und vielseitige Verwendung ist vor allem für eine interaktive Verwendung am PC schlecht abgesichert, was zu häufigen Systemabstürzen und unverständlichen Fehlermeldungen am falschen Platz führt. Die Behandlung von Missing Values bleibt gänzlich dem Programmiergeschick des Anwenders überlassen.

Beispiel OLS-Regression:

a) Prozedurdefinition:

```
PROCEDURE myreg y x nrow ncol betavec
TYPE RECTANGULAR x
TYPE VECTOR y betavec
TYPE INTEGER nrow ncol
LOCAL VECTOR yhat u ymeand
LOCAL REAL ressq rsquare tssq ymean
LOCAL INTEGER df
DIMENSION yhat(nrow) u(nrow) ymeand(nrow)
MATRIX betavec = INV( TR(x)*x )*TR(x)*y
MATRIX yhat = x*betavec
MATRIX u = y-yhat
MATRIX ressq = TR(u)*u
IEVAL df = nrow-ncol
MATRIX ymean = SUM(y)
EVAL ymean = ymean/nrow
EWISE ymeand(i) = y(i)-ymean
MATRIX tssq = TR(ymeand)*ymeand
EVAL rsquare = 1-ressq/tssq
WRITE betavec
END myreg
```

b) Prozeduraufruf:

```
IEVAL nobs = 16
IEVAL nvar = 7
ALL nvar nobs
OPEN DATA longley.dat
DECLARE RECTANGULAR data(nobs,nvar)
INPUT(UNIT=DATA) data
DECLARE VECTOR yvec(nobs) dyov b(nvar)
OVERLAY data(1,nvar) WITH dyov(nobs)
EWISE yvec(i) = dyov(i)
IEVAL nvar1 = nvar-1
DECLARE RECTANGULAR xmat(nobs,nvar) dxov xov ones
OVERLAY data(1,1) WITH dxov(nobs,nvar1)
OVERLAY xmat(1,2) WITH xov(nobs,nvar1)
OVERLAY xmat(1,1) WITH ones(nobs,1)
EWISE xov(i,j) = dxov(i,j)
MATRIX ones = CONST(1.0)
EXECUTE myreg yvec xmat nobs nvar b
```

4.3 SAS/IML

SAS/IML ist ein Produkt aus der großen SAS-Familie für die interaktive Verarbeitung von Matrizen, die die zentrale Datenstruktur des Pakets darstellen. Hervorzuheben ist die stark an die übliche mathematische Notation angelehnte Programmsprache. Es existieren Befehle für einfache Verzweigungen, bedingte und unbedingte Wiederholungen und Sprünge sowie Dialoggestaltung. Die vorhandenen Matrixoperatoren und eingebauten Funktionen sind umfassend und lassen Anwendungen der linearen Algebra leicht lösbar erscheinen. Der Benützer kann seine Verfahren in Prozeduren ("Modules") speichern, später laden und exekutieren. Diese Prozeduren können Parameter oder globale Größen verwenden, dürfen aber nicht rekursiv sein. Die Kommunikation mit anderen Statistikverfahren erfolgt über SAS-Data-Sets und ist daher etwas mühsam. SAS/IML ist in die SAS-Linie via das Basis-SAS eingebunden; es stehen somit auch dessen Möglichkeiten zur Verfügung (z.B. Fenstertechnik, programmierbare Funktionstasten).

Nachteile sind die Notwendigkeit des Erlernens und der Anschaffung des Basis-SAS und der damit verbundene hohe Preis. Falls man weiters noch an den "üblichen" statistischen Verfahren und Farbgraphik interessiert ist, müssen zumindest noch SAS/STAT und SAS/GRAPH bezogen werden. Andere Produkte wie z.B. SAS/ETS könnten ebenfalls von Interesse sein, wodurch sich die Anschaffungskosten

weiter bedeutend erhöhen. Die Produktlinie der SAS-Inc. bietet aber dafür eine umfassende und flexible Palette statistischer Methoden, in dem lediglich ein klar durchgängiges, auch für SAS/IML gültiges Konzept für die Behandlung von Missing Values nicht erkennbar ist.

