CLPP: A User-Friendly Platform for Nonlinear Robust Observer Design

M. Alamir [*] **P. Bellemain** [*] **L. Boillereaux** [**] **I. Queinnec** [****]
M. Titica [**] **N. Sheibat-Othman** [***] **C. Cadet** [*] **G. Besançon** [*]

[*] *CNRS-University of Grenoble*
Gipsa-Lab, Control Systems Department.,
Rue de la Houille Blanche, Domaine Universitaire, ST Martin d'Hères,
France (Tel: +33 4-76-82-63-26), (e-mail: mazen.alamir@inpg.fr)
[**] *GEPEA, CRTT, 37 bd de l'Université, BP 406, 44602 Saint-Nazaire*
Cedex, France.
[***] *Université de Lyon, Univ. Lyon 1, CPE Lyon, CNRS, UMR 5007,*
Laboratoire dAutomatisme et de Génie des Procédés (LAGEP), 43 Bd
du 11 Novembre 1918, F-69616 Villeurbanne, France
[****] *CNRS ; LAAS ; 7 avenue du Colonel Roche, F-31077 Toulouse,*
France
Université de Toulouse ; UPS , INSA , INP, ISAE ; LAAS ; F-31077
Toulouse, France

Abstract: In this paper, a new user-friendly platform for robust observer design is presented. The aim of this software is to render the process of observer design as simple as possible for process practitioners and researchers involved in control or supervision tasks. The platform is dedicated to laboratory scale processes in which key variables are not directly measured and therefore need to be estimated. Moreover, model mismatches and uncertainties can be potentially recovered. The tool can also be used to analyze the feasibility of the related inverse problem for a given choice of the sensors, the sampling time, the observation window, etc. making it a precious tool to design the instrumentation of the process. A particularly useful feature for researchers is the possibility to automatically generate a Matlab *S-function* that may be connected to the user's own control/diagnosis modules to perform the estimation task. The software is intended to be freely available (by simple request) for research and educational purposes by mid 2010.

Keywords: Software Sensor; Moving-Horizon Observer; User-Friendly; Nonlinear Systems; Optimization.

1. INTRODUCTION

Chemical and biological processes are generally characterized by highly nonlinear dynamics that involve badly known parameters. They also suffer from the lack of sensors making the use of state observers mandatory.

While many analytic observer design methods are conceptually available for nonlinear systems (Gauthier et al. [1992], Slotine et al. [1987]), the need for structural properties that have to be satisfied by the system model tremendously reduces the class of systems to which analytic observers can be applied.

On the contrary, optimization-based observers (Michalska and Mayne [1995], Alamir [1999]) that reconstruct the state by minimizing output prediction error related cost are particularly suitable as they enable complex modeling as well as state constraint handling.

The French National Research project ANR [1] -CLPP [2] has been initiated based on the observation according to which, many process researchers, when dealing with their own works on the control, the supervision and/or the optimal design of their processes, are quite frequently faced with the problem of observer design. For non specialists, this task is cumbersome because analytic observers rarely apply while optimization-based approaches need some technicalities that may be time-consuming and difficult to master.

The Anr-clpp project ambition is to offer a user-friendly tool that renders these difficulties transparent for the process researchers that can therefore focus on their own research effort. This paper describes this tool which is intended to be freely available to academic research groups by mid 2010, namely, the official term of the Anr-clpp project.

This paper is organized as follows: First, the heart of the estimation algorithm is briefly described in section 2. A

[⋆] This work has been partially supported by the French National Research Agency (Anr-clpp)

[1] Agence Nationale pour la Recherche
[2] from the french "Capteurs Logiciels Plug & Play"

typical working session is then detailed in section 3 in order to describe the basic features of the software. This is done using the example of a waste-water treatment process.

2. THEORETICAL BACKGROUND

CLPP adresses systems that are governed by an Ordinary Differential Equation (ODE) of the form:

$$\dot{x} = f(x, u, w) \quad (x, u, w) \in \mathbb{R}^n \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_w} \tag{1}$$

$$y = h(x, u, w) \quad y \in \mathbb{R}^{n_y} \tag{2}$$

where x is the state vector, u is the vector of measured inputs, w is the vector of unmeasured disturbances while y is the measured output. In order to perform the estimation task, CLPP uses moving-horizon strategy (Michalska and Mayne [1995]) in which the measurements collected during the past observation horizon $[t - T, t[$ (where t is the current time while T is called the observation horizon) are used to recover the value of the unknown variables. The latter are the state $x_0 = x(t - T)$ at the beginning of the observation horizon and the disturbance/uncertainty profile $w(\cdot)$ over $[t - T, t]$. In order to define a finite dimensional optimization problem, the user has to provide a *parametrization* map:

