UNIVERSITÄT POTSDAM

Wirtschafts- und Sozialwissenschaftliche Fakultät

STATISTISCHE DISKUSSIONSBEITRÄGE
Nr. 7

Hans Gerhard Strohe Frank Geppert

DPLS

Algorithmus und Computerprogramm für
dynamische Partial-Least-Squares-Modelle

Potsdam 1997
ISSN 0949-068X

STATISTISCHE DISKUSSIONSBEITRÄGE

Nr. 7

Hans Gerhard Strohe Frank Geppert

DPLS

Algorithmus und Computerprogramm für dynamische Partial-Least-Squares-Modelle

Herausgeber: Prof. Dr. Hans Gerhard Strohe, Lehrstuhl Statistik
der Wirtschafts- und Sozialwissenschaftlichen Fakultät
der Universität Potsdam
Postfach 90 03 27
D-14439 Potsdam
Tel. (+49 331) 977-32 25
Fax. (+49 331) 977-32 10
1997, ISSN 0949-068X

Inahltsverzeichnis

1. Einführung

Lineare Modelle mit latenten Variablen sind seit langem verbreitete Analyse- und Prognoseinstrumente in den Sozialwissenschaften. Auch in der Ökonometrie gibt es einige Anwendungen.

Die meistverbreiteten Modellierungs- und Schätzverfahren sind LISREL von Jöreskog und Sörbom (z.B. 1987) und Partial Least Squares (PLS) von H. Wold (1973). Während LISREL mehr modellorientiert und in der Anwendung konfirmativ ist, kann man PLS als datenorientiert und eher deskriptiv oder explorativ bezeichnen. Charakteristisch für Wolds Herangehen ist, daß das PLS-Modell eigentlich nur durch den Algorithmus zu seiner Schätzung definiert wird.

Das umfassendste Programmsystem für PLS ist LVPLS von J. B. Lohmöller (1984). Es lehnt sich sehr eng an die Theorie von Wold an und ist trotz mangelnden Nutzerkomforts in seiner Vielseitigkeit und Zuverlässigkeit unübertroffen. Weder Wolds Verfahren noch Lohmöllers Programm sehen die Anwendung auf dynamische Modelle, etwa VARs, explizit vor. Die Einbeziehung verzögerter Variablen ist nur in Form selbständiger Variablen möglich, was zu Inkonsistenzen bei der Gewichtung führt.

Im folgenden zweiten Abschnitt wird ein Verfahren skizziert (vgl. Strohe 1995), das sich einerseits sehr eng an den Woldschen Algorithmus anlehnt, das aber andererseits speziell auf die Behandlung von dynamischen Modellen mit verzögerten latenten Variablen ausgerichtet ist.

Der dritte Abschnitt bringt dann eine Einführung in das entsprechende ISP™ Computerprogramm DPLS (vgl. Geppert 1995). Er besteht aus einer allgemeinen Programmbeschreibung und einer detaillierten Nutzeranleitung. Hinzu kommt die Bearbeitung eines kleinen ökonometrischen Demonstrationsmodells.

Im vierten Abschnitt werden mit einer Simulationsstudie die Eigenschaften des Schätzverfahrens DPLS unter verschiedenen Verteilungsannahmen geprüft. Der Anhang bringt die vollständigen Listings der kommentierten Programm-Macros.

2. Das dynamische Pfadmodell

2.1 Die Modellstruktur

Eine ausführliche und tiefgründige Darstellung von Theorie und Praxis von Pfadmodellen mit latenten Variablen und des Schätzvefahens PLS findet man bei Lohmöller (1989). Die Begründung der Notwendigkeit einer Erweiterung auf Modelle mit zeitverzögerten latenten Variablen sowie der dafür geeignete heuristisch abgeleitete Algorithmus DPLS sind zu finden in Strohe (1995). Daher soll hier die dynamische Partial-Least-Squares-Modellierung nur kurz skizziert werden.

Das PLS- Modell besteht aus einem inneren Modell und einem äußeren Modell. Das innere Modell beschreibt die linearen Beziehungen zwischen den latenten Variablen. Es kann in Matrixform folgendermaßen geschrieben werden:

$$\eta_t = \mathbf{B}\eta_t + \mathbf{C}\eta_{t-1} + \nu_t \qquad (1)$$

wobei η_t ein Vektor von K latenten Variablen mit dem Mittelwertsvektor **0** und dem Einsvektor als Varianz und ν_t ein Störterm mit Erwartung Null ist. Der Zeiger t bezeichnet die Zeit: t=1,...,T. Die K×K-Matrix **B** wird ohne Beschränkung der Allgemeinheit als triangulär mit Nulldiagonale vorausgesetzt, um die Rekursivität des Modells zu sichern. Die KxK- Koeffizientenmatrix **C** für die verzögerten Abhängigkeiten ist ebenfalls von Dreiecksform, die Diagonalelemente können aber von Null verschieden sein, um Autoregression zuzulassen.

Die weitere Struktur von **B** und **C** wird entsprechend der angenommenen Abhängigkeitsstruktur zwischen den latenten Variablen durch Nullrestriktionen in Design-Matrizen $\mathbf{D}_B$ bzw. für die verzögerten Beziehungen $\mathbf{D}_C$ geregelt.

Ein Beispiel (Abb. 1) für ein inneres Eingleichungsmodell mit 3 latenten Variablen ist

$$\eta_t^3 = b_1\, \eta_t^1 + b_2\eta_t^2 + c_1\eta_{t-1}^1 + c_2\eta_{t-1}^2 + c_3\eta_{t-1}^3 + \nu_t \qquad (2)$$

mit den Pfadkoeffizientenmatrizen

$$B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ b_1 & b_2 & 0 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ c_1 & c_2 & c_3 \end{bmatrix}$$

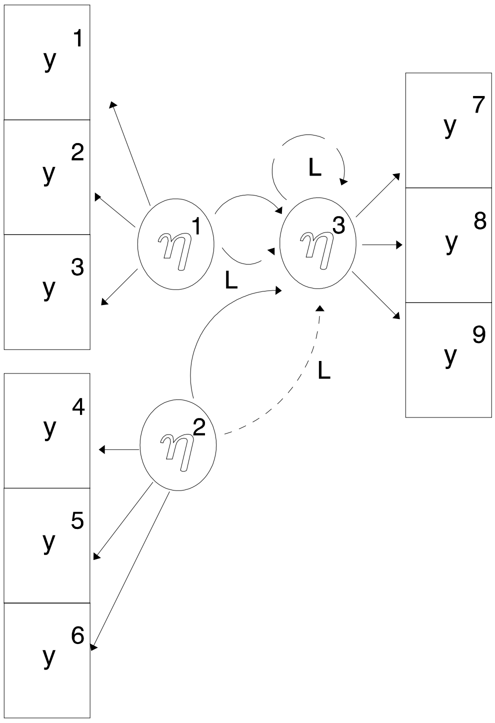


Abb. 1: Dynamisches Drei-Block-Pfadmodell

Das äußere Modell beschreibt, wie die nicht beobachtbaren latenten Variablen durch die beobachtbaren manifesten Variablen gemessen werden können. Seine Struktur wird durch die blockdiagonale Designmatrix $\mathbf{D}_W$ festgelegt. Diese MxK-Matrix gibt durch Spaltenabschnitte von Einsen an, welche der M manifesten Variablen zu welcher latenten Variablen gehören. Nichtzugehörigkeit wird durch 0 gekennzeichnet.

Das äußere Modell besteht genau wie im klassischen PLS-Modell aus der Ladungs- und der Gewichtsbeziehung. Die Ladungsbeziehung beschreibt die lineare Abhängigkeit der manifesten Variablen von den latenten Variablen:

$$\mathbf{y}_t = \mathbf{P}\boldsymbol{\eta}_t + \boldsymbol{\nu}_t, \tag{3}$$

wobei $\mathbf{P}$ eine M×K-Matrix von Ladungen, $\mathbf{y}_t$ ein Spaltenvektor von M manifesten Variablen zum Zeitpunkt t und $\boldsymbol{\nu}_t$ ein Störterm mit Mittelwert Null ist. Ohne wesentliche Beschränkung der Allgemeinheit sei y_t zentriert, das heißt über t=1,..,T im Mittel gleich 0.

Die Gewichtsbeziehung definiert die latenten Variablen als Linearkombinationen der ihnen zugeordneten manifesten Variablen, also als Komponenten im Sinne der Hauptkomponentenanalyse oder der kanonischen Korrelation**:**

$$\boldsymbol{\eta}_t = \mathbf{W}' \, \mathbf{y}_t \tag{4}$$

Die Gewichtsmatrix **W** hat die gleiche blockdiagonale Struktur wie **P**.

Im oben angeführten Beispiel ist das äußere Modell

$y_t^m = p_{m1}\, \eta_t^1 + \varepsilon_t^m$ für $m = 1,\ldots,3$

$y_t^m = p_{m2}\, \eta_t^2 + \varepsilon_t^m$ für $m = 4,\ldots,6$

$y_t^m = p_{m3}\, \eta_t^3 + \varepsilon_t^m$ für $m = 7,\ldots,9$

und

$\eta_t^1 = w_{11}\, y_t^1 + w_{21}\, y_t^2 + w_{31}\, y_t^3$

$\eta_t^2 = w_{42}\, y_t^4 + w_{52}\, y_t^5 + w_{62}\, y_t^6$

$\eta_t^3 = w_{73}\, y_t^7 + w_{83}\, y_t^8 + w_{93}\, y_t^9$

mit den Ladungs- und Gewichtsmatrizen

$$P = \begin{bmatrix} p_{11} & 0 & 0 \\ p_{21} & 0 & 0 \\ p_{31} & 0 & 0 \\ 0 & p_{42} & 0 \\ 0 & p_{52} & 0 \\ 0 & p_{62} & 0 \\ 0 & 0 & p_{73} \\ 0 & 0 & p_{83} \\ 0 & 0 & p_{93} \end{bmatrix}, \quad W = \begin{bmatrix} w_{11} & 0 & 0 \\ w_{21} & 0 & 0 \\ w_{31} & 0 & 0 \\ 0 & w_{42} & 0 \\ 0 & w_{52} & 0 \\ 0 & w_{62} & 0 \\ 0 & 0 & w_{73} \\ 0 & 0 & w_{83} \\ 0 & 0 & w_{93} \end{bmatrix},$$

Dabei ist y_t^m der Wert der m-ten manifesten Variablen zum Zeitpunkt t.

Das innere dynamische PLS-Modell (1) kann in seiner äußeren Form in die Gestalt des PLS-Modells tranformiert werden:

$$\eta_t = \mathbf{F}\eta_t + \nu_t \tag{5}$$

wobei

$$\mathbf{F} = \mathbf{B} + \mathbf{C}L \tag{6}$$

eine Matrix ist, die den Lag-Operator L enthält.

Der Schätzalgorithmus DPLS besteht nun darin, auf dieses Modell (6) das formal gleiche Verfahren wie in Wolds PLS-Algorithmus anzuwenden, nur unter Berücksichtigung, daß es sich bei **F** um eine Operator-Matrix mit einer etwas komplizierteren Struktur als der der alten Pfad-koeffizientenmatrix handelt.

2.2 PLS-Schätzung mit dynamischer innerer Approximation

In diesem Abschnitt wird für Näherungs- oder Schätzwerte die gleiche Symbolik wie für die "wahren" Parameter verwendet. Dies ist der Übersichtlichkeit halber gerechtfertigt, solange keine Verwechslungsgefahr besteht. Hinzu kommt die Eigentümlichkeit schon des klassischen PLS-

Verfahrens, daß das Modell mit seinen Parametern eigentlich erst durch den iterativen Schätz-algorithmus definiert wird.

