MODELING, IDENTIFICATION AND CONTROL, 1980, VOL. 1, NO. 4, 215–229
doi: 10.4173/mic.1980.4.2

CYPROS—cybernetic program packages

ARNE TYSSØ†

Keywords: *computer aided design, computer programming, interactive program, software packages.*

CYPROS is an interactive program system consisting of a number of special purpose packages for simulation, identification, parameter estimation and control system design. The programming language is standard FORTRAN IV and the system is implemented on a medium size computer system (Nord-10). The system is interactive and program control is obtained by the use of numeric terminals. Output is rapidly examined by extensive use of video colour graphics. The subroutines included in the packages are designed and documented according to standardization rules given by the SCL (Scandinavian Control Library) organization. This simplifies the exchange of subroutines throughout the SCL system. Also, this makes the packages attractive for implementation by industrial users. In the simulation package, different integration methods are available and it can be easily used for off-line, as well as real time, simulation problems. The identification package consists of programs for single-input/single-output and multivariable problems. Both transfer function models and state space models can be handled. Optimal test signals can be designed. The control package consists of programs based on multivariable time domain and frequency domain methods for analysis and design. In addition, there is a package for matrix and time series manipulation. CYPROS has been applied successfully to industrial problems of various kinds, and parts of the system have already been implemented on different computers in industry. This paper will, in some detail, describe the use and the contents of the packages and some examples of application will be discussed.

1. Introduction

At The Norwegian Institute of Technology, Division of Engineering Cybernetics and at SINTEF, The Foundation of Scientific and Industrial Research at the University of Trondheim, Department of Automatic Control, development and use of interactive computer packages started in the early seventies (Pedersen *et al.* 1972). The field of interest was concentrated on simulation and control system design based on state space models. A program system, DAREK, was implemented on a central computer system, UNIVAC 1108, and the programming language was ALGOL.

The program system was made strongly dependent on the computer configuration and could, therefore, only be transferred to similar computer systems. Some installations were made outside the university, even outside the country (Helsingfors and Oulu in Finland, 1977), with success, but it was almost impossible to spread it among industrial users.

Experiences with the system showed that it was excellent for educational purposes, but the research engineers found computational costs high and the efficiency in interactive mode proved to be poor because of low capacity in the input/output handling. No graphic displays were available, and programs for solving identification and estimation problems were not implemented.

Received 18 January 1980.

† Division of Engineering Cybernetics, The Norwegian Institute of Technology, University of Trondheim, Trondheim, Norway.

In 1975, when SINTEF, Division of Automatic Control, bought a medium size computer system, Nord-10, (extended with a Nord-10S computer in 1978) with all necessary peripheral equipment, it was decided to develop an improved interactive program system with a broader field of application.

At the same time, a cooperation project between Scandinavian Control Laboratories was going on and, in this project, one of the main objectives was to establish programming and documentation rules to be used in developing interactive program systems (Elmqvist *et al.* 1976). The rules were determined primarily to improve flexibility and portability but, also, to simplify the documentation problem. It was, therefore, natural to base the new program system on these rules.

The new program system, named CYPROS (cybernetic program packages), covers today a broad field of control engineering problems and, in the following, the main contents will be outlined. Some important points concerning the design of interactive program systems will be briefly discussed, and examples of applications and experiences from industrial installations will be given.

2. Hardware configuration

CYPROS is implemented on a medium size computer system, Nord-10S. It has a core memory of 256 K, 16 bits word length. Floating point multiplication and addition requires 13·5 μs and 3·6–14 μs respectively. Program control is obtained by use of numeric terminals (8) and output is examined by use of video colour graphics. Figure 1 shows the peripheral equipment connected to the computer.

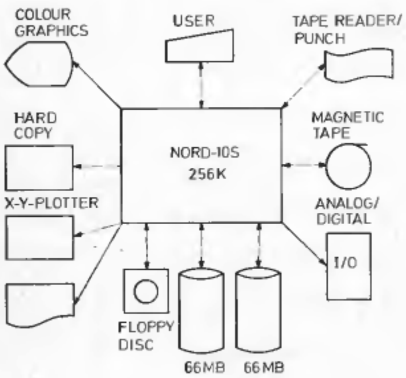

Figure 1. Hardware configuration.

The most frequently used programs are stored permanently on file, but the main storage medium is floppy discs. Because of the computer capacity, several users can be handled simultaneously with reasonable efficiency. However, the efficiency drops drastically if more complex, high order problems, like non-linear estimation and non-linear simulation, are handled at the same time.

3. Structure of the program packages

3.1. *Aims and requirements*

The CYRPOS packages are mainly oriented towards research activities at the Institute, but hopefully they will be found attractive among industrial users too. It is emphasized that maintenance and transfer of programs are made as simple as possible. Therefore, it is necessary to establish some guidelines regarding program design and documentation.