$$[x_0, w(\tau)] = param(p, \tau) \quad \textit{for all } \tau \in [t - T, t] \tag{3}$$

that gives a finite dimensional parametrization of the unknowns. Typically, the first n components of p are used to represent the unknown states while the remaining components are used to describe the temporal evolution of w over $[t - T, t]$. To cite just an exemple, for ramp-like temporal profiles of w, $2 \cdot n_w$ components may be used to describe the evolution of $w(\cdot)$ over $[t - T, t]$. Note also that discontinuous profile can also be used in which the switching instants may be used as unknown parameter, etc. See (Alamir [2008b]) for more details.

Based on the measurements that are acquired during the time interval $[t - T, t]$, an optimization problem can be defined in the decision variable p. The cost function represents the difference between the predicted output and the effectively measured values. More precisely:

$$J^{(t)}(p) := \sum_{i=1}^{n_y} \Big[\sum_{k \in \mathcal{K}_i(t)} \big| y_i(t_k) - y_i^p(t_k|p) \big|^2 \Big] \tag{4}$$

where $\mathcal{K}_i(t)$ are indices of instants $t_k \in [t - T, t]$ where a measurement of the component y_i is available. The reason for this rather non standard definition is that in real context, the rates of acquisition of the different measurements are never the same. $y_i^p(t_k|p)$ is the predicted output based on the initial state and the disturbance profile given by the parameter vector p through (3).

During each observer updating period $[\tau_j, \tau_{j+1}]$ (where $\tau_j = j \cdot \tau_o$), a finite number q of function evaluations are allowed in order to look for a minimum of the cost function $J^{(\tau_j)}(\cdot)$ starting from an initial guess $p^+(\tau_{j-1})$ that is compatible with the past estimate $p(\tau_{j-1})$ leading to the following updating process for the dynamic variable p:

$$p(\tau_j) := \mathcal{S}^q\big(p^+(\tau_{j-1})\big) \tag{5}$$

where $\mathcal{S}$ denotes an iteration of some optimization algorithm and $\mathcal{S}^q$ denotes successive iterations of $\mathcal{S}$ that involves q function evaluations. In its current version, CLPP already implements several Gradient-free direct search algorithms (*Simplex, Torcszon, etc.*). Such algorithms enable non smooth inverse problems to be tackled, avoid asking the user to provide analytical gradient or the numerical troubles associated to the computation of the sensitivity matrices.

Using the current value $p(\tau_j)$ in (3) gives the current estimations $\hat{x}(\tau_j - T|p(\tau_j))$ as well as the uncertainty profiles $\hat{w}(\cdot|p(\tau_j))$ on $[\tau_j - T, \tau_j]$ which gives the estimation $\hat{x}(\tau_j)$ by integrating the system model (1).

Note that the cost function (4) is generally non convex and the problem of avoiding local minima is crucial. In order to enhance global convergence of the iterations, CLPP implements the singularity avoidance technique proposed in (Alamir [2008a], Alamir et al. [2009]). Briefly speaking, this technique involves iterations that switch between several cost functions defined by:

$$J_\sigma^{(t)}(p) := \sum_{i=1}^{n_y} \Big[\sum_{k \in \mathcal{K}_i(t)} \phi_\sigma(t_k) \cdot \big| y_i(t_k) - y_i^p(t_k|p) \big|^2 \Big] \tag{6}$$

where $\big\{\phi_\sigma(\cdot)\big\}_{\sigma \in \{1,\dots,n_\phi\}}$ is a family of weighting profiles. The rationale behind this is that for all σ and in the absence of measurement noise, the global solution is shared by all the resulting cost functions. This makes it possible to define an iterative scheme that exploits this property in order to avoid potential accidental singularities.