Zu Beginn muß die Modellstruktur durch Festlegung der logischen Design-Matrizen $\mathbf{D_B}$, $\mathbf{D_C}$ und $\mathbf{D_W}$ entsprechend den unverzögerten und verzögerten Abhängigkeiten zwischen den latenden Variablen bzw. der Blockzugehörigkeit der manifesten Variablen fixiert werden.

$\mathbf{D}_B$ erhält an der Stelle (j,k) eine 1, wenn die j-te latente Variable von der k-ten ohne Verzögerung abhängt, analog $\mathbf{D}_C$ bei Abhängigkeit mit Verzögerung. $\mathbf{D}_W$ erhält an der Stelle (m,k) eine 1 wenn die m-te manifeste Variable $\mathbf{y}^m$ zum Block der k-ten latenten Variablen gehört.

Der entscheidende Teil des DPLS-Algorithmus ist, wie auch sonst bei PLS, die iterative Schätzung der Gewichte, mit denen die manifesten Variablen in die latenten Variablen eingehen, also der Gewichtsmatrix **W**.

Hierzu werden iterativ folgende Schritte durchlaufen:

1. Konstruktion latenter Variablen mit Hilfe mehr oder weniger willkürlich gewählter Start-werte für die Gewichte gemäß der Beziehung

$$\eta_t = \mathbf{W}'\mathbf{y}_t \tag{7}$$

2. Standardisierung der latenten Variablen auf Varianz 1:

$$\eta_t := \sqrt{T}(I * \mathrm{HH}')^{-1/2}\,\eta_t \tag{8}$$

wobei **H** die K×T-Matrix aller Werte (scores) des Vektors η_t für t = 1, ..., T ist. Die Diagonal-Elemente von $\mathbf{HH}'/T$ sind die empirischen Varianzen der individuellen Variablen im Vektor η_t. Die elementweise Multiplikation von Matrizen wird mit * dargestellt.

3. Berechnung von Umgebungsvariablen entsprechend dem inneren Pfadmodell als gewogene Summe aus allen latenten Variablen, mit denen die jeweilige Variable durch einen Pfad, das heißt durch eine verzögerte oder unverzögerte Beziehung, verbunden ist. Als Gewichte werden hier die entsprechenden Elemente der Korrelationsmatrix beziehungsweise die Autokorrelationsmatrix der latenten Variablen verwendet :

$$\begin{aligned} \eta_t^* &= \mathbf{F}^*\eta_t \\ &= (\mathbf{B}^* + \mathbf{C}^*L + \mathbf{C}^{*\prime}L^{-1})\,\eta_t \\ &= \mathbf{B}^*\eta_t + \mathbf{C}^*\eta_{t-1} + \mathbf{C}^{*\prime}\eta_{t+1} \end{aligned} \tag{9}$$

$\mathbf{B}^*$ und $\mathbf{C}^*$ sind im wesentlichen die Korrelations- und Autokorrelationsmatrizen **R** bzw. **A**, aber durch Anwendung der Designmarizen $\mathbf{D}_B$ bzw. $\mathbf{D}_C$ beschränkt auf Elemente die für vorab postulierte Zusammenhänge stehen. Alle anderen Elemente werden durch die Nullrestriktionen im Modell gelöscht. Die Richtung des Zusammenhanges wird an dieser Stelle nicht berücksichtigt:

$$\mathbf{B}^* = (\mathbf{D_B} + \mathbf{D_B}')*\mathbf{R} \quad (10)$$

$$\mathbf{C}^* = \mathbf{D}_C*\mathbf{A} \quad (11)$$

$$\mathbf{R} = \mathbf{HH}'/T \quad (12)$$

$$\mathbf{A} = \mathbf{H}(L\mathbf{H})'/T \quad (13)$$

$$\mathbf{H} = (\eta_1, \eta_2, ..., \eta_T), \quad (14)$$

$$L\mathbf{H} = (\eta_0, \eta_1, ..., \eta_{T-1}). \quad (15)$$

4. Als Kleinstquadratschätzung für lineare Regressionen zwischen der Umgebungsvariablen und den dazugehörenden manifesten Variablen werden neue Gewichte bestimmt:

$$y_t^m = w_{mk}\, \eta_t^{*k} + \nu_t^{mk} \qquad \text{wenn } d_{mk}=1 \qquad \text{(Wolds Modus A)} \quad (16)$$

wobei $\mathbf{D}_W = [d_{mk}]$ die Gewichtsdesignmatrix ist.

5. Die so bestimmten neuen Gewichte werden anstelle der Startwerte für die Matrix $\mathbf{W}:=[w_{mk}]$ eingesetzt. Fortgesetzt wird bei Schritt 1.

Der Iterationsprozeß wird abgebrochen, wenn sich die Gewichte oder die Scores der latenten Variablen in einem Zyklus nicht mehr wesentlich von denen im vorangegangenen unterscheiden.

Anschließend können die Pfadkoeffizientenmatrizen **B** und **C** entprechend dem inneren Modell

$$\eta_t = (\mathbf{B} + \mathbf{C}L)\, \eta_t + \nu_t \quad (17)$$

mit einer geeigneten Methode der dynamischen Modellierung, aber auch mit OLS oder GLS geschätzt werden. Die Ladungen **P** des äußeren Modells werden als einfache Kleinstquadratschätzung entsprechend (3) bestimmt.

2.3 Prognose und Anpassungsgüte

Durch Einsetzen von (3) für $\mathbf{y}_t$ in (5) erhalten wir

$$\mathbf{y}_t = \mathbf{P}\eta_t + \varepsilon_t$$

$$= \mathbf{PF}\eta_t + \mathbf{P}\nu_t + \varepsilon_t \quad (18)$$

Dann wird (4) für η_t eingesetzt:

$$\mathbf{y}_t = \mathbf{PFW}'\mathbf{y}_t + \mathbf{P}\nu_t + \varepsilon_t$$

$$= \quad \mathbf{P(B+C}\mathrm{L}\mathbf{)W'y}_t + \mathbf{P}\nu_t + \varepsilon_t \tag{19}$$

So liefert die Ausführung des Lag-Operators die Prognoseformel

$$\mathbf{y}_t \quad = \quad [\mathbf{PBW'y}_t + \mathbf{P\ C\ W'\ y}_{t-1}] + [\mathbf{P}\nu_t + \varepsilon_t] \tag{20}$$

Diese Prognoseformel kann im Beobachtungsbereich auch zur Konstruktion eines Anpassungsmaßes genutzt werden:

Aus (18) folgt, daß der vorhersagbare Teil von $\mathbf{y}_t$ das Glied $\mathbf{y}_t^* = \mathbf{PF}\eta_t$ ist. Mit $\mathbf{Y}^*=(\mathbf{y}_1^*, ..., \mathbf{y}_T^*)$ werde jetzt der gesamte durch das Modell vorhersagbare Teil der manifesten Datenmatrix $\mathbf{Y}$ bezeichnet. Dann ist die Kovarianz dieser Vorhersagen

$$\begin{aligned} \mathbf{cov}(\mathbf{Y}^*) &= \mathbf{P\ F\ cov}(\mathbf{H})\ \mathbf{F'\ P'} \\ &= \mathbf{P\ F\ R\ F'\ P} \end{aligned} \tag{21}$$

wobei $\mathbf{R}$ die Korrelations- oder Kovarianzmatrix der latenten Variablen $\mathbf{H} = (\eta_1, ..., \eta_T)$ ist.

Es folgt

$$\begin{aligned} \mathrm{T}\mathbf{cov}(\mathbf{Y}^*) &= \mathbf{P(B+C}\mathrm{L}\mathbf{)\ H\ H'\ (B+C}\mathrm{L}\mathbf{)'\ P'} \\ &= \mathbf{(PB+PC}\mathrm{L}\mathbf{)\ H\ [(PB+PC}\mathrm{L}\mathbf{)\ H]'} \\ &= \mathbf{PBHH'B'P' + PC(}\mathrm{L}\mathbf{H)H'B'P' + PBH(}\mathrm{L}\mathbf{H)'C'P' + PC(}\mathrm{L}\mathbf{H)(}\mathrm{L}\mathbf{H)'C'P'} \\ &\approx \mathrm{T}\ \{ \quad \mathbf{PBRB'P' + PBAC'P' +} \\ &\qquad\quad \mathbf{PCRC'P' + (PBAC'P')'}\ \} \end{aligned}$$

mit $\mathbf{A}$ als Autokorrelationsmatrix erster Ordnung. Die leichte Unschärfe in der letzten Beziehung rührt daher, daß sich die Kovarianzen der latenten Variablen $\mathbf{HH'}/\mathrm{T}$ und die der verzögerten latenten Variablen $(\mathrm{L}\mathbf{H})(\mathrm{L}\mathbf{H})'/\mathrm{T}$ vor allem bei kurzen Reihen unterscheiden können.

Bezeichnen wir

$$\mathbf{cov}(\mathbf{Y}^*) \approx \mathbf{PBRB'P' + PCRC'P' + PBAC'P' + (PBAC'P')'} = \mathbf{G}^*, \tag{22}$$

dann ist leicht zu sehen, daß $\mathbf{G}^*$ in seiner Diagonale die Varianzen des prognostizierbaren Teils, oder wie Lohmöller (1989) sagt, des "redundanten" Teils, der manifesten Variablen enthält.

Folgt man Lohmöller weiter, so kann man den "Quotienten" zweier Varianz-Diagonalmatrizen berechnen:

$$\mathbf{G} \quad = \quad (\mathbf{I} * \mathbf{G}^*)\ (\mathbf{I} * \Sigma_y)^{-1} \tag{23}$$

wobei Σ_y die empirische Kovarianz-Matrix der manifesten Variablen bezeichnet.

Die Zahlen in der Diagonalen von $\mathbf{G}$ sind Quotienten, die ausdrücken, in welchem Grad die Varianz jeder manifesten Variablen durch das Modell reproduziert wird. Der Durchschnitt dieser Maßzahlen

$$\mathrm{G}^2 \quad = \quad \mathrm{trace}\ \mathbf{G} / \mathrm{M}^* \tag{24}$$

mit der Anzahl endogener manifester Variablen M^* ist der Redundanzkoeffizient oder die mittlere Redundanz. Sie wird als Anpassungsmaß und als Maß für die Prognoserelevanz des DPLS-Modells verwendet.

3. Das Computer-Programm DPLS

Für den oben beschriebenen Ansatz wurde ein Programm entwickelt (Geppert 1995), das inzwischen um Simulations- und Redundanzprozeduren sowie einen Eingabedialog erweitert wurde.

3.1 Allgemeine Erläuterungen zum Programm DPLS

Die hier aufgeführten Programme laufen nur in der statistischen Programmumgebung ISP™ unter DOS. Sie ist ein eingabeorientiertes System mit einer Vielzahl von mathematischen und statistischen Prozeduren und Funktionen und der Möglichkeit, eigene Befehlsketten in ASCII-Dateien zu speichern.

Das Analysemacro, das die dynamische PLS-Analyse durchführt (also das Kernstück), heißt DPLS. Für die komfortable Bedienung dieses Analysemacros wurde das Mantelprogramm DPLSMAC geschrieben und für die Redundanzberechnung das Macro REDUN.

Das Redundanzmacro, als eigenständiges Programm, berechnet die Redundanz des gesamten dynamischen Partial-Least-Squares-Modells und liefert außerdem Einzelredundanzen für jede endogene latente Variable in einem Redundanzvektor. Es benötigt Eingabedaten, die man als Resultate vom DPLS-Macro erhält.

