

Geppert, Frank; Strohe, Hans Gerhard

**Working Paper**

## DPLS-partial least squares program for dynamic path models

SFB 373 Discussion Paper, No. 2000,61

**Provided in Cooperation with:**

Collaborative Research Center 373: Quantification and Simulation of Economic Processes, Humboldt University Berlin

*Suggested Citation:* Geppert, Frank; Strohe, Hans Gerhard (2000) : DPLS-partial least squares program for dynamic path models, SFB 373 Discussion Paper, No. 2000,61, Humboldt University of Berlin, Interdisciplinary Research Project 373: Quantification and Simulation of Economic Processes, Berlin,  
<https://nbn-resolving.de/urn:nbn:de:kobv:11-10047870>

This Version is available at:

<https://hdl.handle.net/10419/62262>

**Standard-Nutzungsbedingungen:**

Die Dokumente auf EconStor dürfen zu eigenen wissenschaftlichen Zwecken und zum Privatgebrauch gespeichert und kopiert werden.

Sie dürfen die Dokumente nicht für öffentliche oder kommerzielle Zwecke vervielfältigen, öffentlich ausstellen, öffentlich zugänglich machen, vertreiben oder anderweitig nutzen.

Sofern die Verfasser die Dokumente unter Open-Content-Lizenzen (insbesondere CC-Lizenzen) zur Verfügung gestellt haben sollten, gelten abweichend von diesen Nutzungsbedingungen die in der dort genannten Lizenz gewährten Nutzungsrechte.

**Terms of use:**

*Documents in EconStor may be saved and copied for your personal and scholarly purposes.*

*You are not to copy documents for public or commercial purposes, to exhibit the documents publicly, to make them publicly available on the internet, or to distribute or otherwise use the documents in public.*

*If the documents have been made available under an Open Content Licence (especially Creative Commons Licences), you may exercise further usage rights as specified in the indicated licence.*

# DPLS—Partial Least Squares Program for Dynamic Path Models

*Frank Geppert and Hans Gerhard Strohe*

The approach to dynamic modeling with latent variables has been developed on the base of H. Wold's **partial least squares** (PLS). The original PLS estimation algorithm is virtually applicable. In addition to that lagged and leaded latent variables are used in the iterative process of estimating the weights of the manifest variables. The path coefficients are estimated by OLS. A redundancy coefficient allows to measure the forecasting validity. Finally the algorithm has been programmed in XploRe .

This chapter surveys the theoretical background and explains how **dynamic partial least squares models** are implemented in XploRe . The last part focuses on an example for German share prices.

## 1 Introduction

In addition to the well known estimation and confirmation approach for path models, LISREL by Jöreskog and Sörbom (1987) the partial least squares (PLS) algorithm by Wold (1973) has gained popularity as an instrument of analysis and forecasting in sociometrics and econometrics during recent years. The PLS approach to path models is data oriented and mostly descriptive or explorative, the model being defined purely by an algorithm.

The traditional PLS model involves  $M$  observable, **manifest variables** (MV)  $y^m$  ( $m = 1, \dots, M$ ) and  $K < M$  **latent**, i.e., **not directly observable variables** (LV)  $\eta^k$  ( $k = 1, \dots, K$ ). The latter are assumed to be certain constructs composed from those MVs. Furthermore, the LVs are assumed to be connected with each other by the linear inner or structural model:

$$\eta_t = b_0 + B\eta_t + \nu_t \tag{1}$$

where  $\eta_t = (\eta_t^1, \eta_t^2, \dots, \eta_t^K)^T$  is the column vector of the scores of all latent variables  $\eta^1, \dots, \eta^K$  for one case  $t$  ( $t = 1, \dots, T$ ),  $B$  is a triangular  $K \times K$  matrix of path coefficients with zero diagonal, and  $b_0$  is a location parameter vector usually set equal zero. The error term  $\nu_t$  has zero expectation.