Software and hardware facilities differ quite a lot from one computer system to another, and it is difficult to make general rules. However, it is possible to structure the programs in such a way that all input–output handling, e.g. use of graphical displays and files, is taken care of in special, computer dependent subroutines, while all general computation is taken care of in subroutines designed according to specified rules. Regarding use and transfer of programs and subroutines, it should be stressed that strict documentation is a necessity.

The documentation should be of such quality that other users easily understand the program performance and are able to modify the programs without difficulty. Experience shows that programs often have to be modified, not only because of improvements suggested by users, but also because of changes in the computer systems.

If the program implementor works without guidelines, the result often turns out to be a product which works well but with a personal style that makes it inconvenient for others to use and very difficult to modify.

The rules proposed in the SCL (Elmqvist *et al.* 1976) have proved to be useful but sometimes difficult to maintain. FORTRAN IV (USA ANS X3.9, 1966) is chosen as the programming language. Obviously, FORTRAN IV is not a very effective language, but, on the other hand, FORTRAN IV compilers are available at almost all computer systems. This makes exchange and transfer of programs and subroutines attractive.

CYPROS is designed according to well-accepted ideas for interactive programs (Lemmens and van den Boom 1979). The organization of the interaction is based on a question–answer type model, and the user does not have to learn any commands. He or she has only to select the program of current interest and the program gives the relevant information as well as the necessary instructions during operation. The user must have some insight into the computer's file organization, and he/she must be able to specify his/her models as FORTRAN subroutines. As regards the more advanced programs, e.g. non-linear estimation, it is required that the user has studied the user's manual and that he/she knows, to some extent, the theoretical background for the problem under consideration. Otherwise, there will be a number of wasted computer runs and, at worst, erroneous results.

Owing to the limitation in core capacity there exist maximum allowed orders of the models to be handled. For instance, in the simulation program SIM the maximum allowed system order is 100, and in the identification programs EXKALM and MLPROG the maximum allowed system order is 20.

3.2. *Program structure*

The structure of a typical program is shown in Fig. 2. If the user is not familiar with the program, he/she may at any stage obtain the necessary information using the command HELP.

Each program usually solves a number of related problems, for example the parameter estimation program MLPROG computes maximum likelihood estimates as well as curvature and sensitivity functions. Within each main program there will thus be a number of subprograms. Figure 3 shows the structure of a typical subprogram. The subprogram fetches the necessary data from the data base. The user may then

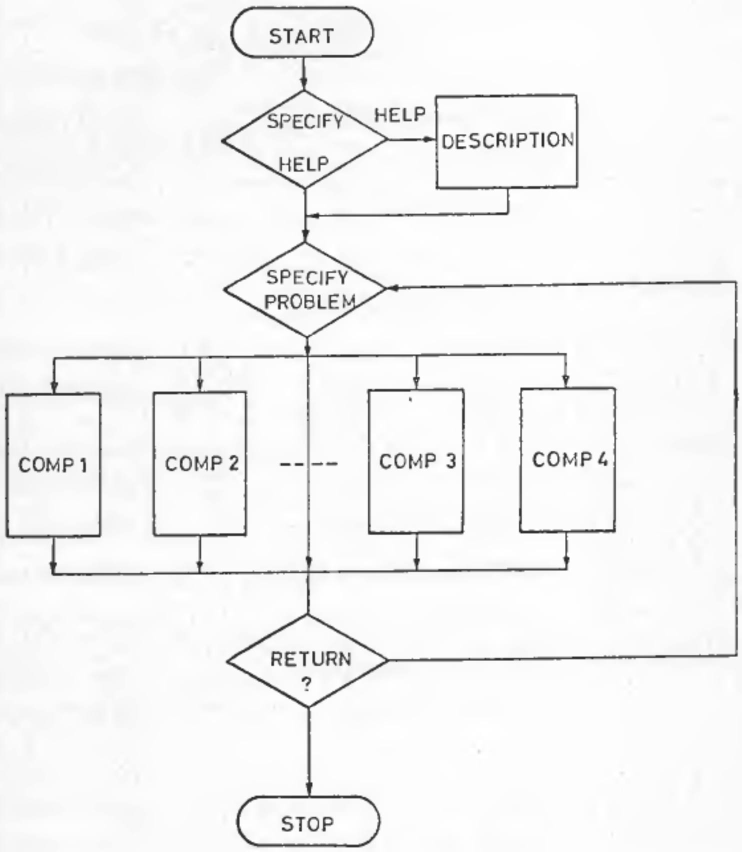

Figure 2. General program structure.

check the data and make the necessary changes. During the computation, the user is allowed to communicate with the program. In order to give more insight into the interactive aspects of the system, a more detailed description of an example is given in Appendix 1.