Das Menüprogramm DPLSMAC führt einen Eingabedialog über Menü- und Prompteingaben bis zur Analyse und Ausgabe der Daten. Es steuert sowohl den Gebrauch der Programme DPLS und REDUN. Damit ist ein Wissen über die Bedienung dieser Macros nicht notwendig, es genügt dafür zu sorgen, daß sie im Verzeichnis von ISP zusammen mit DPLSMAC aufzufinden sind. Der Vorteil der Benutzung des Programmes DPLSMAC ist die Möglichkeit der Bedienung der beiden Analyseprogramme ohne Kenntnisse über den genauen Aufbau der geforderten Eingabematrizen. Diese werden durch den Eingabedialog vom Programm erstellt. Zudem ist es möglich, verschieden angeordnete Tabellen als Indikatormatrizen sowohl aus dem ISP-Arbeitsspeicher als auch von Datei zur Verfügung zu stellen, da diese vom Programm in die richtige Matrixgestalt gebracht werden.

Die Benutzung der Analysemacros ohne das Mantelprogramm DPLSMAC steht geübten Benutzern dennoch offen, da man diese in eigene Programme integrieren könnte (z.B. Simulationsmodelle) oder einfach eine schnellere Bedienung ermöglichen könnte.

3.2 Installation der Software

Zur Installation der Software ISP™ sollte das entsprechende Handbuch PC-ISP (1992) konsultiert werden. Für die Funktionalität der Macros müssen nur die drei Programme (DPLS.MAC, DPLSMAC.MAC, REDUN.MAC) in das jeweilige ISP-Verzeichnis kopiert werden (nicht in die Macrobibliothek, Änderungen vorbehalten). Ein Beispiel der Installation unter DOS wäre:

```
copy a:*.mac c:\isp
```

Da diese Programme nicht in der Makrobibliothek installiert sind, müssen sie beim Start von ISP™ aktiviert werden, wenn man eine DPLS-Analyse vornehmen möchte. Dazu braucht man nach dem Start von ISP™ nur folgende Zeile einzugeben:

```
run „dplsmac.mac“
```

Nun befindet sich die Programmsyntax im Arbeitsspeicher. Ein erneutes Laden ist erst beim Neustart von ISP™ nötig. Wenn man das Programm starten will, gibt man einfach dessen Namen ein:

```
dplsmac
```

Der Start der beiden Analysemacros wird durch das obige Macro selbständig ausgeführt.

3.3 Die Bedienung des Menüprogramms DPLSMAC

3.3.1 Ein einfaches dynamischen Modell als Beispiel

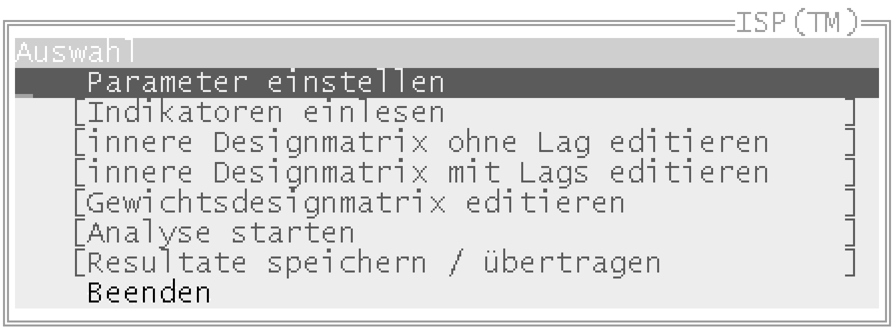
Die Bedienung sei hier kurz am Beispiel eines einfachen dynamischen Modells dargestellt:
Der Konsum als latente Variable hängt von den latenten Variabeln Einkommen und Vermögen direkt ab und weiterhin besteht eine einfach verzögerte Abhängigkeit vom Einkommen und von sich selbst (die Zeitreihen sind in der Datei INDIKAT.DAT zu finden und beziehen sich auf die Jahre 1968-1994 (108 Quartalsdaten). Der Einfachheit halber wird in der Demonstration auf Saisondummies verzichtet. Die Analyseergebnisse eines aufwendigeren Modells auf gleicher Datenbasis liegen vor.

Variablenblöcke:

Einkommen	**Vermögen**	**Konsum**
Bruttolöhne[1]	private Spareinlagen[2]	privater Konsum[3]
öffentliche Übertragungen[4]	inländ. festverzinsl. Wertpapiere[5]	staatlicher Konsum[6]
Einkünfte aus Unternehmert. und Verm.[7]	inländ. Aktienumlauf[8]	
verfügbare Einkommen[9]		

3.3.2 Die Dateneingabe

Zu Beginn (nach dem Informationstext) wird das Hauptmenü angezeigt, von dem vorerst nur die Punkte „Parameter einstellen“ und „Beenden“ auswählbar sind:



Man wählt durch Cursorbewegen (Pfeil hoch/runter) den Punkt „Parameter einstellen“ und bestätigt mit Enter.

Nun gibt man die Anzahl der Variablen und das Abbruchkriterium für die Iterationen (Rechengenauigkeit) als Anzahl der geforderten Nachkommastellen ein. Die folgenden Parameter sind für das Beispielmodell und für die beiliegende ASCII-Beispieldatei „INDIKAT.DAT“ einzustellen:

[1] VGR, Einkommen priv. Haushalte, Bruttolöhne und Gehälter (Mrd. DM), DBBK, Monatsberichte
[2] Spareinlagen von inländ. Nichtbanken (3 monatige Kündigungsfrist) alle Bankengruppen (Mrd. DM), DBBK
[3] privater Verbrauch in Preisen von 1991 (Mrd. DM), Statistisches Bundesamt
[4] VGR, Einkommen priv. Haushalte, öffentl. Übertragungen (Mrd. DM), DBBK, Monatsberichte
[5] Wertpapiere bei Kreditinstituten von Inländern, börsengängige Dividendenwerte insg. (Mrd. DM), DBBK
[6] staatlicher Verbrauch in Preisen von 1991 (Mrd. DM), Statistisches Bundesamt
[7] Einkommen aus Unternehmertätigkeit und Vermögen (Mrd. DM), DIW
[8] Aktien inländischer Emittenten, Gesamtumlauf am Ende des Berichtszeitraumes (Mrd. DM), DBBK
[9] VGR, Einkommen priv. Haushalte, verfügbares Einkommen (Mrd. DM), DBBK, Monatsberichte

```
Anzahl latenter Variablen: >3
Anzahl manifester Variablen: >9
Zeitreihenlänge:>108
Abbruchkriterium (Nachkommastellen der Veränderg.): >5_
```

Dem Programm liegt die ASCII-Datei "INDIKAT.DAT" bei.

Nun werden im Hauptmenü bis auf die Punkte „Analyse starten“ und „Resultate speichern“ alle Punkte anwählbar (erkennbar an der eckigen Klammer bei den nichtwählbaren). Es ist also jetzt möglich die Designmatrizen zu editieren (sowohl für die Blockstruktur als auch für die unverzögerten und verzögerten Abhängigkeiten zwischen den latenten Variablen).

Zuerst sollte aber der Punkt „Indikatoren einlesen“ ausgewählt werden. Die Matrix der manifesten Variablen wird damit eingelesen.

Für das Matrixformat kann man wählen zwischen der Variablenanordnung über die Zeilen oder die Spalten der Tabelle:

```
ISP(TM)
Tabellenformat der Manifesten
   Zeilen=Variablen, Spalten=Beobachtungen
   Zeilen=Beobachtungen, Spalten=Variablen
```

Im vorliegenden Beispiel ist der zweite Menüpunkt zu wählen, da die Spalten die Variablen repräsentieren.

Nun entscheidet ein ähnlich aussehendes Menü über die Wahl, ob die Daten von Diskette/Festplatte oder aus dem ISP-Arbeitsspeicher gelesen werden sollen. Danach gibt man entweder den Dateinamen in Anführungszeichen (z.B.: ’indikat.dat’) oder den ISP-Objektnamen (z.B.: indikat) ein. Die Dimensionen werden aus den zum Anfang eingegebenen Werten bestimmt. Eine zu kleine oder zu große Datei wird dann also entweder mit Missing Values gefüllt oder beschnitten eingeladen. Man muß den Namen des ISP-Objektes (oder der Datei in Anführungsstrichen) fehlerfrei eingeben, da das Macro sonst fehlerhaft durchgeführt wird. In diesem Fall wiederholt man die Eingabe durch erneutes Aufrufen des entsprechenden Menüpunktes vom Hauptmenü aus.

Für das Beispiel sollte nun vom Datenträger eingelesen werden, indem man nach der Aufforderung in Anführungszeichen den Dateinamen eingibt (eventuell angeführt von Laufwerk und Verzeichnis), z.B.:

```
"a:\indikat.dat"
```

Nach dem fehlerfreien Einlesen der Indikatoren editiert man die Designmatrizen über ein Menü, in welchem sämtliche latenten Variablen angezeigt werden. Der Benutzer kann über die Leertaste die Variablen markieren (nur in der Studentenversion von ISP, sonst die Cursor-Taste: nach rechts), von dem die in der Titelzeile des Menüs gezeigte latente Variable direkt abhängen soll. Dieses Menü erscheint deswegen auch für jede latente Variable neu. Eine Markierung zeigt die Postulierung einer Abhängigkeit der aktuellen Variable von der markierten Variable. Eine Abhängigkeit der dargestellten Variable von sich selbst ist nur bei den Lag-beziehungen erlaubt.

```
mark                                  ISP(TM)
3. lat.Var.hängt ab von: ( -> oder SPACE)
    latente Variable 1
    latente Variable 2
   [latente Variable 3                     ]
```

Im Beispiel kann also mit der ENTER-Taste durchgeschaltet werden, bis im Titel die 3.latente Variable erscheint. Dann bewegt man den Cursor auf den ersten Eintrag und drückt die CURSOR-NACH-RECHTS (->) - Taste (Leertaste in der Studentenversion), um die Abhängigkeit von der ersten und genauso für die zweiten Variable darzustellen.

Bei der 3. Variable die Abhängigkeit von der ersten und der zweiten Variable markieren.

Ebenso verfährt man beim Editieren der verzögerten Abhängigkeiten. Hier ist eine Markierung der jeweils editierten Variablen von sich selbst ein Zeichen für das Postulieren von Autoregression, sonst entspricht es der Annahme einer einfach verzögerten Abhängigkeit von der markierten Variable.

Bei der 3. Variable die Abhängigkeit von der ersten und von sich selbst markieren.

Weiterhin wird nun nach obigem Schema die Zugehörigkeit der manifesten Variablen zu den einzelnen Blöcken editiert.

Zum ersten Block gehören im Beispiel die ersten 4 Variablen (markieren mit -> bzw. Space), zum zweiten Block gehören die folgenden 3 (5.,6. und 7. Variable) und zum dritten Block gehören die letzten beiden Variablen.

Da es im Beispiel drei latente Variablen gibt, wird dieses Menü drei mal dargestellt um drei Blöcke zu erstellen. Die Indikatoren können wie gewünscht in jeder Reihenfolge für jeden Block markiert werden.

Nach dieser Eingabe erfolgt die Analyse der Daten durch den automatischen Aufruf des Macros DPLS. Man muß nun einige Iterationen abwarten.