What is called the outer or measurement model describes the assumed linear relations between the MVs and the LVs:

$$y_t = p_0 + P\eta_t + \varepsilon_t \quad (\text{loadings relation}) \quad (2)$$

with a block diagonal  $M \times K$  matrix  $P$  of path coefficients and a zero expectation disturbance term  $\varepsilon_t$ . Again the location parameters  $p_0$  are usually transformed to zero. The latent variables are taken to be weighted sums of manifest variables with a block diagonal weight matrix  $W$ :

$$\eta_t = W'y_t \quad (\text{weight relations}) \quad (3)$$

The iterative estimation of the weights  $W$  is the main aim of Wold's PLS algorithm. The procedure is to start with more or less arbitrarily chosen weights  $W$ . The next stage involves the calculation of proxies  $\eta^{k*}$  for the latent variables  $\eta^k$ .  $\eta^{k*}$  is a weighted sum of all LVs directly connected with  $\eta^k$ .

Finally new weights  $W = (\omega_{mk})$  are estimated by OLS regression between the MVs and the proxies. The scores of the LVs being approximately known after stopping the iteration process, we can easily estimate the parameter matrices  $B$  and  $P$  by OLS, using model equations (1) and (2) respectively.

More details about PLS are given by Lohmöller (1984) who is the author of the PLS computer programme LVPLS.

In Section 2 of this paper a PLS-like approach to a class of dynamic models with latent variables will be suggested. The way to use these models for prediction will be shown, and a measure for goodness of fit will be deduced. Then, a computer programme for dynamic partial least squares modelling will be presented. The final section will show a small five-block model with an first order autoregressive distributed lag relation between the LVs.

## 2 Theoretical Background

### 2.1 The Dynamic Path Model DPLS

The dynamic form of the structural model can be transformed into the exterior shape of the “normal” PLS model:

$$\eta_t = F\eta_t + \nu_t \quad (4)$$

where

$$F = B + CL \quad (5)$$

is a matrix containing the lag operator  $L$  with  $L\eta_t = \eta_{t-1}$ . On what we call now the dynamic PLS model (DPLS)

$$\begin{aligned} \eta_t &= F\eta_t + \nu_t \\ y_t &= P\eta_t + \varepsilon_t \\ \eta_t &= W^T y_t \end{aligned} \quad (6)$$

the original PLS algorithm is applicable. Initially, Boolean design matrices  $D_B$ ,  $D_C$  and  $D_P$  corresponding to the unlagged and lagged dependencies in the inner model (6) and to the outer model, i.e., to the zero restrictions for the coefficient matrices  $B$ ,  $C$  and  $P$ , must be fixed. The inner model can be illustrated by a path diagram, including additional arrows for the lagged relationships. The inner design matrix  $D_B$  contains the digit one where there is a connection between two LVs in the path model and consists of zeros elsewhere. Similarly, the lag design matrix  $D_C$  consists of ones and zeros corresponding to whether or not there is assumed to be first order lagged (auto)regression between latent variables.  $D_P = (d_{mk})$  is the outer design matrix corresponding to whether or not a variable  $y^m$  of  $Y$  belongs to the block of a certain latent variable, i.e., a row  $\eta^k$  of  $H$ .

### 2.2 PLS Estimation with Dynamic Inner Approximation

For simplicity, the symbols for the empirically estimated LVs and coefficients will not be distinguished from those for the corresponding theoretical quantities in this section. In order to estimate the weight matrix  $W$ , the following steps will be repeatedly executed:

1. Initial representation of the latent variables as components of the manifest variables with chosen starting values for the matrix  $W$

$$\eta_t = W^T y_t \quad (7)$$

2. Standardization of the LVs to unit variance

$$\eta_t := \sqrt{T} (I * HH^T)^{-\frac{1}{2}} \eta_t \quad (8)$$

where

$$H = (\eta_1, \eta_2, \dots, \eta_T), \quad (9)$$

is the  $K \times T$  matrix of all time scores of  $\eta_t$  for  $t = 1, \dots, T$ . Elementwise multiplication of matrices is denoted by  $*$ .