The results may be stored at the data base for later use or may be presented directly on the colour graphic display. Using a colour graphic display makes it very convenient to compare and examine more time series simultaneously.

3.3. *Model description*

The programs dealing with non-linear problems are based on the general state space description:

$$\left.\begin{aligned}\dot{x}(t)&=f(x(t),\, u(t),\, v(t),\, \theta(t),\, t)\\ y(t)&=g(x(t),\, u(t),\, w(t),\, \theta(t),\, t)\end{aligned}\right\} \quad (1)$$

where

$x(t)$: process state vector,

$u(t)$: control vector,

$v(t)$: process disturbance vector,

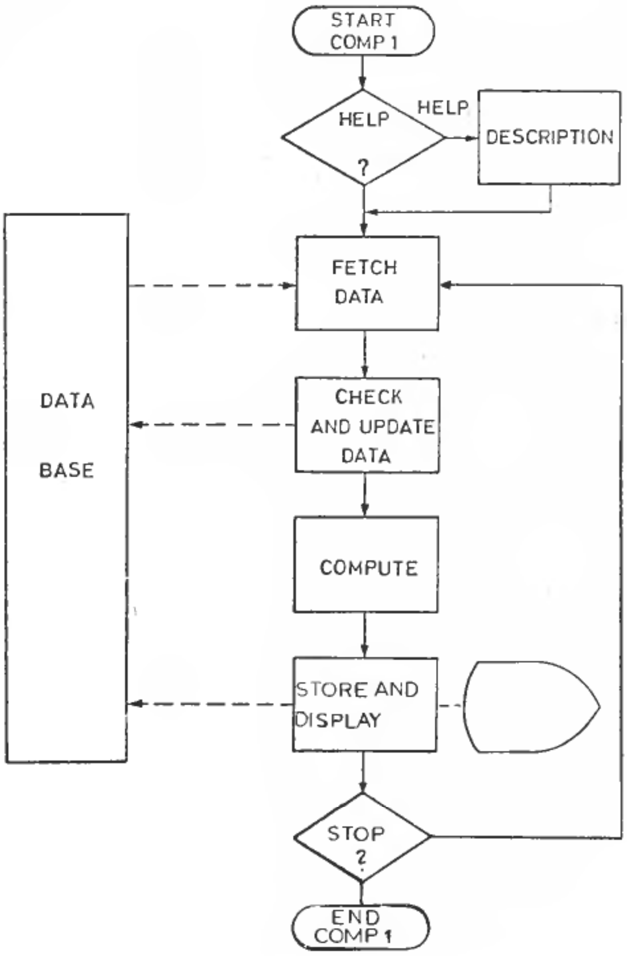

Figure 3. Subprogram structure.

$\boldsymbol{\theta}(t)$: parameter vector,

$\boldsymbol{w}(t)$: measurement disturbance vector,

$\boldsymbol{y}(t)$: measurement vector.

In the discrete case, the models have the following form:

$$\left.\begin{aligned} \boldsymbol{x}(k+1) &= \boldsymbol{f}(\boldsymbol{x}(k), \boldsymbol{u}(k), \boldsymbol{v}(k), \boldsymbol{\theta}(k)) \\ \boldsymbol{y}(k) &= \boldsymbol{g}(\boldsymbol{x}(k), \boldsymbol{u}(k), \boldsymbol{w}(k), \boldsymbol{\theta}(k)) \end{aligned}\right\} \tag{2}$$

The corresponding linearized versions are

$$\left.\begin{aligned} \dot{\boldsymbol{x}} &= A\boldsymbol{x} + B\boldsymbol{u} + C\boldsymbol{v} \\ \boldsymbol{y} &= D\boldsymbol{x} + E\boldsymbol{u} + \boldsymbol{w} \end{aligned}\right\} \tag{3}$$

and

$$\left.\begin{aligned} x(k+1) &= \Phi x(k) + \Delta u(k) + \Omega v(k) \\ y(k) &= Dx(k) + Eu(k) + w(k) \end{aligned}\right\} \quad (4)$$

Equation (3) can also be described as a transfer function:

$$\left.\begin{aligned} y(s) &= G(s) \cdot u(s) \\ G(s) &= D(sI - A)^{-1}B + E \end{aligned}\right\} \quad (5)$$

where

4. Contents of the packages

CYPROS consists of a number of special purpose packages for simulation, identification, parameter estimation and control system design. In addition, there are packages for general matrix handling and time series manipulation and a general subroutine library (Tyssø 1978 a).