3.3.3 Die Datenausgabe

Betätigt man den Menüpunkt "RESULTATE SPEICHERN", dann wird die Ausgabdatei "DPLS_OUT.TXT" angelegt und eine Eingabe (in Anführungszeichen eingeschlossenen Zeichenkette) für eine Anmerkung zu Beginn der Ausgabedatei gefordert. Diese Datei kann man folgend am Bildschirm betrachten. Sie stellt sowohl die erstellten Designmatrizen als auch die Analyseresultate (Ladungen, Gewichte, Pfadkoeffizienten) dar. Die latenten Variablen können nur in den Arbeitsspeicher von ISP übernommen werden. Zu beachten ist, daß die Datei DPLS_OUT.TXT immer im aktuellen ISP-Verzeichnis gespeichert wird und die eventuell schon vorhandene Datei einfach überschreibt. Sie sollte also vorher gesichert werden. Wenn man aus ISP heraus sichern will, kann man zum Beipiel mit dem Befehl „cli“ DOS-Kommandos ausführen:

```
cli „copy c:\isp\dpls_out.txt d:\dpls\mod1\modell.txt“
```

Die Ausgaberesultate sind als Matrizen zu interpretieren. Bei den Ladungen und den Gewichten zeigen die Zeilen die aktuelle manifeste Variable und jede Spalte weist auf eine latente Variable.

Bei den Pfadkoeffizienten weist die Anzahl der Zeilen/Spalten die Zahl der latenten Variablen hin (im Beispiel 3 Zeilen, wobei nur die 3. Variable endogen ist und somit Koeffizienten enthält). Im Beispiel hat der 1. Block 4 manifeste Variablen und deshalb enthält die Ladungs-/Gewichtsmatrix in der ersten Spalte genau 4 von Null verschiedene Werte:

```
      T | | | | | | | | | | B          PgUp ↑ ↓ PgDn
Gewichte W:
    .41036660E-02     .00000000         .00000000
    .12524030E-02     .00000000         .00000000
   -.69494910E-04     .00000000         .00000000
    .58110030E-02     .00000000         .00000000
    .00000000         .15399120E-01     .00000000
    .00000000         .21494570E-02     .00000000
    .00000000         .70098510E-02     .00000000
    .00000000         .00000000         .16649950E-01
    .00000000         .00000000         .46897620E-02

Ladungen:
    78.476340         .00000000         .00000000
    23.739080         .00000000         .00000000
   -3.2590020         .00000000         .00000000
    111.51290         .00000000         .00000000
    .00000000         54.125020         .00000000
    .00000000         6.5044240         .00000000
    .00000000         21.761000         .00000000
    .00000000         .00000000         55.625100
    .00000000         .00000000         15.746000

Pfadkoeff. (3*3-Matrix (ohne Lagbez.):
browse: PgDn/Up, keep: Ins, else press return>_
```

Die komplette Ausgabedatei DPLS_OUT.TXT sieht im Beispiel folgendermaßen aus:

Dateiname: DPLS_OUT.TXT
Testmodell

Dynamische PLS-Analyse, Ausgabe der Resultate:

Designmatrix:

0.00000000E+00	0.00000000E+00	0.00000000E+00
0.00000000E+00	0.00000000E+00	0.00000000E+00
1.0000000	1.0000000	0.00000000E+00

Designmatrix der Lagbeziehungen:

0.00000000E+00	0.00000000E+00	0.00000000E+00
0.00000000E+00	0.00000000E+00	0.00000000E+00
1.0000000	0.00000000E+00	1.0000000

Designmatrix der Blockbeziehungen:

1.0000000	0.00000000E+00	0.00000000E+00
1.0000000	0.00000000E+00	0.00000000E+00
1.0000000	0.00000000E+00	0.00000000E+00
1.0000000	0.00000000E+00	0.00000000E+00
0.00000000E+00	1.0000000	0.00000000E+00
0.00000000E+00	1.0000000	0.00000000E+00
0.00000000E+00	1.0000000	0.00000000E+00
0.00000000E+00	0.00000000E+00	1.0000000
0.00000000E+00	0.00000000E+00	1.0000000

Rechengenauigkeit:
5.0000000

Anzahl der Beobachtungen:
108.00000

Gewichte W:

0.41036662E-02	0.00000000E+00	0.00000000E+00
0.12524028E-02	0.00000000E+00	0.00000000E+00
-0.69494934E-04	0.00000000E+00	0.00000000E+00
0.58110030E-02	0.00000000E+00	0.00000000E+00
0.00000000E+00	0.15399122E-01	0.00000000E+00
0.00000000E+00	0.21494566E-02	0.00000000E+00
0.00000000E+00	0.70098508E-02	0.00000000E+00
0.00000000E+00	0.00000000E+00	0.16649947E-01
0.00000000E+00	0.00000000E+00	0.46897624E-02

Ladungen:

78.476326	0.00000000E+00	0.00000000E+00
23.739075	0.00000000E+00	0.00000000E+00
-3.2590008	0.00000000E+00	0.00000000E+00
111.51289	0.00000000E+00	0.00000000E+00
0.00000000E+00	54.125019	0.00000000E+00
0.00000000E+00	6.5044236	0.00000000E+00
0.00000000E+00	21.761000	0.00000000E+00
0.00000000E+00	0.00000000E+00	55.625099
0.00000000E+00	0.00000000E+00	15.745996

Pfadkoeff. (3*3-Matrix (ohne Lagbez.):

0.00000000E+00	0.00000000E+00	0.00000000E+00
0.00000000E+00	0.00000000E+00	0.00000000E+00
1.3390000	0.17999999E-01	0.00000000E+00

Pfadkoeff. (3*3-Matrix (mit Lagbez.):

0.00000000E+00	0.00000000E+00	0.00000000E+00
0.00000000E+00	0.00000000E+00	0.00000000E+00
-0.80400002	0.00000000E+00	0.42699999

Redundanz:
0.88102752

Redundanzvektor:
0.00000000E+00
0.00000000E+00
0.00000000E+00
0.00000000E+00
0.00000000E+00
0.00000000E+00
0.00000000E+00
0.90810114
0.85395390

Anzahl der Iterationen:
8.0000000

3.4 Die Analysemacros, ihre Syntax und ihre Funktionalität

3.4.1 Das Analysemacro DPLS

Die Funktion des Analysemacros DPLS

Das Macro DPLS ist das Kernstück des Programms und enthält die Routinen für die Analyseiterationen. Es werden die Gewichte, Ladungen, latenten Variablen und Pfadkoeffizienten ermittelt. Das Programm liegt als eigenständiges Modul vor und kann damit vielseitiger eingesetzt werden. Es funktioniert wie alle üblichen ISP-Befehle durch Kommandoeingabe als Prozedur und verlangt ISP-Objektnamen zur Eingabe der Daten und weitere ISP-Objektnamen für die Ausgabe der Resultate in den Arbeitsspeicher von ISP.

Für den Anwender, der nur wenige Analysen vornehmen möchte und wenig Zeit hat, sich mit den Datenformaten der Eingabematrizen zu beschäftigen, ist es sinnvoll, das im vorigen Abschnitt erwähnte Mantelprogramm für Ein- und Ausgabe zu benutzen. Für Anwender, die mehrfach Analysen oder Simulationen durchführen wollen und somit naturgemäß öfter das Macro aufrufen, kann es durchaus sinnvoll sein, sich mit den Matrizenformaten und der Syntax dieses Macros zu beschäftigen.

Die Syntax des Analysemacros DPLS

Die Syntax des Programmaufrufs sieht folgendermaßen aus:

```
Syntax:     dpls w d dy dl y genau > w b pk lt iter

Eingabe:
            w = Startwerte für Parametermatrix
            d = Designmatrix der unverzögerten Beziehungen der latenten Variablen
           dy = Designmatrix der Blockzugehörigkeit der manifeste Variablen
           dl = Designmatrix der verzögerten Beziehungen der latenten Variablen
            y = Manifeste Variablen
        genau = Abbruchkriterium (Zahl der Nachkommastellen)

Ausgabe:
            w = Gewichtsmatrix
            b = Ladungsmatrix
           pk = Pfadkoeffizienten
           lt = Latente Variablen
         iter = Anzahl an benötigten Iterationen
```

Die Designmatrix (unverzögerte Beziehungen bei den latenten Variablen) könnte dabei folgende Form haben (Zweiblockmodell mit einer Abhängigkeit), wobei für das Setzen der 1 gilt:

$$\begin{array}{lc} \begin{array}{l}\text{1. Variable:}\\ \text{2. Variable:}\end{array} & \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{1} & \mathbf{0} \end{pmatrix} \\ & \begin{array}{cc} \begin{array}{l}\text{direkte, lineare}\\ \text{Abhängigkeit von}\\ \text{1. Variable}\end{array} & \begin{array}{l}\text{direkte lineare}\\ \text{Abhängigkeit von}\\ \text{2. Variable}\end{array} \end{array} \end{array}$$

Das Setzen einer 1 innerhalb der Matrixdiagonale wäre logisch falsch, da keine Abhängigkeit der Variable von sich selbst analysiert wird. Anders ist es bei der Designmatrix, die die verzögerten Variablenbeziehungen kodiert. Ihre Dimension ist identisch, nur würde eine 1 in der Diagonale den autoregressiven Beziehungen entsprechen.

Eine mögliche Version in einem Zweiblock-Modell wäre (einfach verzögerte Abhängigkeit der zweiten latenten Variable von der ersten und von sich selbst):

$$\begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix}$$

Außerdem gibt es eine Designmatrix für die Anordnung der manifesten Variablen innerhalb der Blöcke. In einem Zweiblockmodell gibt es 2 latente Variablen (2 Spalten) und es könnte beispielsweise 8 manifeste Variablen geben (8 Zeilen), wobei zu jedem Block vielleicht 4 manifeste Variablen gehören:

$$\begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$$

Das Setzen der 1 bestimmt, ob die der Zeilennummer entsprechende manifeste Variable zu dem durch die Spaltennummer identifizierten Block gehört.

Identisch ist die Startmatrix für die Gewichte ***w*** dimensioniert, nur daß die Matrixzellen, die in der letzten Matrix mit einer 1 gefüllt waren, durch die gewünschten Startwerte ersetzt werden. In vielen Fällen wählt man der Einfachheit halber auch dort die Zahl 1, womit diese Matrix völlig identisch zur Blockdesignmatrix wird.

Die manifesten Variablen sollen in einer Matrix vorliegen, in welcher die Zeilen die Variablen darstellen und die Spalten die verschiedenen Beobachtungen. Die Anzahl der Zeilen sollte also mit der Anzahl der Zeilen der Blockdesignmatrix oder der Gewichtsstartwertematrix übereinstimmen. Die Zentrierung, das heißt die Subtraktion der Mittelwerte über die Zeit, erfolgt im Programm.

Zum Schluß gibt man noch das Abbruchktriterium ein, welches durch eine Zahl respräsentiert wird, die die Nachkommastellen darstellt, auf die die Änderungen der Gewichte nach jeder Iteration geprüft werden.

Für die Ausgabe der Daten muß man lediglich die Bezeichner der gewünschten ISP-Objektnamen nennen, in denen die geschätzten Gewichte und die Ladungen in dem oben genannten Formaten gespeichert werden. Auch die Pfadkoeffizienten werden als Matrix ausgegeben, wobei die Zeilen den latenten Variablen entsprechen und die Spalten latenten Variable enthalten, von denen die jeweilige Zeilenvariable mit dem Koeffizienten im Kreuzungspunkt abhängt. Die Ergebnismatrix ist eine Verkettung der unverzögerten und verzögerteten Pfadkoeffizientenmatrix.

Weiterhin werden die geschätzten Werte der latenten Variablen ausgegeben, wobei die Variablen wieder über die Zeilen und die Beobachtungen über die Spalten zu lesen sind. Der letzte Wert enthält die Anzahl der Iterationen, die das Verfahren bis zur Konvergenz benötigte.