3. Calculation of “neighbourhood” variables corresponding to the inner path model:

$$\eta_t^* = F^* \eta_t \quad (10)$$

$$F^* = B^* + C^* L + C^{*T} L^{-1} \quad (11)$$

that means

$$\eta_t^* = B^* \eta_t + C^* \eta_{t-1} + C^{*T} \eta_{t+1} \quad (12)$$

where  $B^*$  and  $C^*$  are suitable inner weighting matrices, e.g.:

$$B^* = (D_B + D_B^T) * R \quad (13)$$

$$C^* = D_C * A \quad (14)$$

with  $D_B$  and  $D_C$  being the design matrices for the inner model.

$$R = HH^T / T \quad (15)$$

$$A = H (LH)^T / T \quad (16)$$

are the correlation matrix and the first order autocorrelation matrix of LVs, respectively, with

$$H = (\eta_1, \eta_2, \dots, \eta_T), \quad (17)$$

$$LH = (\eta_0, \eta_1, \dots, \eta_{T-1}) \quad (18)$$

4. New values of the weight matrix  $W$  are gained by OLS estimation:

$$y_t^m = \omega_{mk} \eta_t^{k*} + v_t^{mk} \quad \text{if } d_{mk}=1 \quad (\text{Wold's Mode A}) \quad (19)$$

where  $D_P = (d_{mk})$  is the outer design matrix.

5. The estimated coefficients  $\omega_{mk}$  are substituted for the previous elements of the weight matrix  $W$

$$W := (\omega_{mk}).$$

Using this new weight matrix we continue the procedure by repeating step 1. The iteration process is stopped when subsequent estimations of the LVs  $\eta_t$  in step 2 do not relevantly differ from the previous ones.

Then the coefficient matrices  $B$  and  $C$  of the inner model (4)

$$\eta_t = (B + CL) \eta_t + v_t \quad (20)$$

can be estimated by a suitable method for dynamic models, such as OLS, GLS, Cochrane-Orcutt, ECM etc. The loadings  $P$  of the outer model (2) are estimated by simple OLS.

### 2.3 Prediction and Goodness of Fit

By substituting (4) for  $y_t$  in (2) we obtain

$$\begin{aligned} y &= P\eta_t + \varepsilon_t \\ &= PF\eta_t + P\nu_t + \varepsilon_t \end{aligned} \quad (21)$$

Then substituting (6) for  $\eta_t$ , we have

$$y_t = PFW^T y_t + P\nu_t + \varepsilon_t \quad (22)$$

or

$$y_t = (PBW^T y_t + PCW^T y_{t-1}) + (P\nu_t + \varepsilon_t) \quad (23)$$

Using this prediction formula, we can construct a goodness-of-fit criterion. From (21) it follows that the predictable part of  $y_t$  is  $y_t^* = PF\eta_t$ . Let  $Y^* = (y_1^*, \dots, y_T^*)$  denote the whole predicted data matrix,  $H = (\eta_1, \dots, \eta_T)$

the matrix of the LVs, and  $R$  the empirical correlation or covariance matrix of  $H$ . Then the empirical covariance of these predictions is

$$\begin{aligned} \text{cov}(Y^*) &= P F \text{cov}(H) F^T P^T \\ &= P F R F^T P^T \end{aligned} \quad (24)$$

$$\begin{aligned} \text{cov}(Y^*) &= P(B + CL)H H^T (B + CL)^T P^T / T \\ &= (PBH H^T B^T P^T + PC(LH)H^T B^T P^T + PBH(LH)^T C^T P^T \\ &\quad + PC(LH)(LH)^T C^T P^T) / T \\ &\approx PBRB^T P^T + PBAC^T P^T + PCRC^T P^T + (PBAC^T P^T)^T = G^* \end{aligned} \quad (25)$$

with  $A$  being the first order autocorrelation matrix. The inconsiderable inaccuracy of the last relationship arises from tiny differences that might occur between the covariances of the latent variables  $HH^T / T$  and those of the lagged LVs  $(LH)(LH)^T / T$ .