Beispiel OLS-Regression:

a) Prozedurdefinition:

```
START myreg(y,x,beta);
beta = INV(x`*x)*x`y;
yhat = x*beta;
u = y-yhat;
ressq = u`*u;
df = NROW(x)-NCOL(x);
ymeand = y-y[:];
rsquare = 1-((u`*u)/(ymeand`*ymeand));
PRINT beta;
.
Ausgabe weiterer Ergebnisse
.
FINISH;
```

b) Prozeduraufruf:

```
RESET NOPRINT;
USE longley;                    /* Die Daten werden aus einem */
READ ALL INTO data;             /*     SAS-Data-Set gelesen   */
y = data[,7];
x = J(16,1,1)||data[,1:6];
RUN myreg(x,y,beta);
```

4.4 S

S ist als methodisch flexibles Datenanalysesystem auf Minicomputern unter dem Betriebssystem UNIX entwickelt worden. Der zentrale Datentyp ist der Vektor, alle anderen Datentypen werden daraus aufgebaut. Unterstützte Datentypen sind z.B. Zeitreihen, mit Zeit und Periodizität verbunden, oder kategoriale Variable mit Wertebereich und Werteetiketten; Matrizen werden als mehrdimensionale Vektoren gebildet. Die Programmsprache wirkt durch die Verwendung vieler Sonderzeichen und dem Versuch, ein objektorientiertes Analyse- und Programmierungskonzept zu verwirklichen, auf ungeübte S-Benützer kryptisch. Die wichtigsten Befehle für einen strukturierten Programmablauf liegen vor. Eine Vielzahl von Operatoren und eingebauten Funktionen sowie viele vorgefertigte statistische Me-

thoden, ebenfalls als Funktionen ausgefertigt, stehen dem Benützer zur Verfügung. Weitere Eigenschaften sind On-line Help, Durchgriff auf UNIX und damit Verwendung beliebiger Editoren, geräteunabhängige Graphik mit Betonung der explorativen Datenanalyse und die Definition von Prozeduren ("Macros") für beliebige Aufgabestellungen.

Nachteile sind vor allem der erhöhte Lernaufwand für die Programmsprache, die nur einem UNIX-Kundigen entgegenkommt, der relativ hohe Preis, ein bescheidenes Verarbeitungstempo, die (noch) geringe Verbreitung in Europa und ein ungenügendes Missing Values-Konzept.

Beispiel OLS-Regression:

a) Prozedurdefinition:

```
define
MACRO myreg(x,y)
beta <- solve(t(x)%*x)%*t(x)%*y
yhat <- x%*beta
u <- y-yhat
ressq <- t(u)%*u
df <- nrow(x)-ncol(x)
ymeand <- y-mean(y)
rsquare <- 1-((t(u)%*u)/(t(ymeand)%*ymeand))
print(beta)
.
Ausgabe weiterer Ergebnisse
.
rm(beta,yhat,u,ressq,df,ymeand,rsquare,value=beta)
END
```

b) Prozeduraufruf:

```
data <- matrix(read("longley.dta"),ncol=7,byrow=TRUE)
y <- data[ ,7]
x <- data[ ,1:6]
b <- ?myreg(cbind(1,x),y)
```

4.5 GAUSS

GAUSS als Mathematik- und Statistiksystem besteht aus den Teilen GAUSS-Programmsprache, einer reinen Matrixmanipulationssprache mit angeschlossenem Full-Screen-Editor, sowie einer Sammlung von fertigen Anwendungsprogrammen, die in dieser Programmsprache geschrieben sind. Die Programmsprache orientiert sich stark an der

gängigen mathematischen Notation, sodaß der versierte Benützer schnell eigene Verfahren in Prozeduren ablegen kann. Zentraler Datentyp ist die zweidimensionale Matrix, für die mehr als 250 Operatoren und eingebaute Funktionen existieren. Diese sind prinzipiell im Assembler der 8086-Prozessorfamile codiert. Manche der Operationen und Prozeduren greifen aber direkt auf Funktionen des arithmetischen Co-Prozessors zu, woraus die Schnelligkeit und die Genauigkeit - manche komplexe Matrixoperationen werden auf 80 Bit genau berechnet (das entspricht ca. 19 signifikanten Stellen!) - von GAUSS folgt. Weitere Gestaltungselemente sind Verzweigungs, Iterations- und Sprungbefehle sowie Dialogsteuerung. Zugriff auf das Betriebssystem MS-DOS ist erlaubt und damit die Verwendung beliebiger Editoren, falls der eingebaute Full Screen Editor von GAUSS nicht ausreicht. Ein ebenfalls komplett in GAUSS geschriebener Graphikmodul für Schwarz-Weiß-Graphiken liegt ebenso vor.