3.4.2 Das Analysemacro REDUN

Die Funktion des Analysemacros REDUN

Das Macro REDUN ist eine Ergänzung des Programms DPLS und enthält die Routinen für die Berechnung des Redundanzwertes der kompletten Analyse wie auch der Redundanzwerte der einzelnen endogenen latenten Variablen. Das Programm liegt als eigenständiges Modul vor und kann damit vielseitiger eingesetzt werden. Es funktioniert wie alle üblichen ISP-Befehle durch Kommandoeingabe als Prozedur und verlangt ISP-Objektnamen zur Eingabe der Daten und weitere ISP-Objektnamen für die Ausgabe der Resultate in den Arbeitsspeicher von ISP.

Für den Anwender, der nur wenige Analysen vornehmen möchte und wenig Zeit hat, sich mit den Datenformaten der Eingabematrizen zu beschäftigen, ist es sinnvoll, das im vorigen Abschnitt erwähnte Mantelprogramm für Ein- und Ausgabe zu benutzen. Für Anwender, die mehrfach Analysen oder Simulationen durchführen wollen und somit naturgemäß öfter das Macro aufrufen, kann es durchaus sinnvoll sein, sich mit den Matrizenformaten und der Syntax dieses Macros zu beschäftigen.

Die Syntax des Analysemacros REDUN

Die Syntax des Programmaufrufs sieht folgendermaßen aus:

Syntax: redun b pk t dl y > red redv

Eingabe:
b (Ladungen)
pk (Pfadkoeff.)
t (latente Variablen; Zeilen = Variable)
dl (Designmatrix verzögerten Beziehungen der latenten Variablen)
y (manifeste Variablen; Zeilen = Variable)

Ausgabe:
red (Redundanz)
redv (Einzelredundanzvektor)

Für die Gestalt und den Inhalt der Eingabematrizen gilt hier das gleiche wie im Abschnitt über die Syntax von DPLS. Die Redundanz hat einen Wert zwischen 0 und 1 und der Redundanzvektor enthält soviel von Null verschiedene Werte, wie endogene manifeste Variablen existieren und stellt die Redundanzen dieser Variablen dar.

4. Eine Simulationsstudie

Auf der Grundlage dieses Programms wurde die folgende Reihe von Simulationen durchgeführt. Mit ihrer Hilfe soll das Verhalten des Schätzverfahrens bei verschiedenen Fehlerverteilungen geprüft werden. Der Einfachheit halber wird das Eingleichungsmodell aus Abschnitt 2 mit drei latenten Variablen benutzt.

Die Ladungen, Gewichte, unverzögerten und verzögerten Pfadkoeffizienten werden in Matrixform auf folgende Werte gesetzt:

$$P = \begin{bmatrix} 0.3 & 0 & 0 \\ 0.6 & 0 & 0 \\ 0.5 & 0 & 0 \\ 0 & 0.2 & 0 \\ 0 & 0.4 & 0 \\ 0 & 0.6 & 0 \\ 0 & 0 & 0.9 \\ 0 & 0 & 0.3 \\ 0 & 0 & 0.1 \end{bmatrix}, \quad W = \begin{bmatrix} 0.434 & 0 & 0 \\ 0.867 & 0 & 0 \\ 0.723 & 0 & 0 \\ 0 & 0.363 & 0 \\ 0 & 0.726 & 0 \\ 0 & 1.089 & 0 \\ 0 & 0 & 0.997 \\ 0 & 0 & 0.332 \\ 0 & 0 & 0.111 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0.189 & 0.534 & 0 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0.237 & 0.062 & 0.1 \end{bmatrix}$$

Die dazugehörenden Designmatrizen $\mathbf{D}_W$, $\mathbf{D}_B$ und $\mathbf{D}_C$ haben die gleiche Struktur mit Einsen an den von Null verschiedenen Stellen.

Abbildung 2 zeigt das Pfadmodell mit den durch Parameter gekennzeichneten Pfaden. Die gestrichelten Pfade bezeichnen zeitverzögerte Abhängigkeiten.

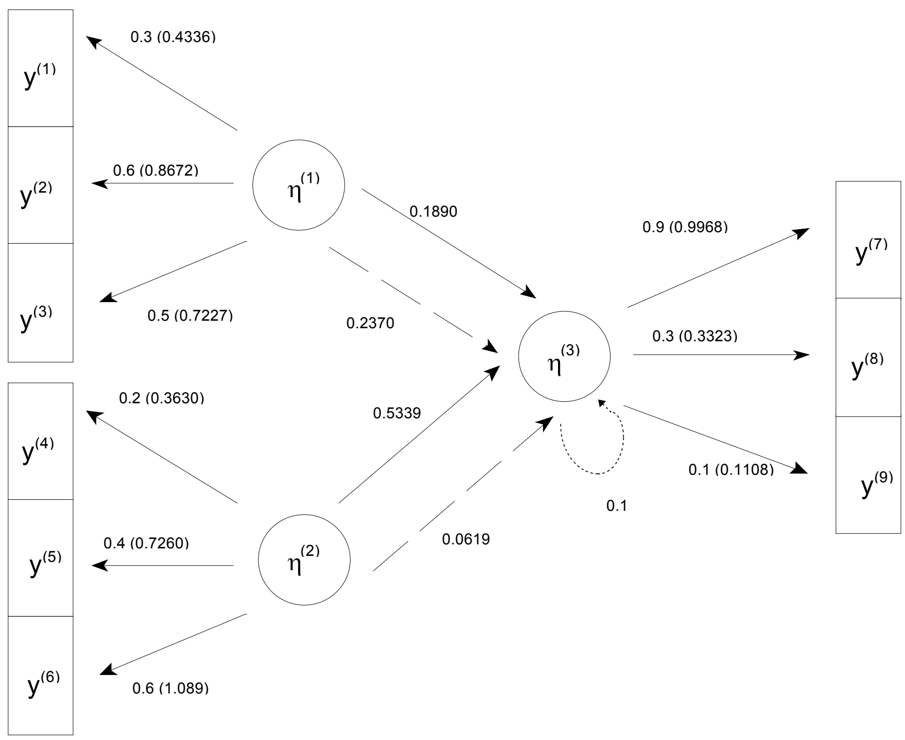


Abb. 2: Drei-Block-Modell mit Parametervorgaben

Im ersten Teil der Studie werden normalverteilte Fehler angenommen. Ihre Varianzen variieren im inneren Modell zwischen 0 und 0,1 und im äußeren Modell zwischen 0 und 0,5. Damit wird die Wirkung der Modellfehlervarianz auf die Parameterschätzung, insbesondere ihre Verzerrung illustriert.

Der zweite Teil simuliert ein Modell mit **$**-verteiltem Fehler variierender Schiefe von stark linkssteil über symmetrisch bis stark rechtsteil. Hier soll der Einfluß der Schiefe auf die Schätzung der Parameter untersucht werden.

Jeder Parameterkombination entspricht ein eigenes DPLS-Modell. Auf diese Weise wurden mehrere hundert Modelle analysiert. Mit jedem Modell wurden die Zeitreihen der manifesten Variablen 500 mal auf einer Länge von je 200 Datenpunkten mit dem ISP-Zufallszahlengenerator erzeugt.

So standen für jede Modellversion 500 Replikationen der Datenmatrix für die Parameterschätzung zur Verfügung. Im Rahmen der Programmumgebung PC-ISP/DGS™ wurden mit dem Macro-System DPLS für jede Datenmatrix und jede Modellversion Ladungen, Gewichte, unverzögerte und verzögerte Pfadkoeffizienten berechnet. Für jedes Modell, also für jede Gruppe von 500 Parameterschätzungen wurden dann die Mittelwerte der Schätzungen, ihre Standardfehler und ihre Verzerrung berechnet.

Einige Ergebnisse wurden für die einzelnen Modelle, das heißt über den variierenden Parametern graphisch aufgetragen. Die Abbilungen 3 und 4 zeigen die Schätzungen für variierende Varianzen normalverteilter Fehler des inneren bzw. äußeren Modells. Abbildung 5 stellt die Schätzungen für **$**-verteilte Fehler dar.

Die wichtigsten Ergebnisse, die man aus diesen Graphiken ablesen kann sind die folgenden:

- Die geschätzten durchschnittlichen Pfadkoeffizienten variieren nur unwesentlich mit veränderlichen inneren Fehlervarianzen. (Abb. 3 und 4 oben).
- Die Standardfehler der Schätzungen wachsen fast linear mit den Modellfehlervarianzen (Abb. 3 und 4 Mitte).
- Die Beträge der Verzerrungen der geschätzten verzögerten Pfadkoeffizienten fallen mit zunehenden inneren Modellfehlern, während die Verzerrungen der unverzögerten Pfadkoeffizienten keinen eindeutigen Trend zeigen (Abb. 3 unten).
- Die Beträge der Verzerrungen der Pfadkoeffizienten nehmen mit wachsender äußerer Fehlervarianz zu (Abb. 4 unten).
- Es ist keine klare Tendenz der Über- oder Unterschätzung von Pfadkoeffizienten durch DPLS zu erkennen (Abb. 3 und 4 oben und unten).
- Die Schiefe einer asymmetrischen Fehlerverteilung hat keinen unmittelbaren Einfluß auf die Verzerrung der Pfadkoeffizientenschätzung (Abb. 5 unten).