3. A TYPICAL WORKING SESSION

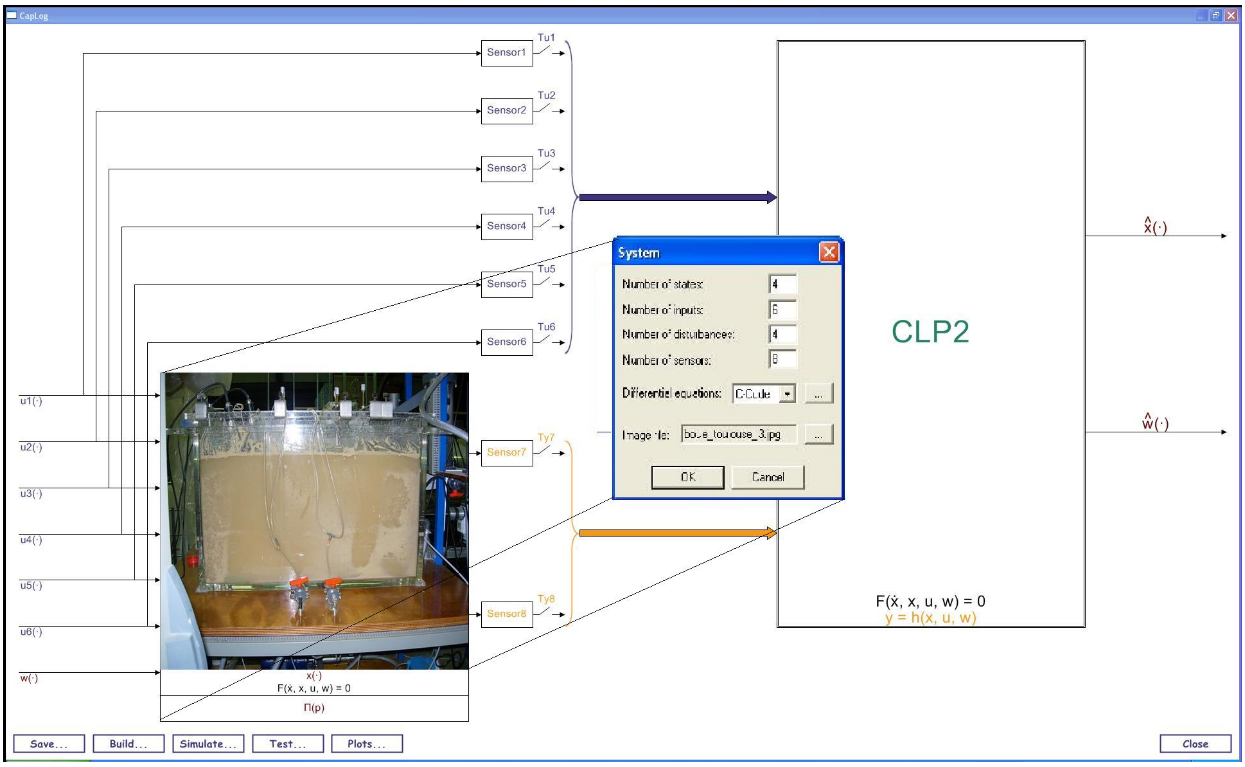
In this section, a typical working session on CLPP is described in order to better understand the steps leading to the observer construction. The problem of the simultaneous estimation of the state and the model parameters of an activated sludge (Gomez-Quintero et al. [2000]) is used as a support for illustration. It is needless to say that due to the lack of space, only a very brief presentation of the problem can be proposed here in order to concentrate on the software presentation.

Note that CLPP is a tool that generates model-based observation algorithm. Therefore, having a dynamic model of the process is a precondition to the use of CLPP. The reduced model proposed in (Gomez-Quintero et al. [2000]) for the activated sludge process takes the following form:

$$\dot{x} = f(x, u, w) \quad (x, u, w) \in \mathbb{R}^4 \times \mathbb{R}^6 \times \mathbb{R}^4 \tag{7}$$

$$y = h(x, u, w) = (x_2, x_4) \tag{8}$$

where $x = (S_S, S_{NO_3}, S_{NH_4}, S_{O_2})$ in which S_S is the biodegradable substrate concentration, S_{NO_3} is the nitrate concentration, S_{NH_4} is the ammonia concentration while S_{O_2} is the dissolved oxygen. The vector of inputs u gathers many flow-rates and concentration related information while the uncertainty vector w contains the unknown model parameters that have to be identified on line. Only



System

Number of states: 4

Number of inputs: 6

Number of disturbances: 4

Number of sensors: 8

Differentiel equations: C-Code

Image file: boue_toulouse_3.jpg

OK Cancel

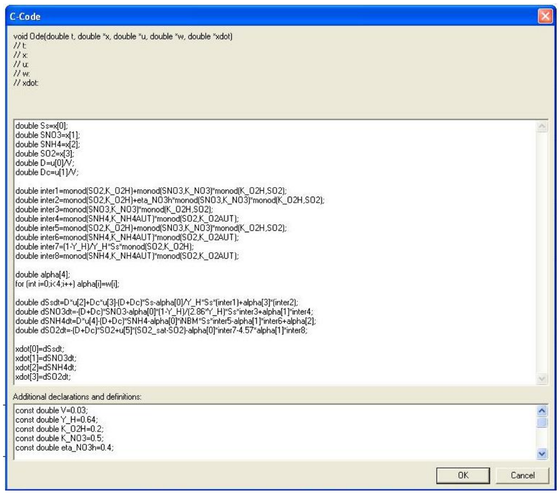
Fig. 1. CLPP *environment for the activated sludge simultaneous estimation problem*