4.1. *Simulation package, SIM*

In the SIM package, different methods are available for solving differential and difference equations. In the continuous case, it is assumed that the process can be described by eqn. (1) and the corresponding control system by the following equations:

$$\left.\begin{aligned} \dot{z} &= f_z(z, y, r, \theta, t) \\ u &= f_u(z, y, r, \theta, t) \end{aligned}\right\} \quad (6)$$

where

z: control system state vector

r: reference vector

In addition we have

$$r = f_r(\theta, t), \quad v = f_v(\theta, t) \quad \text{and} \quad w = f_w(\theta, t)$$

Figure 4 shows the connection between the system equations.

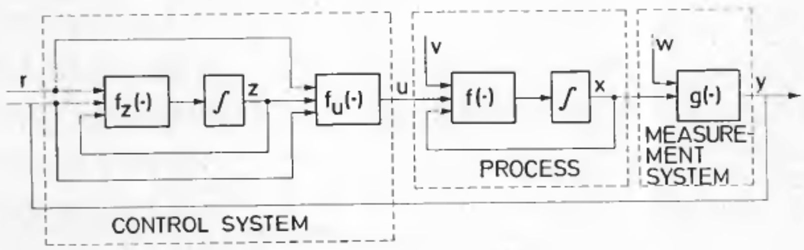

Figure 4. Model structure.

The user specifies his models by means of FORTRAN subroutines, and the model parameters are stored on the data base and are easily changeable. One or more of the specified models may be omitted.

Discrete time systems (eqn. (2)) are treated in a similar way. The simulation results are presented on a colour graphic display and hard copies are easily produced. The most commonly used integration methods are implemented, such as Euler and Kutta–Merson methods, but also a method for solving stiff differential equations is included. SIM can be used in off-line simulation as well as real time simulation. An example of use of the program is presented in Appendix 1. Further details are given in Opdal (1977 c).

4.2. *Identification package, PROSID*

The PROSID package contains programs for solving identification and parameter estimation problems. Optimal test signals can be designed.

MLPROG

This program is based on the model description given in eqn. (2). The maximum likelihood (ML), the least squares (LS) and the prediction error (PE) methods are implemented. The LS method and the PE method can be regarded as special cases of the ML method and, therefore, only the ML method will be briefly outlined.

The criterion to be minimized is given by

$$L=\tfrac{1}{2}\sum_{k=1}^{N}\left[\varepsilon_k{}^T R_k{}^{-1}\varepsilon_k+\ln(\det R_k)\right] \tag{7}$$

where ε_k is the innovation process and R_k is the innovation covariance matrix. ε_k can be found using the Kalman filter approach, and because the model is non-linear an extended Kalman filter must be used.

The structure of the estimator is shown in Fig. 5.

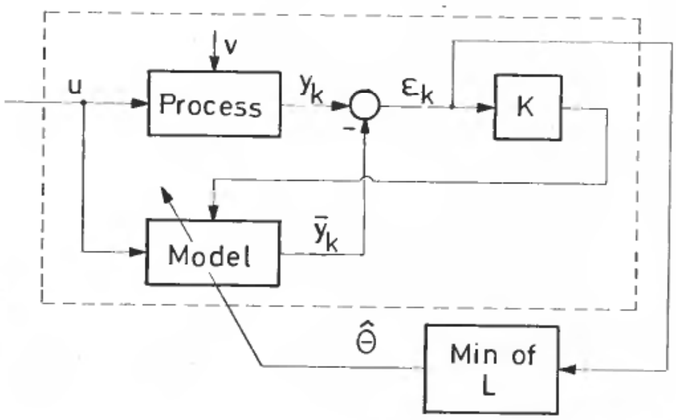

Figure 5. Maximum likelihood estimator with a Kalman filter for a given **θ** value.

The maximum likelihood algorithm works as follows:

(1) Make an initial choice of the estimate $\hat{\theta}_0$. The counting variable, i, is set to zero.

(2) Use the extended Kalman filter and perform a simulation with $\hat{\theta}=\hat{\theta}_i$. Compute $L=L_i$.

(3) Use the Powell minimization method to update $\hat{\theta}$:

$$\hat{\theta}_{i+1} = \hat{\theta}_i + \Delta\theta_i$$

(4) Test if convergence has occurred. If not, go to point 2 and put $i = i + 1$.

The identifiability of the unknown parameters can be studied by means of one subprogram which calculates the sensitivity functions and another which performs a curvature calculation. All time functions, such as measurements, predicted measurements, innovations and estimated state variables, can be presented on the graphic display. Hard copies are easily obtained. Further details are given in Jenssen (1978).

EXKALM

EXKALM is based on the model description given in eqn. (2) and a time varying Kalman filter where the state vector is augmented with the parameter vector, $\hat{\theta}_k$. The parameter estimation problem is thus transformed to a state estimation problem. The structure of the augmented Kalman filter is shown in Fig. 6 and the estimation algorithm is given in Appendix 2. The program is described in Onshus (1978).