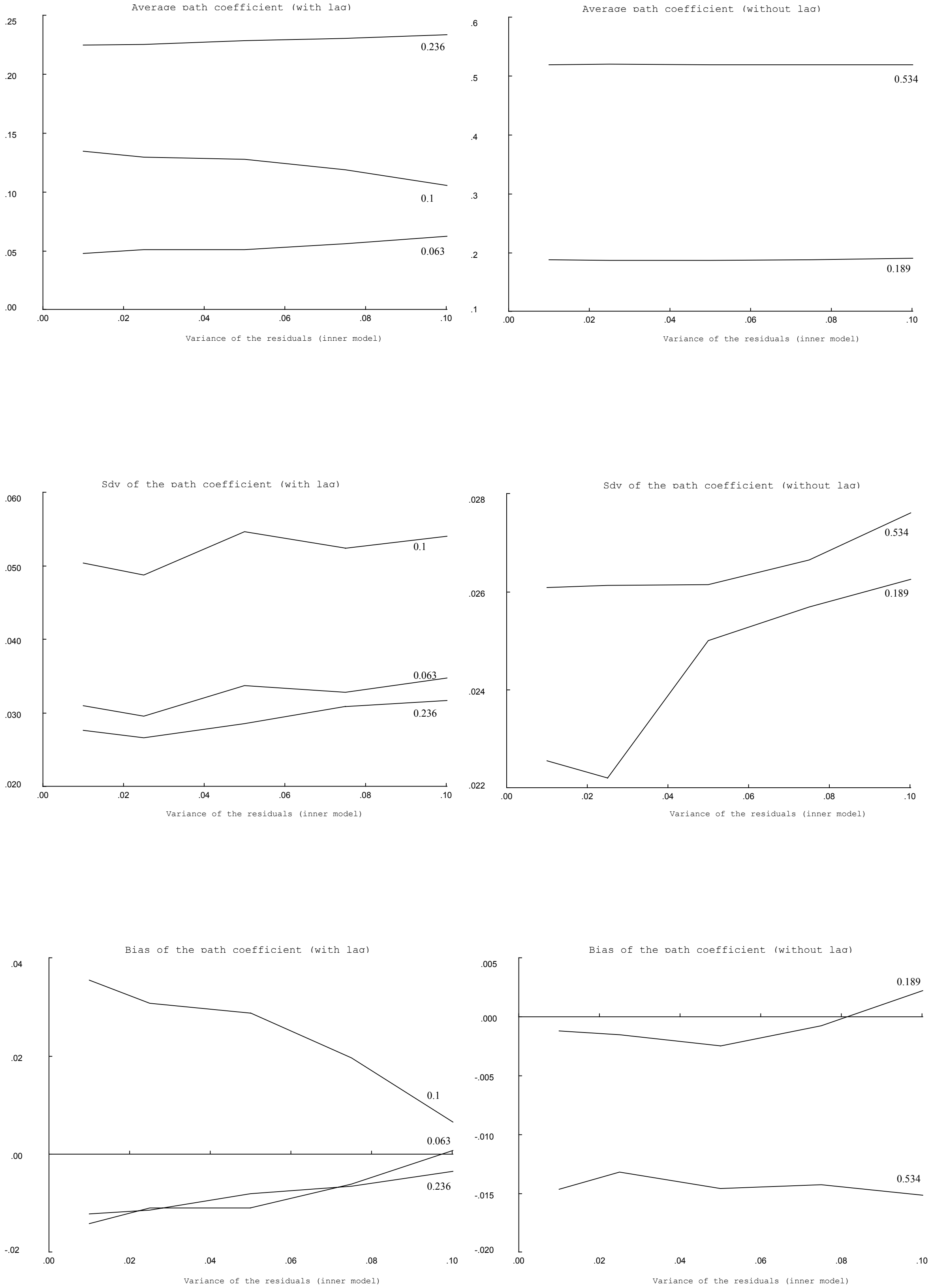


Abb. 3: Ergebnisse der Simulation 1

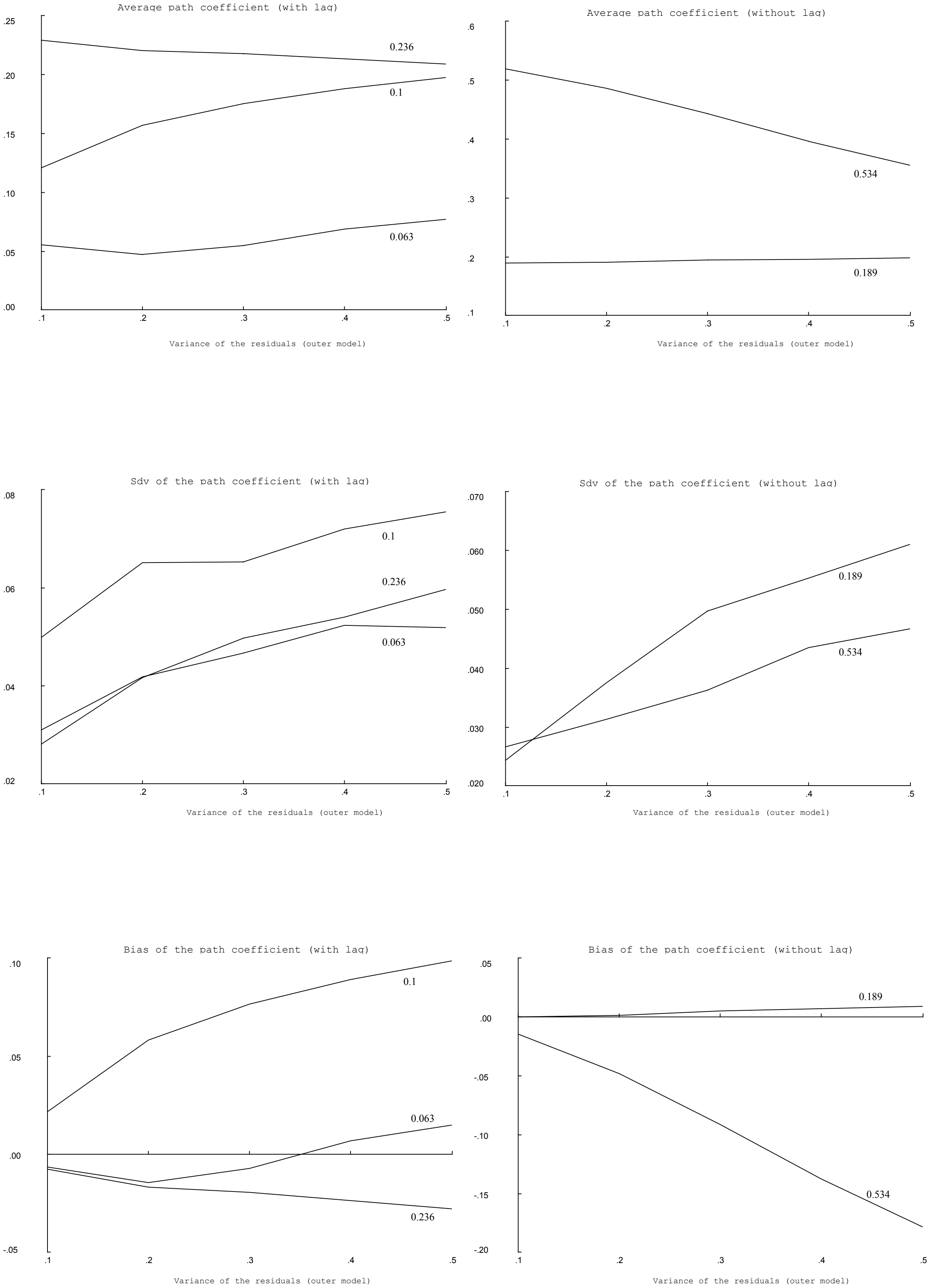


Abb. 4: Ergebnisse der Simulation 2

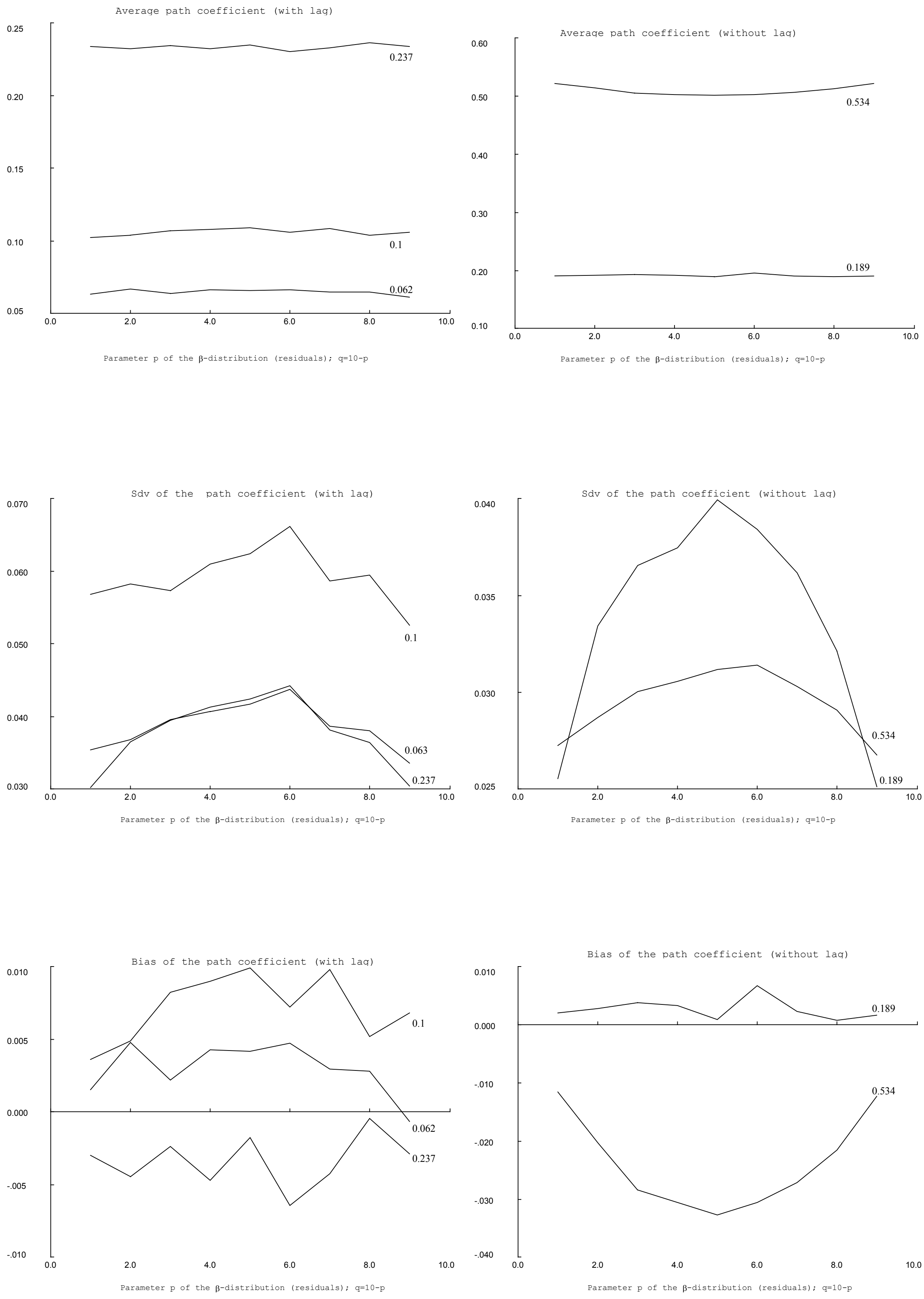


Abb. 5: Ergebnisse der Simulation 3

5. Schlußbemerkung

DPLS (Partial Least Squares für dynamische Pfadmodelle mit latenten Variablen) ist ein vielversprechendes Instrument der dynamischen Modellierung mit Blöcken gleichartiger Variablen, wenn Verteilungsvorausetzungen, wie für Verfahren auf Maximum-Likelihood-Basis erforderlich, nicht erfüllt sind. Das sonst in solchen Fällen sehr nützliche PLS von Wold, insbesondere das i.a. äußerst verdienstvolle Programm LVPLS von Lohmöller, liefern im Fall des gleichzeitigen Auftretens verzögerter und unverzögerter latenter Variablen unbefriedigende Ergebnisse. Der vorläufige Mangel an Prüfmöglichkeiten mit DPLS wird durch die Möglichkeit der Prognose und der Modellvalidierung mit dem Redundanzmaß kompensiert.

Das Progamm DPLS ist durch seine Einbindung in die bewährte Statistik-Umgebung PC-ISP/DGS™ zur Verarbeitung von mit ISP vorbehandelten Daten sowie zur Weitergabe seiner Ergebnisse an ISP zwecks Weiterverarbeitung und dynamischer graphischer Darstellung hervorragend geeignet. Die Eingabe von Daten, Steuerparametern und Modellstruktur für DPLS ist durch seine Menu-Steuerung einfach und nutzerfreundlich. Ein Prognoseprogramm zu DPLS ist in Vorbereitung.

Die Simulationsbeispiele zeigen, daß DPLS in aller Regel schnell kovergiert und daß die Schätzergebnisse nicht mit auffälligen Unschärfen oder Verzerrungen belastet sind.

Die Beispiele können noch nicht die Wirkung des Verfahrens bei großer Anzahl von manifesten Variablen in den Blöcken zeigen, wie sie in dynamischen Finanz- und Kapitalmarktmodellen zu erwarten sind. Hier ist aber wegen der Parametersparsamkeit des inneren Modells mit dem breitesten Anwendungsgebiet zu rechnen.

Literatur

Geppert, F. (1996): Bearbeitung, Programmierung, Simulation und Anwendung eines PLS-Algorithmus für einfache dynamische Modelle mit latenten Variablen. Diplomarbeit unter Betreuung von H. G. Strohe, Universität Potsdam.

Jöreskog, K.G. / Sörbom, D. (1987): LISREL VII Program Manual. International Educational Services, Chicago.

Lohmöller, J.-B. (1984): LVPLS 1.6 - Program Manual (Latent Variables Path Analysis with Partial Least Squares Estimation). Zentralarchiv für empirische Sozialforschung, Universität Köln.

Lohmöller, J.-B. (1989): Latent Variable Path Modelling with Partial Least Squares. Heidelberg.