It is easy to see that  $G^*$  contains in its diagonal the variances of the predictable part, or what Lohmöller (1989) calls the “redundant” part, of the MVs.

Following Lohmöller (1989) again we calculate the ratio of two diagonal matrices

$$G = (I * G^*)(I * \Sigma_y)^{-1} \quad (26)$$

where  $\Sigma_y$  denotes the empirical covariance matrix of the manifest variables. The entries in the diagonal of  $G$  are ratios expressing to what extent the variance of each manifest variable is reproduced by the variance of the predictable part, i.e., by the model.

The average of these measures

$$G^2 = \text{trace } G / M \quad (27)$$

is the redundancy coefficient or average redundancy and is used for the evaluation of the goodness of fit of the model.

## 3 Estimating a DPLS-Model

### 3.1 The Computer Program DPLS

The very sophisticated computer program LVPLS for static partial least squares models was developed by Lohmöller (1984)(1989). Unfortunately, it is not effectively applicable to dynamic path models.

The computer programme DPLS (Strohe and Geppert 1997) is available as both PC-ISP macros and XploRe (Härdle, Klinke, and Turlach 1995) quantlets. The syntax to control the programme is nearly identical in both versions. In XploRe it consists of three basic modules. The first one puts the DPLS algorithm within the XploRe environment into action, the second one calculates a redundancy measure and the last one is a tool for easier creating input variables. This one works on the base of several menus and dialogs. So it is not necessary to know all input matrices exactly in advance and the only input to be prepared by the use of ordinary XploRe commands is the indicator matrix.

The further input matrices firstly are to be created and defined by the above mentioned tool (third basic module) or “by hand”. But if a user wants to calculate a model several times one after the other and if he/she wants to modify parameters on the fly in a programmed simulation or something like this he/she can take an advantage by using the basic programmes directly. In that case one has to know more about the details of input and output of the corresponding module.

### 3.2 Creating Design-Matrices

```
design = makedesign(y)
      generates the design matrices by a dialogue
```

At first, for running a DPLS session, XploRe has to be started. One must load the quantlib `metrics` into memory by using the `library` command. XploRe users have to type:

```
library("metrics")
```



First, a design-making session (Figure 1) could be started by the command mentioned above. There  $y$  is a  $n \times l$ -matrix with manifest variables (indicators). The output design contains several variables. Among them  $dy$  is matrix  $(l \times k)$  with outer designs (0 or 1). Rows are counting manifest variables. Furthermore,  $d$  is a  $k \times k$  matrix with inner unlagged designs (0 or 1). No diagonal values are allowed. The  $(k \times k)$  matrix  $d1$  contains the inner lagged designs (0 or 1). Diagonal elements are showing autoregression. The matrix  $w$  represents start weights with same dimensions as  $dy$ . It is simply a copy of  $dy$ . It is possible to

