# IFTtune: a PID automatic tuning software tool ⋆

**Rogelio Mazaeda** * **César de Prada** **

* *Systems and Automation Department, School of Engineering, University of Valladolid, Valladolid, CP. 47011, Spain (e-mail: rogelio@cta.uva.es).*
** *Systems and Automation Department, School of Engineering, University of Valladolid, Valladolid, CP. 47011, Spain (e-mail: rogelio@cta.uva.es).*

**Abstract:** The regulatory layer of every process industry is made up of many decentralized loops using, almost exclusively, PID controllers. The task of keeping each of these devices correctly tuned is key to the efficient and safe operation of the factory but very demanding on the time of technical and maintenance personnel should it be carried out manually. The tool here described is designed to perform the automatic tuning of PID regulators facilitating its application in an industrial setting. The regulator parameters are updated starting from their current values so that a performance index is locally optimized using the Iterative Feedback Tuning algorithm. The tool is easily configurable and uses the widely adopted OPC industrial communication standard.

*Keywords:* PID controllers, close loop controllers, industrial control, communication protocols, computer software, configuration management, iterative methods, autotuners.

## 1. INTRODUCTION

It is a fact that the majority of controllers to be found at the so called regulatory layer in the process industry are of the PID type. It is also known that a large percentage of the existing PID loops are performing poorly due to bad controller's parameters tuning, implying a considerable aggregated economic loss. It is understandable, then, the importance that the problem of PID tuning has received over the years (Aström, 1988).

There are dozens of documented tuning rules (O'Dwyer, 2006). Some of the proposed methods follow an heuristic approach, some others presuppose the existence of a plant model, some strive to optimize a given performance criteria. A very relevant category is the one that offers the possibility of performing the tuning on line, with the PID already connected. The possibility of controller auto-tuning recognizes the fact that the loop performance can degrade in time. Difficult to predict variations in the plant are to be expected justifying the need of performing intermittent adaptation procedures.

For auto-tuning to be useful it should not require the knowledge of any explicit model of the plant and the procedure should be performed without disturbing too much the nominal workings of the loop. The adaptation procedure could be manually triggered by a human operator or, more desirable, by some automatic diagnosing system.

There are a number of interesting auto-tuning algorithms which are applicable to PIDs. The relay method (Aström and Hägglund, 1984) is widely known. Other important alternatives are the correlation based approach due to Karimi et al. (2003) and the Virtual Reference Feedback Tuning (VRFT) method as reported in Campi et al. (2002).

In this paper, a software system for automatic tuning of PID-type controllers based on Iterative Feedback Tuning (IFT) algorithm is described. The tool implements ideas which are relevant for the practical application of the method in an industrial setting, allowing the specification of different PID implementations and the definition of other arbitrarily described reduced order controllers. The software communicates with the plant using the OPC protocol which is widely supported in process industries.

The article is organized as follows: in section 2 the basic ideas for IFT are summarized, and some recent ideas and additions to the original method are recounted, section 3 describes the IFTtune tool features, stressing its open and amply configurable character, finally in section 4 some ideas for the future enhancement of the tool are put forward.

## 2. THE ITERATIVE FEEDBACK TUNIG ALGORITHM

Relevant to the PID tuning problem is the more general issue of finding the right parameters of a reduced order controller. A possible approach is to identify a low order model of the plant from the data collected on line in closed loop, followed by the design of a matching reduced controller. The strategy is sound despite the fact that the reduced model so obtained will inevitable exhibit a

⋆ The authors are grateful to for financial suport to the MICINN of Spain under the grant DPI2009-12806.

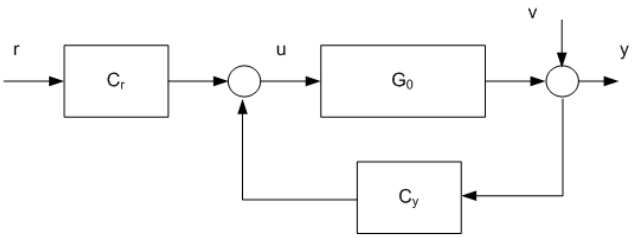


Fig. 1. Closed loop system

possible large error, since the purpose of identification is the achievement of a good enough closed loop performance. As extensively discussed in Gevers (2002), this type 'identification for control' solution can only be achieved by following an iterative scheme.