Mathes, H. (1993): Der PLS-Ansatz für die Analyse von Pfadmodellen; Mathematical Systems in Economics. Anton Hain, Frankfurt/Main.

PC-ISP (1992): Users Guide and Command Descriptions, Datavision AG, Schweiz.

Schneeweiß, H. (1993): Consistency at Large in Models with Latent Variables; in: Haagen / Bartholomew / Deistler (eds): Statistical Modelling and Latent Variables, Elsevier, p. 299-320.

Strohe, H.G. (1995): Dynamic Latent Variables Path Models - An Alternative PLS Estimation. Statistische Diskussionsbeiträge Nr. 1, Universität Potsdam.

Strohe, H.G. (1996): A Dynamic Partial Least Squares Approach to Macroeconomic Modelling; in: Welfe / Wdowinski (eds): Macromodels ´96 on Integration and Development, Lodz, p. 99-119.

Wold, H. (1973): Nonlinear Iterative Partial Least Squares (NIPALS) Modelling - Some Current Developement; in P.R. Krishnajah (Ed.), Multivariate Analysis (Vol. 3, p. 383-407), New York; Academic Press.

Anhang

Programmlistings

DPLSMAC

```
###############################################################################
# Dynamische PLS-Analyse Eingabeprogramm                          #
# Universität Potsdam                                  #
#                                                #
#      Grundsystem                          01.09.1995      #
#      Hauptmenue eingerichtet am                   01.03.1996     #
#      letzte Aenderung am                         16.01.1997     #
#                                                #
# Frank Geppert                                       #
#                                                #
# Syntax:       dplsmac                                #
#                                                #
# Eingabe:                                          #
#                Anzahl latenter Variablen                     #
#                Anzahl manifester Variablen                    #
#                Anzahl der Beobachtungen                      #
#                Rechengenauigkeit (Stellen hinter Komma)           #
#           ( weitere Eingaben je nach Dialog)                  #
#                                                #
# Ausgabe:                                          #
#            ( je nach Dialog )                            #
#                                                #
# Hinweis:                                          #
#      funktioniert nur bei existierendem DPLS.MAC (im ISP-Verzeichnis) #
#                und bei existierendem REDUN.MAC (im ISP-Verz.)      #
###############################################################################
input /macro /*termin=n > dplsmac
set *erract=n
local i j a aglue ans ansglue main
local d dy dl w n tab s wout lout skout ltout iter
local suc1 suc2 suc3 suc4 suc5 suc6 suc7 suc8 enabled
local red redm skn skl fmy fmy1 dim tit
############################################ Erfolgreich setzen
suc1=0
suc2=0
suc3=0
suc4=0
suc5=0
suc6=0
suc7=0
suc8=0
############################################ MainMenu erzeugen
local a1 a2 a3 a4 a5 a6 a7 a8 menuglue men_able
cli "cls"
putstr ""
putstr "DDD    PPP   LL     SS"
putstr "D  D   P  P  LL    S"
putstr "D  D   PPP   LL     SS"
putstr "D  D   P     LL       S"
```

```
putstr "DDD    P     LLLL   SS"
putstr ""
putstr "Dynamische Partial Least Squares Modellierung"
putstr "--------------------------------------------"
putstr "Universität Postdam"
putstr "Wirtschafts- und Sozialwissenschaftlische Fakultät"
putstr "Lehrstuhl für Statistik und ™Ökonometrie"
putstr "Verfahren: Prof. Dr. H. G. Strohe"
putstr "Programm.: Frank Geppert"
putstr ""
putstr ""
print  "Weiter mit ENTER oder INS"
a1=stoa('Parameter einstellen                 ')
a2=stoa('Indikatoren einlesen                 ')
a3=stoa('innere Designmatrix ohne Lag editieren')
a4=stoa('innere Designmatrix mit Lags editieren')
a5=stoa('Gewichtsdesignmatrix editieren        ')
a6=stoa('Analyse starten                      ')
a7=stoa('Resultate speichern / übertragen      ')
a8=stoa('Beenden                              ')
glue a1 a2 a3 a4 a5 a6 a7 a8 > aglue /axis=2
menuglue=trn(aglue)
main=1
while (main != 8) # Menü weiterhin aufrufen
men_able=array(8) # Menüpunkte ausschalten, bei Sinnlosigkeit des Aufrufs
men_able(1 8)=1
if (suc1==1) then
 men_able(2:5)=1
else
 men_able(2:5)=0
end if
if (suc1==1) then
 if (suc2==1) then
  if (suc3==1) then
   if (suc4==1) then
    if (suc5==1) then
     men_able(6)=1
    end if
   end if
  end if
 end if
else
 men_able(6)=0
end if
if (suc6==1) then
 men_able(7)=1
else
 men_able(7)=0
end if
menu menuglue men_able /title='Auswahl' > main
if (main==1) then
 ################################### Parameter einstellen ##
 getarg "Anzahl latenter Variablen: " > l
 getarg "Anzahl manifester Variablen: " > m
 getarg "Zeitreihenlänge:" > n
 getarg "Abbruchkriterium (Nachkommastellen der Veränderg.): " > genau
 if (l<=0) or (genau<=0) or (m<=0) then
  cli 'cls'
  print 'positive Werte eingeben'
```

```
  suc1=0
 else
  suc1=1
 end if
 if (m<l) then
  cli 'cls'
  Print 'Es dürfen nicht mehr latente als manifeste Variablen existieren'
  suc1=0
 else
  suc1=1
 end if
end if
if (main==2) then
 ################################### Einlesen ###
 if (suc1==1) then
  aglue=stoa('Zeilen=Variablen, Spalten=Beobachtungen')
  a    =stoa('Zeilen=Beobachtungen, Spalten=Variablen')
  glue aglue a > aglue /axis=2
  aglue=trn(aglue)
  menu aglue /title='Tabellenformat der Manifesten' > tab
  aglue=stoa('vom Datenträger einlesen        ')
  a=    stoa('aus dem Arbeitsspeicher einlesen')
  glue aglue a > aglue /axis=2
  aglue=trn(aglue)
  menu aglue /title='Manifeste Variablen einlesen' > ans
  if (ans==1) then
   local y
   getarg "Dateiname in Anführungszeichen (evtl. mit Pfad)" > file
   if (tab==1) then
    s='$(m)%,$(n)'
    input file > y /dims=$s
   else
    s='$(n)%,$(m)'
    input file > y /dims=$s
    y=trn(y)
   end if
  else
   getarg "Manifeste Variablen: " > y
   if (tab==1) then
   else
    y=trn(y)
   end if
  end if
 suc2=1
 print (dims(y)) y
 else
  cli 'cls'
  Print 'Bitte vorerst Parameter korrekt eingeben'
 end if
end if
if (main==3) then
  ############################# Designmatrix erzeugen
 if (suc1==1) then
  ansglue=array($(l))
  do i=iota(array($(l-1))+1)
   aglue=stoa('latente Variable 0')
   enabled=array($(l))
   do j=iota(array($(l)))
    a=stoa('latente Variable $(j)')
```

```
    glue aglue a > aglue /axis=2
    if (j <= i) then
     enabled($(j))=1
    else
     enabled($(j))=0
    end if
   end do
   aglue=trn(aglue(*,2:$(l+1)))
   set title='$(i+1)%. lat.Var.hängt ab von: ( -> oder SPACE)'
   menu aglue enabled /mark=y > ans
   glue ansglue ans /axis=2 > ansglue
  end do
  ansglue=and(or(ansglue,trn(ansglue)),trn(ansglue))
  cli 'cls'
  print ansglue
  putstr ""
  d=ansglue
 suc3=1
 else
  cli 'cls'
  Print 'Parameter bitte vorher korrekt eingeben'
 end if
end if
if (main==4) then
 ################################### Lagmatrix erzeugen
 if (suc1==1) then
  ansglue=array($(l))
  do i=iota(array($l))
   aglue=stoa('latente Variable 0')
   enabled=array($(l))
   do j=iota(array($(l)))
    a=stoa('latente Variable $(j)')
    glue aglue a > aglue /axis=2
    if (j <= i) then
     enabled($(j))=1
    else
     enabled($(j))=0
    end if
   end do
   aglue=trn(aglue(*,2:$(l+1)))
   set title='$(i)%. lat. Var. hängt ab von: ( -> oder SPACE)'
   menu aglue enabled /mark=y > ans
   glue ansglue ans /axis=2 > ansglue
  end do
  ansglue=ansglue(*,2:$(l+1))
  ansglue=and(or(ansglue,trn(ansglue)),trn(ansglue))
  cli 'cls'
  print ansglue
  putstr ""
  dl=ansglue
 suc4=1
 else
  cli 'cls'
  Print 'Parameter bitte vorher korrekt eingeben'
 end if
end if
if (main==5) then
 ################################### Manifeste Designmatrix
 if (suc1==1) then
```

```
  ansglue=array($(m))
  do i=iota(array($(l)))
   aglue=stoa('manif. Variable 0')
   do j=iota(array($(m)))
    a=stoa('manif. Variable $(j)')
    glue aglue a > aglue /axis=2
   end do
   aglue=trn(aglue(*,2:$(m+1)))
   set title='Indikatoren für den $(i)%. Block (-> oder SPACE)'
   menu aglue /mark=y > ans
   glue ansglue ans /axis=2 > ansglue
  end do
  ansglue=ansglue(*,2:$(l+1))
  # ansglue=and(or(ansglue,trn(ansglue)),trn(ansglue))
  dy=ansglue
  w=dy
  suc5=1
 else
  cli 'cls'
  Print 'Parameter bitte vorher korrekt eingeben'
 end if
end if
if (main==6) then
 ############################################ DPLS-Analyse
 if (suc1==1)
  run 'dpls.mac'
  print 'Analysestart mit...' w d dy dl (dims(y)) genau
  dpls w d dy dl y genau > wout lout skout ltout iter
  run "redun.mac"
  dime=dims(skout)
  dime=dime(2)          # Anzahl der Spalten
  fmy=ftos(dime/2)
  fmy1=ftos(dime/2+1)
  skn=skout(*,1:$fmy)
  skl=skout(*,$fmy1%:0)
  redun lout skn ltout skl y > red redm
  suc6=1
 else
  cli 'cls'
  print 'Erforderliche Daten noch nicht eingegeben'
 end if
end if
if (main==7) then
 ############################################ Ausgabe in Datei
 if (suc6==1) then
  s='kein Kommentar'
  getarg "Kommentar: (in Anführungszeichen)" 'kein Kommentar' > s
  open /over /write "dpls_out.txt"
   write 'Dateiname: DPLS_OUT.TXT'
   write s
   write ''
   write 'Dynamische PLS-Analyse, Ausgabe der Resultate:'
   write ''
   write 'Designmatrix:'
   write d
   write ''
   write 'Designmatrix der Lagbeziehungen:'
   write dl
   write ''
```

```
   write 'Designmatrix der Blockbeziehungen:'
   write dy
   write ''
   write 'Rechengenauigkeit:'
   write genau
   write ''
   Write 'Anzahl der Beobachtungen:'
   Write n
   write ''
   write 'Gewichte W:'
   write wout
   write ''
   write 'Ladungen:'
   write lout
   write ''
   write 'Pfadkoeff. ($(l)%*$(l)%-Matrix (ohne Lagbez.):'
   write skn
   write ''
   write 'Pfadkoeff. ($(l)%*$(l)%-Matrix (mit Lagbez.):'
   write skl
   write ''
   write 'Redundanz:'
   write red
   write ''
   write 'Redundanzmatrix:'
   write redm
   write ''
   write 'Anzahl der Iterationen:'
   write iter
  close /write
  look "dpls_out.txt"
  ####### Ausgabe nach ISP
  aglue=stoa('Ja, Daten nach ISP transferieren')
  a=    stoa('Nein, Abbrechen des Transfers   ')
  glue aglue a > aglue /axis=2
  aglue=trn(aglue)
  menu aglue /title='Ausgabe der Resultate nach ISP?' > ans
  if (ans==1) then
   putarg "Designmatrix" d
   putarg "Designmatrix der Lags" dl
   putarg "Designmatrix der Manifesten" dy
   putarg "Gewichte W" wout
   putarg "Ladungen" lout
   putarg "Pfadkoeffizienten" skout
   putarg "Latente Variablenmatrix" ltout
   putarg "Anzahl der Iterationen" iter
  end if
 else
  cli 'cls'
  Print 'Analyse muß vorerst ausgeführt werden'
 end if
end if
if (main==8) then
 ###################################### Beenden ##
 putstr""
 putstr "Datei DPLS_OUT.TXT enthält Analyseresultate"
 return
end if
end while
```

```
return

# jede Routine prüft nach Eingabe auf Korrektheit der Daten und setzt
# Erfolgreich-Marker
# Kalkulation prüft Erfolgreich-Marker
```