S_{NO_3} and S_{O_2} are measured leading to the measurement vector y. It is worth noting that the above model is an oversimplified model that has been precisely derived for estimation purposes.

A typical CLPP working session comprises the following steps:

(1) Definition of the problem dimensions: n, n_u, n_w and n_y.
(2) Definition of the system's ODE's.
(3) Definition of the sensors physical laws.
(4) Definition of the parametrization map *param* invoked in (3)
(5) Definition of the input profiles
(6) Simulation of the so obtained model. This steps enables the user to check whether the dynamic system is well defined by inspecting the state, output and disturbance profiles for a given parameter vector p.
(7) Definition of the observer's related issues such as the observation horizon (T), the optimizer ($\mathcal{S}$), the number of function evaluations (q) and the number of weighting profiles (n_ϕ).

Figure 1 shows a schematic view of the CLPP environment. The dialog box that is shown enables the problem's dimensions to be defined as well as the set of ODE's governing the system dynamics. The latter can be defined by clicking on the *differential equation* button of the *System* dialog-box (Figure 1). The window shown in Figure 2 is then opened and the user can enter the C code of the ODE's in the body of the head-predefined function. Figure 2 shows the ODE's in the case of the activated sludge example (7). Note that the lower window in Figure 2 is devoted to any dependencies and/or constant declarations that may be needed in the computation of the ODE's.



```
void Ode(double t, double *x, double *u, double *w, double *xdot)
// t:
// x:
// u:
// w:
// xdot:
```

```
double Ss=x[0];
double SNO3=x[1];
double SNH4=x[2];
double SO2=x[3];
double D=u[0]/V;
double Dc=u[1]/V;

double inter1=monod(SO2,K_O2H)+monod(SNO3,K_NO3)*monod(K_O2H,SO2);
double inter2=monod(SO2,K_O2H)+eta_NO3h*monod(SNO3,K_NO3)*monod(K_O2H,SO2);
double inter3=monod(SNO3,K_NO3)*monod(K_O2H,SO2);
double inter4=monod(SNH4,K_NH4AUT)*monod(SO2,K_O2AUT);
double inter5=monod(SO2,K_O2H)+monod(SNO3,K_NO3)*monod(K_O2H,SO2);
double inter6=monod(SNH4,K_NH4AUT)*monod(SO2,K_O2AUT);
double inter7=(1-Y_H)/Y_H*Ss*monod(SO2,K_O2H);
double inter8=monod(SNH4,K_NH4AUT)*monod(SO2,K_O2AUT);

double alpha[4];
for (int i=0;i<4;i++) alpha[i]=w[i];

double dSsdt=D*u[2]+Dc*u[3]-(D+Dc)*Ss-alpha[0]/Y_H*Ss*(inter1)+alpha[3]*(inter2);
double dSNO3dt=-(D+Dc)*SNO3-alpha[0]*(1-Y_H)/(2.86*Y_H)*Ss*inter3+alpha[1]*inter4;
double dSNH4dt=D*u[4]-(D+Dc)*SNH4-alpha[0]*iNBM*Ss*inter5-alpha[1]*inter6+alpha[2];
double dSO2dt=-(D+Dc)*SO2+u[5]*(SO2_sat-SO2)-alpha[0]*inter7-4.57*alpha[1]*inter8;

xdot[0]=dSsdt;
xdot[1]=dSNO3dt;
xdot[2]=dSNH4dt;
xdot[3]=dSO2dt;
```