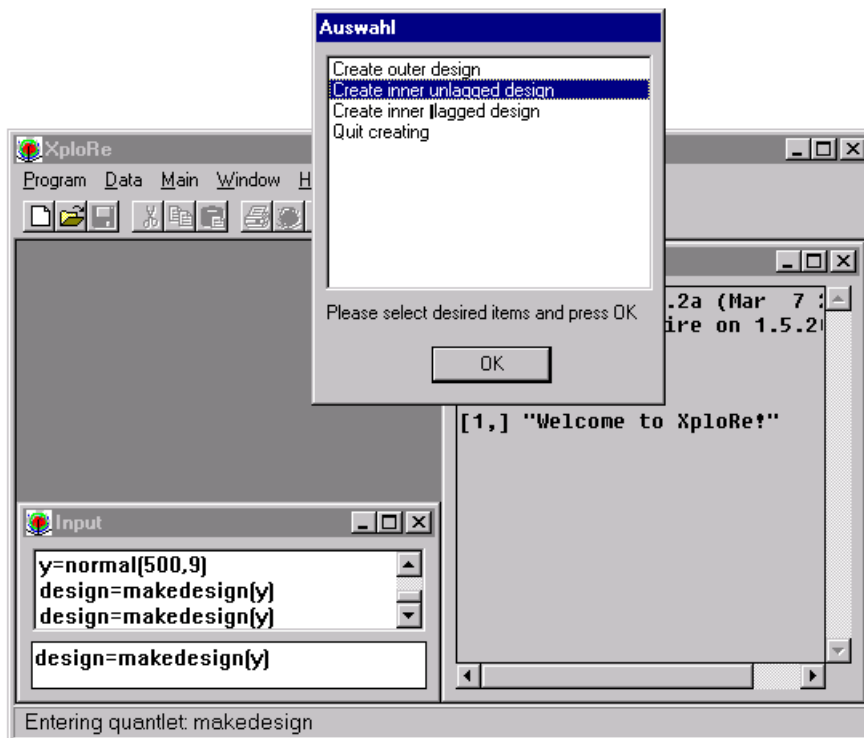


Figure 1: A session for creating design-matrices in XploRe

address these variables in a container-variable with the point as an separator between them both. In the case above:

```
design.dy
design.w
design.dl
design.w
```

The syntax of `makedesign` is to be seen below:

Input parameters:

`y`  
 $n \times l$  matrix, manifest variables (indicators)

Output parameters:

`out.dy`  
 $l \times k$  matrix, outer designs (0 or 1, rows are counting manifest variables)

`out.d`  
 $k \times k$  matrix, inner unlagged designs (0 or 1, no diagonals allowed)

`out.dl`  
 $k \times k$  matrix, inner lagged designs (0 or 1, diagonals are showing autoregression)

`out.w`  
 $l \times k$  matrix, start weights, same as `dy`


Besides that, it is possible to create all matrices by hand. An easy sample can be seen below:

```
library("metrics")
randomize(13409)
b1=0.3
c1=0.6
s=500
n1=normal(s+1)
n1lag=n1[1:s,]
n1=n1[2:rows(n1),] ; innermodel
n2=b1*n1+c1*n1lag+normal(rows(n1))/5
n=n1~n2
nn=n./sqrt(var(n)) ; loadingsmatrix
```

```

p=(1|2|3|4|0|0|0)~(0|0|0|0|5|6|7)
y=nn*p'+normal(rows(n),rows(p))/8
d=(0|1)~(0|0)
dl=(0|1)~(0|0)
w=(1|1|1|1|0|0|0)~(0|0|0|0|1|1|1)
myfit=dpls(w,d,w,dl,y,1,3)
myfit.b
myfit.sk
myfit.sk1

```

 dpls01.xpl

### 3.3 Estimating with DPLS

```

myfit = dpls(w, d, dy, dl, y, lag, acc)
      estimates the weights and loadings matrices

```

As one could see above, a common XploRe session of DPLS (Figure 2) can be started by the described command. There `dpls` denotes the program used and `w` is a term for the weights the algorithm is to start with. The term `d` means the inner and `dy` the outer design matrix. Furthermore, `dl` is a symbol for an inner design matrix as well but it contains the lagged connections in the model. The matrices `d`, `dy` and `dl` represent our idea about the model structure. The scalar `lag` determines the lag order the algorithm has to take into account. The matrix `y` contains the time series of all indicators (manifest variables) and represents virtually all empirical information available. The digit `acc` stands for “accuracy” and controls the final stop of the iteration process. The quantlet `dpls` uses this number in order to check after every iteration whether or not the new calculated values are significantly different from the previously calculated values `acc` specifying how many decimals are taken into consideration.