Als Nachteile sind die unbedingte Erfordernis eines mathematischen Co-Prozessors 8087/80287/80387, das noch eingeschränkte, lieferbare Methodenspektrum sowie die uneinheitliche und unübersichtliche Dokumentation anzusehen. Letzteres gilt vor allem für die Anwendungsprogramme, für deren Verwendung der Anwender vom Programming-Language-Manual, das für Version 1.49b wesentlich gegenüber früheren Versionen verbessert wurde, über auf den Disketten gespeicherten Dokumentationsfiles, auf den GAUSS-Programmcode verwiesen wird. Auf die Zumutbarkeit dieser Vorgangsweise für Standard-Anwendungs-Module, selbst in einer flexible Methodenumgebung wie GAUSS, soll hier nicht eingegangen werden.

Beispiel OLS-Regression:

a) Prozedurdefinition:

```
PROC myreg(y,x);
LOCAL beta,yhat,u,ressq,df,ymeand,rsquare;
beta = INV(x'x)*x'y;
yhat = x*beta;
u = y-yhat;
ressq = u'u;
df = ROWS(x)-COLS(x);
ymeand = y-MEANC(y);
rsquare = 1-((u'u)/(ymeand'ymeand));
PRINT beta;
```

```
.
Ausgabe weiterer Ergebnisse
.
RETP(beta);
ENDP;
```

b) Prozeduraufruf:

```
LOAD data[16,7] = longley.dta;
y = data[.,7];    x = ONES(16,1)~data[.,1:6];
b = myreg(x,y);
```

5 Zusammenfassung

Das Design der bekannten und weitverbreiteten statistischen Standardsoftware basiert auf Konzepten, die in den frühen 70-er Jahren, oft bereits Mitte bis Ende der 60-er Jahre entwickelt wurden. Sowohl in der Computerwissenschaft als auch in der Statistik sind in der Zwischenzeit tiefgreifende Veränderungen eingetreten. Die Statistiksysteme wurden zwar laufend weiterentwickelt - es wurden neue Verfahren hinzugefügt, eine problemadäquate Datenorganisation wurde durch entsprechende Subsysteme entweder ermöglicht bzw. wesentlich vereinfacht und die Benützerschnittstellen wurden bedienungsfreundlicher gestaltet. Dennoch klafft heute die Kluft zwischen den Vorstellungen der Benützer und der Realisierung weiter denn je. Ein Lösungsansatz sind nun Softwareprodukte, die Eingriffe und damit Erweiterungen, Korrekturen und Anpassungen durch den Anwender gestatten.

Im methodischen Bereich stellt die Forderung nach Flexibilität sowohl an die Statistikpakete als auch an die Anwender neue Anforderungen. Eine Methodenentwicklungsumgebung mit einfachem Zugriff auf Daten und Zwischenergebnisse anderer (implementierter) Verfahren, mit entsprechenden Programmierkonstrukten als Teil der Kommandosprache und einer effizienten Matrixalgebra sollte bereits im Konzept berücksichtigt werden. Die Anforderungen an die Anwender sind wohl nur über eine entsprechende Ausbildung zum Computational Statistician erfüllbar.

Aus den fünf in dieser Arbeit einander gegenübergestellten Ansätzen hebt sich in puncto Flexibilität das Programm GAUSS deutlich von den anderen ab. Flexibilität und effiziente Verwendbarkeit sind primäre Designziele. Gepaart mit dem erfreulich niedrigem Preis ergibt sich so ein äußerst attraktives Softwarepaket, das über manche Unzulänglichkeiten beim Methodenspektrum (diese sind ja einfach auszumerzen) und der Dokumentation hinwegtröstet.