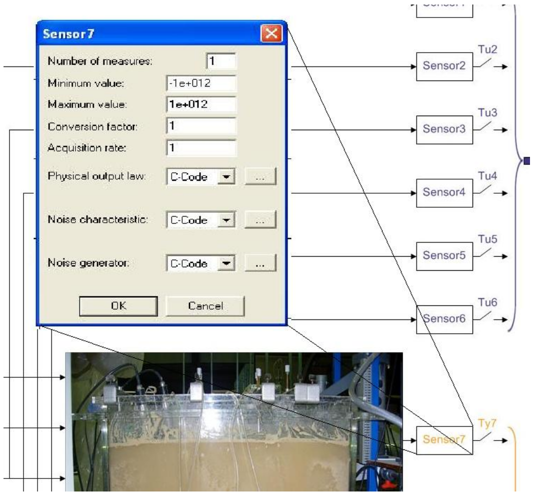
Additional declarations and definitions:

```
const double V=0.03;
const double Y_H=0.64;
const double K_O2H=0.2;
const double K_NO3=0.5;
const double eta_NO3h=0.4;
```

Fig. 2. *C-program defining the system equations (7)*

Once the ODE's are defined, the output laws can be entered through the definition of the sensor blocks. Clicking on each sensor block (see Figure 1) opens a definition window such as the one shown in Figure 3. Here again, clicking on the *Physical Output law* button opens an edition window in which a head-predefined C code of the corresponding $h_i(x, u, w)$ can be inserted together with the conversion rate and the acquisition rate that may be sensor dependent.

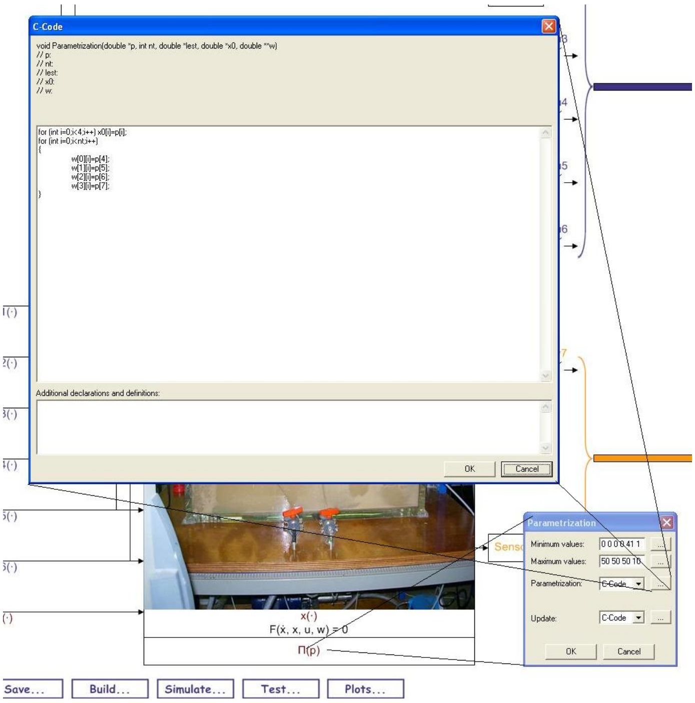
The parametrization map *param* can be defined by clicking on the $\Pi(p)$ button (see Figure 1). This opens the small window depicted in Figure 4 in which the user first defines



Sensor7

Number of measures: 1
Minimum value: -1e+012
Maximum value: 1e+012
Conversion factor: 1
Acquisition rate: 1
Physical output law: C-Code
Noise characteristic: C-Code
Noise generator: C-Code

Fig. 3. *Dialog-Box enabling the first output sensor to be defined. Clicking on the Physical Output law button opens a edition window in which a C code of $h_1(x, u, w)$ can be inserted.*



C-Code

```
void Parametrization(double *p, int nt, double *lest, double *x0, double **w)
// p:
// nt:
// lest:
// x0:
// w:
```

```
for (int i=0;i<4;i++) x0[i]=p[i];
for (int i=0;i<nt;i++)
{
        w[0][i]=p[4];
        w[1][i]=p[5];
        w[2][i]=p[6];
        w[3][i]=p[7];
}
```

Additional declarations and definitions:

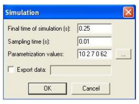
Parametrization

Minimum values: 0 0 0 0.41 1
Maximum values: 50 50 50 10
Parametrization: C-Code
Update: C-Code

Fig. 4. *Dialog-Box enabling the parametrization map param [see equation (3)]*

the minimum and maximum values of the unknown vector p. Then by clicking on the *parametrization* button of this windows, a head-predefined *C-code* window is opened enabling the user to include the script of the parametrization map *param* involved in (3). For the particular activated sludge problem, p is of dimension 8 with the first four components standing for the unknown initial state while the remaining components stand for the unknown parameters that are supposed here to be constant during the observation horizon.