The syntax of quantlet `dpls` has the following structure:

Input parameters:

**w**  
 $l \times k$  matrix, start weights

**d**  $k \times k$  matrix, inner unlagged designs (0 or 1, no diagonal values allowed)  
**dy**  $l \times k$  matrix, outer designs (0 or 1, rows are counting manifests)  
**dl**  $k \times k$  matrix, inner lagged designs (0 or 1, diagonals are showing autoregression)  
**y**  $n \times l$  matrix, manifest variables (indicators)  
**lag** a scalar of lag order  
**acc** scalar, canceling criterion

Output parameters:

**myfit.wg** matrix, weights  
**myfit.b** matrix, loadings  
**myfit.sk** matrix, path coefficients  
**myfit.sk1** matrix, lagged path coefficients  
**myfit.lk** matrix, latent variables  
**myfit.iter** scalar, number of iterations

The matrices **myfit.wg** and **myfit.b** correspond to  $W$  and  $P$  in (2) and (3), respectively. And **myfit.sk** and **myfit.sk1** are the  $B$  and  $C$  in (5).

All matrices are structured in a comparable way. The number of rows are supposed to correspond with the number of variables and the number of columns

should be identical with the corresponding number of observations. If the following model is taken as an example, which contains 41 manifest variables with 74 observations, then the matrix  $y$  has the shape  $41 \times 74$ . But the design matrix  $dy$  which contains the available connections between every manifest and every latent variable ("1" for a connection and "0" for none) must have the shape  $41 \times 5$  because this model contains 5 latent variables. One can observe the same structure in the matrix  $w$  with the difference that the matrix could contain at the spots of "1" any other value which should be used as starting weight by the algorithm.

The  $d$  and  $d1$  matrices are squared. The rows and the columns stand for latent variables. And since the following model is designed with connections all leading to the fifth variable, all rows are filled with noughts except of the last. This row describes which of the variables are connected to the fifth variable. With the same logic one has to decode the output variable  $sk$  which contains the unlagged path coefficients in the first part followed by the lagged ones.

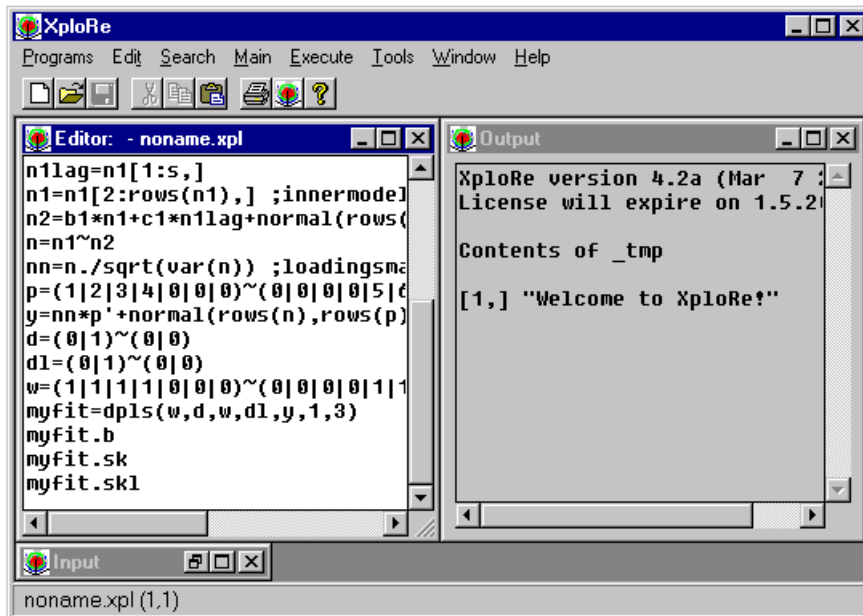


Figure 2: DPLS session in XploRe

### 3.4 Measuring the Forecasting Validity

```
myredun = redun(b, sk, lk, skl, y)
           estimates the redundancy matrices
```

The last tool for a DPLS session is the module `redun`. It calculates on the base of formula (27) the redundancy as a criterion that is used for the evaluation of the goodness of fit of the model. The described overview shortly shows the usage of this module.