DPLS

```
##########################################################################
# Dynamische PLS-Analyse                                    #
# Universitaet Potsdam                        01.09.1995     #
# Frank Geppert                                        #
# letzte Aenderung: 28.01.1997                              #
#                                              #
# Syntax:      dpls w d dy dl y genau > w b sk lt iter               #
#                                              #
# Eingabe:                                          #
#             w = Startwerte fuer Parametermatrix                  #
#             d = Designmatrix fuer unverzoeg. Bez. latenter Variabl. #
#            dy = Blockdesignmatrix                        #
#            dl = Designmatrix fuer verzoeg. Bez. latenter Variablen  #
#             y = Manifeste Variablen                      #
#         genau = Abbruchkriterium (Nachkommastellen)               #
#                                              #
# Ausgabe:                                          #
#             w = Gewichtsmatrix                         #
#             b = Ladungsmatrix                          #
#            sk = Pfadkoeffizienten                        #
#            lt = Latente Variablen                        #
#          iter = Anzahl an Iterationen                      #
##########################################################################
input/macro/*termin=n > dpls
getarg "Startwerte fuer Gewichte" > w     # Eingabe
getarg "Designmatrix unverzoeg. lat. Variablenbez."   > d
getarg "Blockdesignmatrix" > dy
getarg "Designmatrix verzoeg. lat. Variablenbez." > dl
getarg "Manifeste Variablenmatrix (n+2)"> y
getarg "Abbruchkriterium (Nachkommastellen)" > genau
local end_v iter ac w_alt t tn tl tf dia vsk sd
local s c ds da i j coef n m h res dd b dlp vglue
b=w
n=dims(w)                           # Dimensionen
h=n(2,1)                           # Anzahl latenter Variablen
n=n(1,1)                           # Anzahl manifester Variablen
m=dims(y)
m=m(2,1)-2                          # Anzahl Beobachtungen
 dlp=array($h,$h)+1                    # Designmatrix positiver Lags
do i=(iota(array($h)))                  # Diagonalelemente = 0
 dlp(i,i)=0
end
end_v = 0                          # Abbruchbedingung
iter  = 0                         # Anzahl Iterationen
```

```
 dd=d+trn(d)
 y = trn(trn(y)-mean(trn(y)))                    # Zentrierung der Manifesten
 t=mpy(trn(w),y)                                 # Schaetzen von Latenten
 t=t/trn(sdv(trn(t)))                            # Standardisieren
while (end_v != 1)                               # Solange keine Gleichheit
 w_alt=w                                         # Gleichheitsvergleichsvariabl
 tn=t(*,2:$(m+1))                                # Reihe ohne Lag
 tl=t(*,1:$m)                                    # Reihe mit Lag(t,1)
 tf=t(*,3:$(m+2))                                # Reihe mit Lag(t,-1)
 c=trn(mpy(tn,trn(tn))/(m-1))                    # Korrelation
 ac=trn(mpy(tn,trn(tl))/(m-1))                   # Autokorrelation
 ds=dd*c                                         # Designkorrelation
 da=dl*ac                                        # Designautokorrelation
 t=mpy(ds,tn)+mpy(da,tl)+mpy(trn(da*dlp),tf)    # gewichtet Summieren
  j=1
 while (j <= h)                                  # Regressionen alle Lat.
  i=1
  while (i <= n)                                 # Regression alle Manif.
   if (dy(i,j)==1) then
    regress (trn(t(j,*))) (trn(y(i,2:$(m+1)))) /const=n > res coef
    w(i,j)=coef
   end if
   i=i+1
  end while
  j=j+1
 end while
 w=dy*w                                          # Regressionsmatrix berichtig.
 t=mpy(trn(w),y)                                 # Latente schaetzen
 sd=trn(sdv(trn(t(*,2:$(m+1)))))                 # Standardisierungsfaktor
 t=t/sd                                          # Standardisieren
 if (int(w_alt*10**genau) == int(w*10**genau)) then
  end_v = 1
 end if
 iter=iter+1
end while
w=w/trn(sd)
#####################################################################  Schritt 2
j=1
 while (j <= h)                                  # Regressionen alle Lat.
  i=1
  while (i <= n)                                 # Regression alle Manif.
   if (dy(i,j)==1) then
    regress (trn(t(j,2:$(m+1)))) (trn(y(i,2:$(m+1)))) /const=n > res coef
    b(i,j)=coef
   end if
   i=i+1
  end while
  j=j+1
 end while
 b=dy*b                                          # Regressionsmatrix berichtig.
#####################################################################  Schritt 2
 j=1
 vsk=array($(h*2))
 while (j <= h)                                  # Regressionen alle Lat.
  vglue=array($(h*2),$(m))
  i=1
  while (i <= h)                                 # Regression alle Manif.
   if (d(j,i)==1) then
    vglue(i,*)=t(i,2:$(m+1))
```

```
   end if
   if (dl(j,i)==1) then
    vglue($(i+h),*)=t(i,1:$m)
   end if
   i=i+1
  end while
   regress (trn(vglue)) (trn(t(j,2:$(m+1)))) /const=n > res coef
   glue vsk coef > vsk /axis=2
  j=j+1
 end while
 vsk=vsk(*,2:$(h+1))
 vsk=trn(vsk)
 vsk=int(vsk*1000)/1000
###################################################################### Ausgabe
putarg "Parametermatrix" w
putarg "Ladungen" b
putarg "Pfadkoeffizienten" vsk
putarg "Latente Variablenmatrix" t
putarg "Iterationen" (iter-1)
return
```

REDUN

```
############################################################################
# REDUNDANZBERECHNUNG  Version 3.0                          #
# Syntax: redun b vsk t dl y > red redm                     #
#                                                           #
# Eingabe:                                                  #
#  b (Ladungen)                                             #
#  vsk (Pfadkoeff.)                                         #
#  t (latente Variablen; Zeilen = Variable)                 #
#  dl (Designmatrix der Lagbeziehungen                      #
#  y (manifeste Variablen; Zeilen = Variable)               #
#                                                           #
# Ausgabe:                                                  #
#  red (Redundanz)                                          #
#  redm (Einzelredundanzenvektor)                           #
#                                                           #
# letzte Aenderung: 28.1.1996                               #
############################################################################
input/macro/*termin=n > redun
getarg "Ladungen" > b              # Eingabe Ladungen
getarg "Pfadkoeffizienten unverzoegert.Bez." > vsk  # Eingabe Pfadkoeff.
getarg "Etas (Matrix Zeilen=Blockvariablen)" > t
getarg "Pfadkoeff. verzoegerter Beziehungen" > dl
getarg "Manifeste Variablen (Matrix Zeilen=Variablen)" > y
local c ac tn tl gs1 gs2 gs yn yl cy n i g2 my
local cnt stat dg dcy ttl fmy fmy1 ty gs1 m
local pb pc pbapc pbrpb pcrpc
ty=dims(t)
ty=ty(1,1)                          # Anzahl Bloecke
m=dims(y)
my=m(1,1)                           # Anzahl Manifest.
m=m(2,1)-2                          # Anzahl Beobachtg
tn=t(*,2:$(m+1))                    # Reihe ohne Lag
tl=t(*,1:$m)                        # Reihe mit Lag
corr (trn(tn)) > stat c             # Korrelation
glue (trn(tn)) (trn(tl)) > ttl /axis=2
corr ttl > stat ac
fmy=ftos(ty)
fmy1=ftos(ty+1)
ac=ac(1:$fmy%,$fmy1%:0)             # Autokorrelation
cy=trn(sdv(trn(y))**2)              # Varianz von Y
pb=mpy(b,vsk)
pc=mpy(b,dl)
pbrpb=mpy(pb,mpy(c,trn(pb)))
pbapc=mpy(pb,mpy(ac,trn(pc)))
pcrpc=mpy(pc,mpy(c,trn(pc)))
gs=pbrpb+pbapc+trn(pbapc)+pcrpc
mt_diag gs > dg
g2=dg/cy
gs2=sum(g2)/sum(sign(g2))
open "redun.out" /write /over       # Ausgaben in Datei
write "Redundanzberechnung und deren Zwischenergebnisse:"
write "-----------------------------------------------"
write "Korrelationsmatrix lat. V."
write c
write ""
write "Autokorrelationsmatrix lat. V."
```

```
write ac
write ""
write "Varianzvektor manif. V."
write cy
write ""
write "G (Stern)"
write gs
write ""
write "Diagonale von G (Stern)"
write dg
write ""
write "G als Quotient zweier Diagonalen"
write g2
write ""
write "G(quadrat) als Mittelwert von ..."
write gs2
write ""
close /write
### look "redun.out" ###
putarg "Redundanz" gs2                    # Ausgabe in Speicher
putarg "Redundanzvektor" g2                          # dito
return
```

UNIVERSITÄT POTSDAM
Wirtschafts- und Sozialwissenschaftliche Fakultät

STATISTISCHE DISKUSSIONSBEITRÄGE

Herausgeber: Hans Gerhard Strohe
ISSN 0949-068X

Nr.1	*1995*	***Strohe, Hans Gerhard: Dynamic Latent Variables Path Models*** ***- An Alternative PLS Estimation -***
Nr.2	*1996*	Kempe, Wolfram: Das Arbeitsangebot verheirateter Frauen in den neuen und alten Bundesländern ***- Eine semiparametrische Regressionsanalyse -***
Nr.3	*1996*	Strohe, Hans Gerhard: Statistik im DDR-Wirtschaftsstudium zwischen Ideologie und Wissenschaft
Nr.4	*1996*	Berger, Ursula: Die Landwirtschaft in den drei neuen EU-Mitgliedstaaten Finnland, Schweden und Österreich ***- Ein statistischer Überblick -***
Nr.5	*1996*	Betzin, Jörg: Ein korrespondenzanalytischer Ansatz für Pfadmodelle mit kategorialen Daten
Nr.6	*1996*	Berger, Ursula: Die Methoden der EU zur Messung der Einkommenssituation in der Landwirtschaft - Am Beispiel der Bundesrepublik Deutschland -
Nr.7	*1997*	***Strohe, Hans Gerhard / Geppert, Frank: Algorithmus und Computerprogramm für dynamische Partial-Least-Squares Modelle***

Herausgeber: Prof. Dr. Hans Gerhard Strohe, Lehrstuhl Statistik
der Wirtschafts- und Sozialwissenschaftlichen Fakultät
der Universität Potsdam
Postfach 90 03 27
D-14439 Potsdam
Tel. (+49 331) 977-32 25
Fax. (+49 331) 977-32 10
1997, ISSN 0949-068X