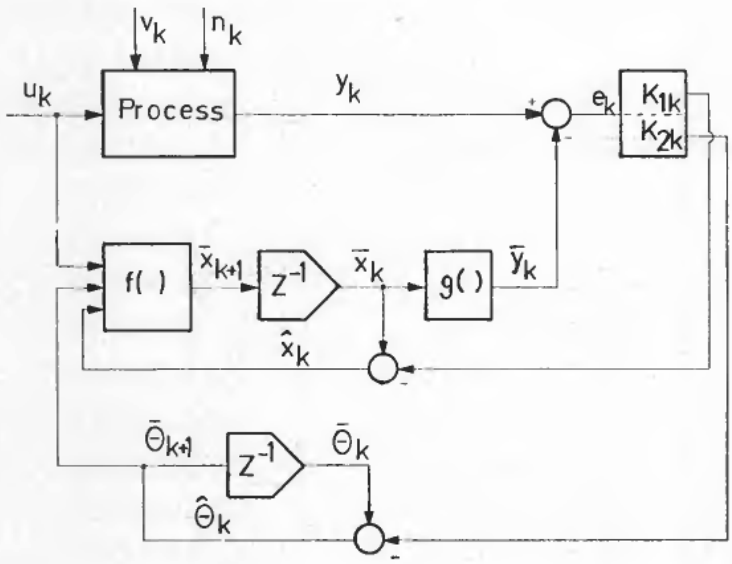

Figure 6. Augmented Kalman filter. $\bar{}$ and $\hat{}$ denote predicted and estimated values respectively.

M-L

This program estimates parameters in single output canonical models using the maximum likelihood method. Transfer function models, difference equation models and state space models are considered.

The criterion to minimize is

$$J = \frac{1}{2} \sum_{i=n+1}^{N} \epsilon_i^2 \tag{8}$$

with respect to a_i, B_i and c_i, where a_i, B_i and c_i are parameters in the ARMA model;

$$y_k + \sum_{i=1}^{n} a_i y_{k-1} = \sum_{i=1}^{n} B_i u_{k-1} + \sum_{i=1}^{n} c_i \epsilon_{k-1} + \epsilon_k \tag{9}$$

The minimization problem can be solved by using Lagrange multipliers or sensitivity functions.

In the first case, the gradient method will have the form

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - A_k \frac{\partial H}{\partial \boldsymbol{\theta}_k^{\mathrm{T}}} \tag{10}$$

where H is the hamiltonian function and A_k is a weighting matrix.

Using sensitivity functions, the weighting matrix is given by

$$A_k = \frac{\partial^2 J}{\partial \boldsymbol{\theta}_k \partial \boldsymbol{\theta}_k^{\mathrm{T}}}$$

The program operates very similarly to the MLPROG program described above. Further details are given in Opdal (1976 b).

OPTEX

Program OPTEX synthesizes test signals for system identification. The model description is given by

$$\left.\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\theta}),\ t \in [kT, (k+1)T] \\ \boldsymbol{y}(k) &= \boldsymbol{g}(\boldsymbol{x}(kT), \boldsymbol{\theta}) + \boldsymbol{\omega}(k) \end{aligned}\right\} \tag{11}$$

where T is the sampling interval and $\boldsymbol{\omega}(k)$ is a gaussian white noise sequence with zero mean value. The amplitude of $\boldsymbol{u}(k)$ is constrained to

$$\boldsymbol{a} \leqslant \boldsymbol{u}(k) \leqslant \boldsymbol{b}$$

and $\boldsymbol{u}(k)$ is assumed to have a user specified number of allowed levels.

The performance indices under consideration are

$$\left.\begin{aligned} J_{\mathrm{D}}(N) &= -\det(M(N)) + \sum_{k=1}^{N} h(\boldsymbol{x}(k)) \\ J_{\mathrm{A}}(N) &= \mathrm{tr}\,(M^{-1}(N)) + \sum_{k=1}^{N} h(\boldsymbol{x}(k)) \end{aligned}\right\} \tag{12}$$

where M is the Fisher's information matrix at time NT and $h(\cdot)$ is a scalar penalty function that makes it possible to include state constraints into the optimization problem. Further details are given in a paper by Kristoffersen and Sælid (1979).

4.3. *Control system design*

The control package consists of programs based on multivariable time domain and frequency domain methods.