The syntax of `redun` is to be seen below:

Input parameters:

`b`  
matrix, loadings

`sk`  
matrix, path coefficients

`lk`  
matrix, latent variables

`skl`  
matrix, lagged path coefficients

`y`  
matrix, manifest variables (indicators)

Output parameters:

`red`  
scalar, redundancy value

`redm`  
vector, redundancy values

## 4 Example: A Dynamic Latent Variable Model for German Share Prices

### 4.1 The General Path Model

The main purpose of DPLS in XploRe is the construction of general variables which optimally represent the dynamics of corresponding set of numerous indicators with their individual but similar dynamics. Indicators or manifest variables of this sort are share prices. Share price indexes are weighted sums of individual share prices. Weights are e.g. returns or quantities of stocks purchased. But by usage of dynamic path models the weights will be estimated as coefficients within the latent variable model. Such a coefficient represents a sort of importance of the individual share price within the construct of the latent variable in the context of the whole model.

The question to be answered by this specific model is, whether a certain construct of share prices can show the dynamic dependence of share prices on economic indicators. This construct could then be used as a kind of new share price index.

The challenge is to specify a dynamic model that represents the most important economic relationship of the index as a stable representant of the set of share prices under consideration.

Besides the latent share price variable (SP), the model under consideration contains the LV **labour market** (LM), **money market** (MM), **domestic economic performance** (DP) and **foreign market** (FM).

The next table shows the manifest variable belonging to these LVs. The next figure shows the relationships selected: The latent share price variable SP is assumed to statistically depend on LM, MM, DP, and FM. Furthermore a first-order auto-regression of SP and a first-order lagged dependency on LM is supposed. This selection is the result of some previous empirical pilot studies eliminating further lagged relationships.

## 4.2 Manifest Variables and Sources of Data

Domestic Performance:

Nr:	Domestic Performance:	Unit:
1	Incoming Orders (Processing Business)	1991=100
2	Incoming Orders (Construction Industry)	1991=100
3	Production (Manufacturing Business)	1991=100
4	Production (Processing Business)	1991=100
5	Commodity Trade (Exportation)	Billion DM

Origin of Data: Monthly Reports DBBK without Table 4 and Zahlungsbilanzstatistik DBBK

Foreign Market:

Nr.:	Foreign Market:	Unit:
6	Dow Jones Industrial Average	Index
7	Commodity Trade (Importation)	1991=100
8	US\$ against 18 Industrial Countries	1991=100
9	Incoming Orders from Foreign Countries	1991=100
10	Discount Rate USA	% p.A.

Origin of Data: Monthly Reports DBBK without Table 4 and Zahlungsbilanzstatistik DBBK

Money Market:

Nr.:	MoneyMarket::	Unit:
11	MoneySupply(M3)	BillionDM
12	DiscountRateDBBK	%p.A.
13	DM against US\$	1972=100

Origin of Data: Monthly Reports DBBK without Table 4



Labour Market:

Nr.:	Labour Market	Unit:
14	Gross Earnings	Quantity
15	Number of Unemployed	Quantity
16	Vacancies Zahlungsbilanzstatistik	Quantity
17	Short-Time Workers	Quantity

Origin of Data: Zahlungsbilanzstatistik DBBK

German Stock Prices:

Nr.:	German Stock Prices	Unit:
18	ALV	DM
19	BAS	DM
20	BAY	DM
21	BMW	DM
22	CBK	DM
23	DAI	DM
24	DBK	DM
25	DGS	DM
26	DRB	DM
27	HEN3	DM
28	HOE	DM
29	KAR	DM
30	LHA	DM
31	LIN	DM
32	MAN	DM
33	MMW	DM
34	PRS	DM
35	RWE	DM
36	SCH	DM
37	SIE	DM
38	THY	DM
39	VEB	DM
40	VIA	DM
41	VOW	DM

Origin of Data: Deutsche Börse

### 4.3 Empirical Results

Both models with levels and with differences had been estimated. Transformations such as differences or logarithms can easily added within the XploRe environment. Despite the models on level base have produced some significant path coefficients and high redundancy we have finally decided for a model on first difference base.