At this stage, CLPP disposes of everything needed to simulate the system model. This can be done by clicking on the *simulate* button depicted on Figure 1. This opens the dialog box shown in Figure 5 where the simulation time, sampling period and the value of the parameter vector p can be entered by the user. Note also that by checking the *Export Data* check-box, the simulation results can be saved in text file for further use with more elaborated graphical softwares. Note however that some aspects of the plots can be customized using the dialog-box of the *plots* button of Figure 1.



Simulation

Final time of simulation (s): 0.25
Sampling time (s): 0.01
Parametrization values: 10 2 7 0 62
□ Export data:

Fig. 5. *Dialog-Box of the simulate button. This enables a simulation to be performed in order to check the model and sensors definitions.*

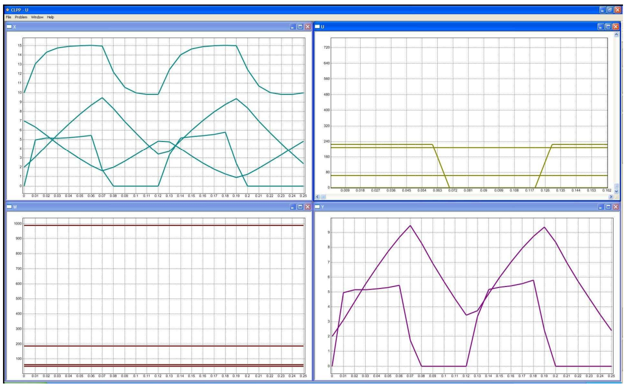
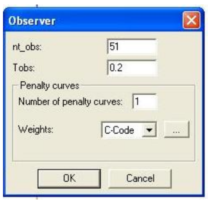


Fig. 6. *Example of simulation plots. The plotting options can be customized using the dialog-box of the plots button*

The definition of the observer parameters T and the different weighting curves $\phi_\sigma(\cdot)$ used in the singularities avoidance technique depicted above can be done by clicking on the *CLP2* box of Figure 1. The corresponding dialog-box is shown in Figure 7.

At this stage, the observation algorithm can be tested by clicking on the *Test* button of Figure 1. This opens the *test* dialog box (see Figure 8) in which the user can define the number of function evaluations and the initial value of the parameter vector p. Typically, this values must be different from the values used to produce the simulation in order to check the convergence of the estimation scheme. The *Solver Settings* button of the *Test* dialog box enables the solver and its parameters to be chosen by the user as depicted in Figure 8. This includes the observation window, the optimization algorithm and its parameters (initial trust region, minimum step, etc.)

Pressing on the *OK* button of the *Test* dialog box (see Figure 8) opens the *plot* window where the estimation results are shown together with a dynamic window showing the evolution of the cost function during the iteration (see Figure 9). At each sampling instant, the observer window is shifted and any new available measurement that might have been acquired since the last sampling instant are



Observer

nt_obs: 51

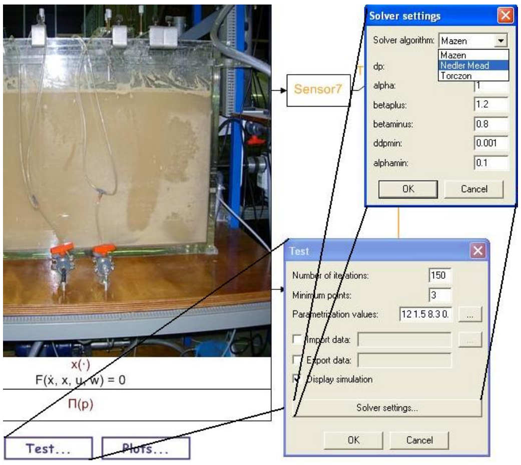
Tobs: 0.2

Penalty curves

Number of penalty curves: 1

Weights: C-Code

Fig. 7. *Dialog Box of the CLP2 block of Figure 1. This enables some the optimization options to be chosen for the observation algorithm.*



Solver settings

Solver algorithm: Mazen

dp:

alpha: 1

betaplus: 1.2

betaminus: 0.8

ddpmin: 0.001

alphamin: 0.1

Test

Number of iterations: 150

Minimum points: 3

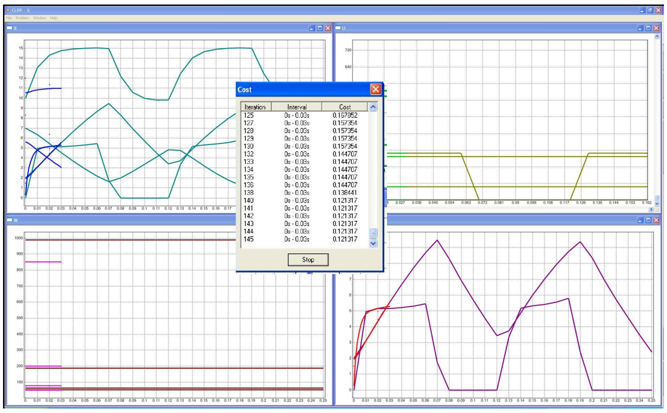
Parametrization values: 12 1.5 8.3 0.