RIC

RIC solves the discrete optimal control problem given by eqn. (4) and the criterion

$$J=\sum_{k=0}^{N} x_{k+1}^{T} Q x_{k+1} + u_k^{T} P u_k \tag{13}$$

The solution is given by

$$u_k = -Gx_k \tag{14}$$

where G is the limit of G_k when $N\to\infty$.

$$G_k=(P+\Delta^{T} R_{k+1}\Delta)^{-1}\Delta^{T} R_{k+1}\Phi \tag{15}$$

and R is determined from the Riccati equation

$$R_k=[\Phi-\Delta G_k]^{T} R_{k+1}[\Phi-\Delta G_k]+G_k^{T} P G_k+Q \tag{16}$$

Because of the duality between optimal control and optimal estimation, the program can be used to solve the optimal estimation problem as well (Opdal 1976 a).

MULFRAP

MULFRAP contains programs for designing multivariable controllers in the frequency domain. MULFRAP is based on the theory of multivariable frequency response analysis. Rosenbrock's inverse Nyquist array method (INA) and MacFarlane's characteristic locus method (CLM) are implemented. The theoretical background can be found in Rosenbrock (1969) and in MacFarlane and Belletrutti (1973). Figures 2 and 3 show the program structure. The most important subprograms are as follows:

- CLOC: This program computes the characteristic loci of a system. The result is given in the complex plane or as the amplitude of the characteristic loci and the corresponding alignment function.
- NYQUI: This program computes the direct or the inverse Nyquist array of a system.
- GERSHG: This program computes the row-wise and the column-wise Gershgorin circles of the open-loop system transfer matrix.

In addition, there are subprograms for manipulation of polynomial matrices and a display program for presentation of the results on video colour graphics. The MULFRAP program is further described by Stuan (1978).

4.4. *Miscellaneous*

Matrix calculations, CAL

CAL performs operations on matrices. The usual arithmetic operations are implemented, as well as computation of eigenvalues, transition and transmission matrices, etc. The matrices are placed in a stack which makes it simple to calculate expressions using reverse Polish notation. The matrices are fetched from the data base file or the terminal. The results are stored on the data file or presented on the terminal. Further details are given in Opdal (1977 a).

Handling of time series, TSP

In the TSP-package, there are programs for time series analysis (Tyssø 1978 b) and data handling, (Onshus 1977). By means of these programs, the user can easily operate on the data files (filtering, editing, scaling, etc.) and perform simple statistical analysis, for instance compute correlation functions and power spectra.

4.5. *Applications and experiences*

The CYPROS system has already proved important in the Institute's research activities, and postgraduate students have found it to be a powerful tool in their studies. Engineers from industry and from other research institutes have turned their attention to interactive program systems and more of the CYPROS packages have already been transferred to other computer systems. The users represent a broad field in engineering cybernetics and many interesting applications have been reported.

Sælid and co-workers (1979) have utilized CYPROS in connection with identification and control studies of a fluidized bed reactor for roasting of pyrite. The reactor model is described by a fifth-order, non-linear multivariable state space model, and unknown parameters have been estimated by means of the maximum likelihood program MLPROG. The control system is based on optimal control theory and Kalman filtering techniques and it is designed by means of the RIC program. The reactor model and the control system are closely examined by extensive use of the simulation program SIM. The control strategy is implemented on a digital computer and results from the real plant in operation show a very satisfactory control system performance.

Aam and Fjeld (1978) have applied the maximum likelihood program MLPROG to estimate unknown parameters in a hydrological model of a catchment area in Norway. The catchment model is described by a non-linear state space model of the fifth order. The estimated run-offs show good agreement with the observed run-offs and the results clearly indicate that it is possible to obtain improved control of water resources as well as better economy and efficiency in the running of power stations.

Opdal (1977 c) has developed a large, real time drilling well control simulator based on the simulation program SIM. The simulator consists of mathematical models describing the oil drilling process and the control systems. An operator's panel is interfaced to the computer, giving a very similar situation to a real drilling rig.

Onshus (1979) has applied the extended Kalman filter program for estimation of parameters in a wave model, representing the force acting on a power-buoy. It is planned to use the power-buoy in a future wave power station. The model is described by a fourth-order non-linear model, with two unknown parameters that have to be estimated on-line. An optimal phase control strategy is designed and, by use of simulation studies, it is very convenient to investigate how parameter variations in the estimator and the controller influence the performance of the control system. The control system is implemented on a small scale power-buoy operating in a hydro-laboratory.

In order to reduce the problems in connection with transfer of program packages, standard FORTRAN IV is used as programming language (USA ANS X3.9, 1966). But experiences from the different installations show that most FORTRAN IV compilers do not follow the standard specifications. Some modifications and extensions are often made and it turns out that handling of Hollerith strings and file handling cause special difficulties. In such cases, it is required that the program receiver knows his computer system and, especially, the FORTRAN compiler, in such a way that the necessary modifications can be performed within a reasonable period of time.