The Figure 3 shows the path coefficients of the latent variables and the weights of the manifest variables. Dotted arrows indicate lagged dependencies. The differentiated latent share-price variable has only a very low autoregressive component (0.08). This is only 1/10 of the amount found for levels. The strongest dependency is that on the latent foreign-market variable (0.33). The weakest one is that on the domestic economic performance (-0.06). We have found a medium degree dependency on the latent money-market variable (0.26). Furthermore there is a significant first order lagged relationship with the latent labour market (0.18).

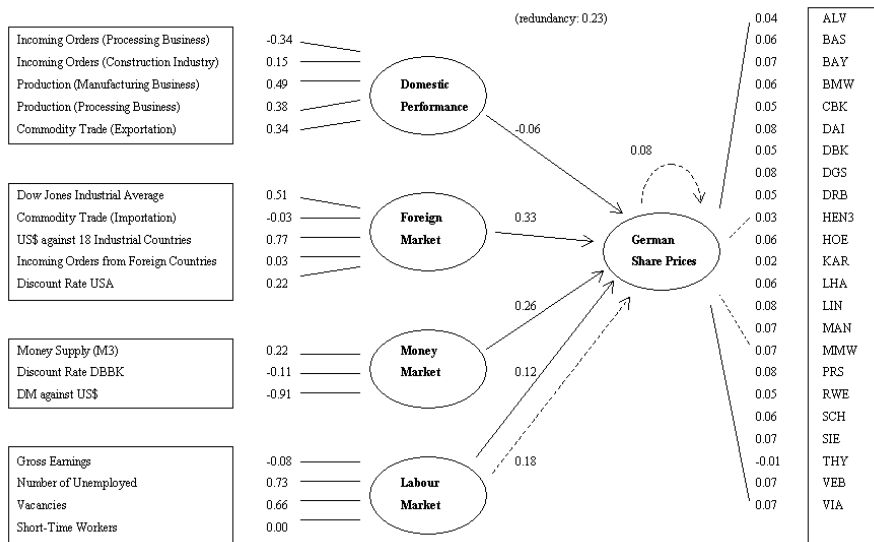


Figure 3: A Model with German share prices

The weights can be interpreted as the partial contribution of the individual manifest variable within the block of indicators belonging to a latent variable. In this meaning, it is easily to see that some of the MVs can be dropped because of their negligible contribution, e.g. **importation**, **gross earnings** and each of the share prices taken alone.

## References

- Geppert, F. (1996 ). Bearbeitung, Programmierung, Simulation und Anwendung eines PLS-Algorithmus für einfache dynamische Modelle mit latenten Variablen. Diploma thesis supervised by H. G. Strohe, Universität Potsdam.
- Härdle, W., Klinke, S., and Turlach, B. A. (1995). XploRe: An Interactive Statistical Computing Environment. Springer-Verlag, Berlin.
- Jöreskog, K. G. and 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.
- Strohe, H. G. (1995). Dynamic Latent Variables Path Models—An Alternative PLS Estimation. Statistische Diskussionsbeiträge Nr. 1, Universität Potsdam.
- Strohe, H. G. and Geppert, F.(1997). DPLS—Algorithmus und Computerprogramm für dynamische Partial-Least-Squares-Modelle. Statistische Diskussionsbeiträge Nr. 7, Universität Potsdam.
- 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.