□ Import data:

□ Export data:

☑ Display simulation

Fig. 8. *Dialog Box of the Test button of Figure 1. This enables the observation horizon T and the weighting curves ϕ_σ to be defined by the user.*



Cost

Iteration	Interval	Cost
125	0s - 0.03s	0.167852
127	0s - 0.03s	0.157354
128	0s - 0.03s	0.157354
129	0s - 0.03s	0.157354
130	0s - 0.03s	0.157354
132	0s - 0.03s	0.144707
133	0s - 0.03s	0.144707
134	0s - 0.03s	0.144707
135	0s - 0.03s	0.144707
136	0s - 0.03s	0.144707
138	0s - 0.03s	0.138441
140	0s - 0.03s	0.121317
141	0s - 0.03s	0.121317
142	0s - 0.03s	0.121317
143	0s - 0.03s	0.121317
144	0s - 0.03s	0.121317
145	0s - 0.03s	0.121317

Fig. 9. *The observer test view. The evolution of the estimated trajectories over the observation window is shown for the state, output and the disturbance vectors. The dynamic evolution of the value of the cost function is also displayed in the central window.*

taken into account in the computation of the new cost function.

Figure 10 shows the results of the simultaneous state and parameter estimation for the activated sludge example. Note that this set of plots is realized using MATLAB graphical tools using the data files that have been designated to export the session results. This can be done using the *Export data* check-box of the *Test* window (see Figure 8).

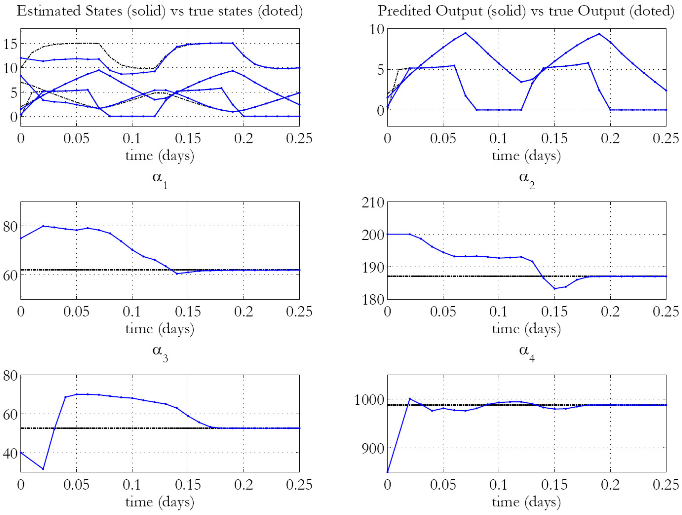


Fig. 10. *Validation results for the simultaneous state/parameter estimation of the activated sludge problem. This is a four state/ two measurement problem with 4 unknown parameters that are modeled in* CLPP *as constant disturbances to be estimated.*

4. FURTHER ISSUES

4.1 Measurement Handling

In this section, the term measurement concerns both the input and the output measurements. One of the nice features of CLPP is the ability to handle non synchronized measurements acquisition which is a quite recurrent situation in process industry. Indeed, some measurements need post-processing protocoles that are time consuming resulting in long time periods between two successive measurement availabilities. Moreover, since some of such operations involve human intervention, this leads to a non uniform sampling time.

CLPP offers two modes of measurement acquisition during the prototyping phase:

(1) **Internal acquisition mode**

In this mode, the measurements that are generated during the simulation phase are used in the observer test phase. Note that in this case, the acquisition rate for each measurement is defined by the *acquisition rate* parameter of the corresponding sensor (see Figure 3). Note that when using this mode, it is still possible to use different models for the observer and the simulation by including additional non zero uncertainty components in the uncertainty vector w for which the search domain used by the observer is $\{0\}$.