Some of the more advanced program packages require a certain theoretical insight if they are to be utilized in an effective and correct way. Therefore, a successful program transfer is also strongly dependent on the receiver's theoretical background. It is recommended that user courses are organized to ensure a required user background during the transfer of program packages.

5. Conclusions

This paper describes an interactive program system, CYPROS, which contains program packages to be used in connection with simulation, control and identification studies. The motivation and the basic ideas behind CYPROS are discussed and the structure of the system, as well as the hardware configuration, are described. The program system has proved to be a useful tool in various fields of application, and has been transferred to industry and other research institutes. Although the packages have been developed and modified during a five year period, they are continuously being revised and new programs are included. This is to meet the requirements and suggestions from the users and to simplify maintenance and improve portability.

Appendix 1

Example of interaction organization

In order to demonstrate briefly how the interaction is organized, the simulation program SIM is chosen as an example. We assume that the user has specified his/her models as FORTRAN subroutines and that these are loaded together with the main program. Further, a symbolic file (data file), containing all the characteristic data of the models (number of variables, names of variables and parameters, numerical values of variables and parameters) is also needed as input. In the following example the underlined characters are given by the user.

Initially the program SIM asks the user to specify the name of the data file:

FILE NAME : <u>MODEL-DATA</u>

The program then asks the user to specify the parameter operation. By typing HELP the program will produce a list of the various parameter operations.

—PARAMETER OPERATION: <u>HELP</u>

CHANGE PARAMETER =C

VALUE OF PARAMETER=V

WRITE =W

READ =R

CONTINUE =0

—PARAMETER OPERATION: <u>0</u>

Then the user has to specify the solution method:

```
—SOLUTION METHOD: HELP
CONTINUED SIMULATION=C
EULER               =E
EULER AUTOMATIC     =EA
KUTTA-MERSON        =K
KUTTA-MERSON AUTO.  =KA
SIRKUS†             =S
SIRKUS AUTOMATIC    =SA
DISCRETE TIME SYSTEM =D
STOP PROGRAM        =0
OUTPUT OPERATION    =O
PARAMETER OPERATION =P
```

If the user wants to apply the Kutta–Merson method, he or she types K, and the program asks for simulation time interval and the step size:

```
—SOLUTION METHOD    : K
TIME INTERVAL (TI, TF): 0, 10
STEP SIZE           : 0·1
```

The command C, (CONTINUED SIMULATION) means that the simulation will be continued in a further time interval. The final values of the previous time interval will be used as initial conditions. The user only specifies the new final time. O (OUTPUT OPERATION) means jumping to an output operation, while P (PARAMETER OPERATION) means returning to a parameter operation (see above).

When the simulation is performed, the program asks the user to specify the output operation:

```
—OUTPUT OPERATION: HELP
DISPLAY ON TV          =D
PLOT ON VERSATEC‡      =P
WRITE ON TERM‡ OR L-P‡=W
STORE ON DATAFILE      =S
FETCH FROM DATAFILE    =F
CONTINUE               =0
—OUTPUT OPERATION: 0
```

† SIRKUS is a method developed by (Nørset 1974) for solving stiff differential equations.

‡ VERSATEC is the hard-copy unit, TERM is the terminal and L-P is the line printer. It is also possible to make diagrams by means of a graphic colour plotter.

. . . .

—PARAMETER OPERATION: 0

—SOLUTION METHOD: 0

DO YOU REALLY WANT TO STOP

THE PROGRAM?: Y

STOP

After presenting a list of the various output operations, the program returns to the initial stage of the program, the parameter operation facility.

Appendix 2

Extended Kalman filter program, EXKALM

The estimation algorithm is as follows:

(1) Compute predicted measurement and the linearized measurement matrix:

$$\bar{y}_{k+1} = g(\bar{x}_{k+1}, u_{k+1}, \bar{w}), \; D_{k+1} = \left.\frac{\partial g}{\partial x^{\mathrm{T}}}\right|_{\substack{x=\bar{x}_{k+1} \\ w=\bar{w}}}$$

(2) Compute the predicted state error covariance matrix:

$$\Delta\bar{X}_{k+1} = \Phi_k \Delta X_k \Phi_k^{\mathrm{T}} + V_k$$

(3) Compute the filter gain matrix:

$$K_{k+1} = \Delta\bar{X}_{k+1} D_{k+1}^{\mathrm{T}} (D_{k+1} \Delta\bar{X}_{k+1} D_{k+1}^{\mathrm{T}} + W_{k+1})^{-1}$$

W: measurement noise covariance matrix.

(4) Compute the *a posteriori* state error covariance:

$$\Delta X_{k+1} = (I - K_{k+1} D_{k+1}) \bar{X}_{k+1} (I - K_{k+1} D_{k+1})^{\mathrm{T}} + K_{k+1} W_{k+1} K_{k+1}^{\mathrm{T}}$$

(5) Compute the estimated state vector:

$$\hat{x}_{k+1} = \bar{x}_{k+1} + K_{k+1}(y_{k+1} - \bar{y}_{k+1})$$

(6) Increase k, $k = k+1$.