SAS/IML stellt eine gelungene und dem methodisch versierten und interessierten Anwender ebenfalls sehr entgegenkommende Abrundung der umfassenden SAS-Produktpalette dar. Auch das Paket S bietet ein hohes Maß an methodischer Flexibilität. Beachtenswert ist S

aber auch wegen der Realisierung einer objektorientierten Benützerschnittstelle.

Bei den beiden anderen Produkte (TSP und RATS) ist die Flexibilität durch den Umfang der zur Verfügung sthenden Operationen und Befehle mehr aber noch durch die mangelhafte Ausführung der Schnittstelle zur Zeit stark eingeschränkt. Im Gegensatz zu TSP, das nur durch eine völlige Neukonzeption der Matrixoperationen den erforderlichen Standard erreichen könnte, sind die erforderlichen Werkzeuge in RATS konzeptionell gut eingebettet. Damit diese auch handhabbar sind, bedarf es aber eines durchgehenden Exception-Handlings und sinnvoller Fehlermeldungen.

Keines der Programme sieht noch Schutzmechanismen vor, die ausgetestete und oft verwendete Prozduren und Module vor unbefugten und auch unbeabsichtigten Eingriffen schützen. Solche Mechanismen werden aber vor allem für Systeme, die von mehreren Statistikern verwendet werden, von Nöten sein. Sie und die entsprechende Ausbildung der methodenentwickelnden Statistiker sind der Garant dafür, daß die Statistik und die Statistik-Software nicht durch inkompetente und mißbräuchliche Verwendung der Werkzeuge der methodischen Flexibilität in Mißkredit gerät.

Literatur:

BECKER R.A., CHAMBERS J.M. (1984): S - An Interactive Environment for Data Analysis and Graphics. Wadsworth, Belmont.

CHAMBERS J.M.(1980): Statistical Computing: History and Trends. The American Statistician, 34, 238-243.

CHAMBERS J.M. (1982): Analytical Computing: Its Nature and Needs, in Caussinus H. et al. (eds.): COMPSTAT 1982, Physica, Wien.

DE JONG V.J (1984): A Multilevel Approach to Improve the Flexibility of Statistical Software. In: Havranek T. et al. (eds.): COMPSTAT 1984, Wien, 323-328.

DE JONG V.J. (1986): CONDUCTOR: A Multilevel Environment for the Development of Statistical Software. Research Report Econometric Institute, University of Groningen.

DOAN T.A., LITTERMAN R.B. (1986): User's Manual RATS Version 2.00. VAR Econometrics, Minneapolis.

EDLEFSEN L.E., JONES S.D. (1986): GAUSS Programming Language Manual, Software Version 1.49b. Aptech Systems, Inc., Kent.

ESRC (1986): Econometric software register. ESRC Centre in Economic Computing: London School of Economics and Political Science.

HAMER R.M. (1981): Papers that evaluate computer programs. The American Statistician, 35, 264.

MOLENAAR I.W. (1984): Behavioral studies of the software user. Computational Statistics & Data Analysis, 2, 1-12.

MOMIROVIC K., STALEC J., ZAKRAJSEK E. (1982): A Programming Language for Multivariate Data Analysis. In: Caussinus H. et al. (eds.): COMPSTAT 1982, Wien.

PAYNE R.W., LANE P.W. (1986): Design Criteria for a flexible Statistical Language. In: De Antoni F. et al. (eds.): COMPSTAT 1986, Physica, Heidelberg-Wien.

PUNTANEN S., PUKKILA T. (1982): The Use of MATRIX, an Interactive Computer System for Analysis of Matrices, in the Teaching of Statistics. Report A 79, Dept. of Mathematical Sciences, University of Tampere.

RODLER K., HANAPPI G. (1986): Entwickungstendenzen ökonometrischer Software. Report Nr. 58: Institut für Ökonometrie und Operations Research, TU Wien.

SAS Institute Inc. (1985): SAS/IML^{TM} User`s Guide for Personal Computers, Version 6 Edition. SAS Institute Inc., Cary.

TUKEY J.M. (1977): Exploratory Data Analysis. Addison-Wesley, Reading.