(2) **External acquisition mode**

In this mode, the measurement data are given by text files *data.txt* having the following structure:

```
%==============
% File Name
%==============
```

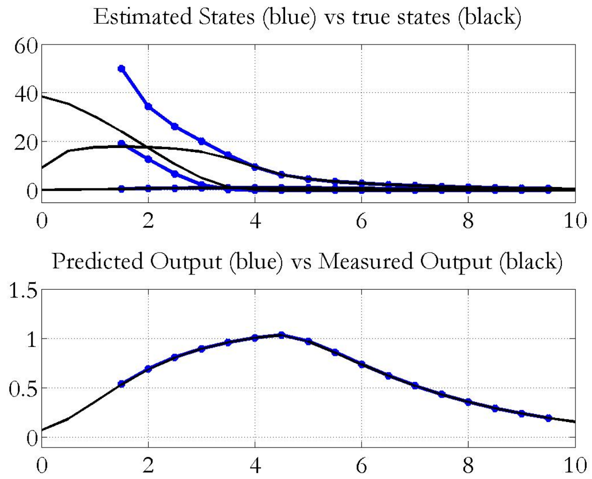


Fig. 11. *infeasibility detection of a particular inverse problem of simultaneous state and parameter estimation of hydrogen production model proposed in Fouchard et al. [2009]*

```
n1  t1  v1
n2  t2  v2
.   .   .
ni  ti  vi
.   .   .
nf  tf  vf
%=============
```

where $t_i \leq t_{i+1}$ are the instants where acquisition of at least one measurement is performed. *ni* is the corresponding sensor number while *vi* is the measurement value. A major advantage of this mode is that experimental acquisition can be used that are not generated by any mathematical model. CLPP can then be used to validate the integrated scheme of both a model and its corresponding observer.

4.2 Observability/Identifiability Analysis

In addition to its use for robust observer design, CLPP is a precious tool when one needs to study the feasibility of the corresponding inverse problem. As an example, Figure 11 shows the results of a simultaneous state and parameter estimation of a recently proposed hydrogen production model with Chlamydomonas reinhardtii (see Fouchard et al. [2009]). This is a typical situation where it is obvious that the inverse problem is, at least badly conditioned if not unfeasible. Indeed, the plots show that the predicted outputs are almost identical to the measurement (over the observation horizon) while the initial states at the beginning of the observation window are quite different. User may now increase the observation horizon or check different kind of measurement, etc.

5. CONCLUSION

In this paper, the software CLPP, a user-friendly software for fast prototyping of robust nonlinear observer is described. The technical difficulties are made transparent to the user. The latter has to feed the model equation, the sensor equation as well as high level parameters such as the observation window, the maximum number of function evaluation, etc. Different and/or irregular measurement acquisition rates can be handled. Once the observer is calibrated, the designed solution can directly be used in a Matlab/Simulink observer block for further use in diagnosis and/or control purposes.

REFERENCES

M. Alamir. Optimization based nonlinear observers revisited. *International Journal of Control*, 72 (13):1204–1217, 1999.

M. Alamir. On useful redundancy in dynamic inverse problems related optimization. In *Proceedings of the IFAC World Congress*, Seoul, South Korea, 2008a.

M. Alamir. *Nonlinear Moving-Horizon Observers: Theory & Real-Time Implementation.* In *Nonlinear Observers & Applications*, G. Besançon (Ed.), Lecture Notes in Control and Information Science. Springer-Verlag, 2008b.

M. Alamir, J. S. Welsh, and G. C. Goodwin. Redundancy vs multiple starting points in nonlinear systems related inverse problems. *Automatica*, 45:1052–1057, 2009.

S. Fouchard, J. Pruvost, B. Degrenne, M. Titica, and J. Legrand. Kinetic modeling of light limitation and sulfur deprivation effects in the induction of hydrogen production with chlamydomonas reinhardtii: Part i. model development and parameter identification. Biotechnology and Bioengineering, 102(1):232–245, 2009.

J. P. Gauthier, H. Hammouri, and S. Othman. A simple observer for nonlinear systems – application to bioreactors. IEEE Transactions on Automatic Control, 37, 1992.

C. Gomez-Quintero, I. Queinnec, and J. P. Babary. A reduced nonlinear model of an activated sludge process. In *Proceedings of the IFAC Int. Symp. ADCHEM*, pages 1037–1042, Pise, Italy, 2000.

H. Michalska and D. Q. Mayne. Moving-horizon observers and observer-based control. *IEEE Transactions on Automatic Control*, 40:995–1006, 1995.

J. E. Slotine, Hedrick J. K., and Misawa E. A. On sliding observers for nonlinear systems. Journal of Dynamics Systems, Measurements and Control, 109, 1987.