(7) Compute the predicted state vector and the system matrix:

$$\bar{x}_{k+1} = f(\hat{x}_k, u_k, \bar{v}_k), \quad \Phi_k = \left.\frac{\partial f}{\partial x^{\mathrm{T}}}\right|_{\substack{x=\hat{x}_k \\ u=u_k \\ v=\bar{v}_k}}$$

(8) Repeat from (1).

The filter structure is shown in Fig. 6.

From the main program, the user can choose among several commands. The most important commands are

EXKALM: solves the Kalman filter equations as described above.

FIXED : runs the Kalman filter with constant K.

SIMPL : runs the Kalman filter with a user specified algorithm for K.

NOCOV : runs the Kalman filter with a user specified algorithm for $\Delta\bar{X}$

BALTRA : simulates the model with the given inputs, but without Kalman filter included.

LOOK : changes program parameters and handles the data files.

DISPLAY : displays variables on video colour graphics.

References

AAM, S., and FJELD, M. (1978). An implementation of estimation techniques to a hydrological model for prediction of run-off to a hydroelectric power station. Report from the Norwegian Research Institute of Electricity Supply, Trondheim, Norway.

ELMQVIST, H., TYSSØ, A., and WIESLANDER, J. (1976). Programming and documentation rules for subroutine libraries—designed for the Scandinavian Control Library. Nordforsk, The Scandinavian Council for Applied Research, Stockholm.

JENSSEN, N. A. (1978). MLPROG, and interactive program for parameter estimation. Report STF48 F78017, SINTEF, Trondheim, Norway.

KRISTOFFERSEN, K., and SÆLID, S. (1979). Computer aided design of suboptimal test signals for system identification. Paper presented at the IFAC Symposium, Computer Aided Design of Control Systems, Zürich.

LEMMENS, W. J. M., and VAN DEN BOOM, A. J. W. (1979). Interactive computer programs for education and research: A Survey. *Automatica*, **15**, 113–121.

MACFARLANE, A. G. J., and BELLETRUTTI, J. J. (1973). The characteristic locus design method. *Automatica*, **9**, 575–583.

NØRSET, S. P. (1974). Semi explicit Runge–Kutta methods. Mathematics and Computation No. 6, 1974. Department of Mathematics, University of Trondheim, Norway.

ONSHUS, T. (1977). Program package for time-series analysis, User's guide. Report STF48 F77034, SINTEF, Trondheim, Norway; (1978). EXKALM, an interactive program system for Kalman filtering. Report STF48 A78005, SINTEF, Trondeim, Norway; (1979). Optimal phase control of a power buoy. Report STF48 F79038, SINTEF, Trondheim, Norway.

OPDAL, J. (1976 a). RIC, interactive program for stationary optimal estimation and control. Report STF48 A76044, SINTEF, Trondheim, Norway; (1976 b). Maximum likelihood estimation of parameters of linear canonical models for single output dynamical systems. Report STF48 A76043, SINTEF, Trondheim, Norway; (1977 a). CAL—an interactive program for matrix calculations. Report STF48 F77025, SINTEF, Trondheim, Norway; (1977 b). Modelling and simulation of drilling well pressure when circulating out a gas kick. Report STF48 F77028, SINTEF, Trondheim, Norway; (1977 c). SIM, an interactive program for dynamic model simulation. Report STF48 A77026, SINTEF, Trondheim, Norway.

PEDERSEN, J. O., PØHNER, F., and SOLHEIM, O. A. (1972). Computer aided design of multivariable control systems. Paper presented at the 5th IFAC Congress, Paris.

ROSENBROCK, H. H. (1969). Design of multivariable control systems using the inverse Nyquist array. *Proc. Instn elect. Engrs*, **116**, 1929–1936.

STUAN, V. (1978). MULFRAP—An interactive program for multivariable frequency response analysis. Report 78-113-W, NTH, Division of Engineering Cybernetics, Trondheim, Norway.

SÆLID, S., JENNSEN, N, A., LINDSTAD, T., and KOLBEINSEN, L. (1979). Modelling, identification and control of a fluidized bed reactor. Paper presented at the 5th IFAC Symposium, Identification and System Parameter Estimation, Darmstadt.

TYSSØ, A. (1978 a). CYPROS, cybernetic program library. Report 78-112-W, NTH, Division of Engineering Cybernetics, Trondheim, Norway; (1978 b). Time series analysis—a program package documentation. Report 78-123-W, NTH, Division of Engineering Cybernetics, Trondheim, Norway.