A further inquiry into the theoretical properties of the above strategy and its limitations concerning the lack of convergence results for the general case, suggested to Hjalmarsson et al. (1998) a clever new algorithm, eventually called Iterative Feedback Tuning, that hinges on the complete abandonment of the intermediate identification step, to directly perform the iterative optimization of the parameters of the reduced order controller.

In the following, a very succinct account of the IFT algorithm is given. Figure 1 shows a linear time-invarying SISO plant, $G_0$, with a two degree of freedom (2DoF) controller $\{C_r, C_y\}$, of known structure which is parameterized by the vector $\rho$. The equations (1) and (2) describe the closed loop system, with $y_t$ being the controlled variable, $u_t$ the controller output, and $v_t$ a perturbation which is going to be considered as a weakly stationary random variable. In equations (3) and (4) the corresponding close loop response and the sensitivity functions are respectively stated for further reference.

$$y = G_0 u_t + v_t \tag{1}$$

$$u_t = C_r(\rho) r_t + C_y(\rho) y_t(\rho) \tag{2}$$

$$T_0(\rho) = \frac{C_r(\rho) G_0}{1 + C_y(\rho) G_0} \tag{3}$$

$$S_0(\rho) = \frac{1}{1 + C_y(\rho) G_0} \tag{4}$$

The on-line tuning problem can be naturally put as that of finding the parameters of the controller in order to minimise a performance criterion $J$ (5). The cost $J$ could be defined as in (6), with $N$ the number of samples to consider. The term $\tilde{y}_t$, the difference between the current $(y_t)$ and the desired system $(y_d)$ output, is the control error, while $\lambda$ is a user chosen parameter which apprises the relative importance of the control effort $(u_t)$ in the cost. The terms $L_y$ and $L_u$ are frecuency dependent filters, which are going to be considered equal to unity, to simplify the explanation.

$$\rho^* = \text{argmin}_\rho J(\rho) \tag{5}$$

$$J(\rho) = \frac{1}{2N} \text{E} \left[ \sum_{t=1}^{N} \left( L_y \tilde{y}_t(\rho) \right)^2 + \lambda \sum_{t=1}^{N} \left( L_u u_t \right)^2 \right] \tag{6}$$

$$\rho_{i+1} = \rho_i - \gamma_i R_i^{-1} \frac{\partial J}{\partial \rho}(\rho_i) \tag{7}$$

The task so posed could be solved iteratively by following a Gauss-Newton scheme as in (7) provided that the gradients of the cost with respect to the parameters, $\partial J / \partial \theta$, are available. The term $R$ represents some appropriate approximation of the Hessian or the identity matrix for a search on the negative gradient direction.

$$\frac{\partial J}{\partial \rho} = \frac{1}{N} E \left[ \sum_{t=1}^{N} \tilde{y}_t(\rho) \frac{\partial \tilde{y}_t}{\partial \rho}(\rho) + \lambda \sum_{t=1}^{N} u_t(\rho) \frac{\partial u_t}{\partial \rho}(\rho) \right] \tag{8}$$

The gradient of the cost is represented by (8). It depends on the control error and on unknown gradients of the process variable error and of the control effort with respect to the vector $\rho$. Note that the term $\partial \tilde{y}_t / \partial \rho$ reduces to $\partial y_t / \partial \rho$ since the desired output $(y_d)$ is independent of $\rho$. The needed terms could be formally obtained from (1) and (2) using (3) and (4) to finally give (9) and (10).

$$\frac{\partial y}{\partial \rho}(\rho) = \frac{1}{C_r(\rho)} \left[ \left( \frac{\partial C_r}{\partial \rho}(\rho) - \frac{\partial C_y}{\partial \rho}(\rho) \right) T_0(\rho) r + \frac{\partial C_y}{\partial \rho} T_0(\rho) \left( r - y \right) \right] \tag{9}$$

$$\frac{\partial u}{\partial \rho}(\rho) = S_0(\rho) \left[ \left( \frac{\partial C_r}{\partial \rho}(\rho) - \frac{\partial C_y}{\partial \rho}(\rho) \right) r + \frac{\partial C_y}{\partial \rho} \left( r - y \right) \right] \tag{10}$$

As can be seen, the required gradients depend on the unknown plant $G_0$, by means of $T_0$ and $S_0$. The key discovery made by the IFT creators, which was suggested by the form of (9) and (10), is that the true gradients could be replaced by some unbiased estimation, namely (11) and (12) respectively, to be obtained from three closed loop experiments which are described in (13).

$$est \left[ \frac{\partial y}{\partial \rho}(\rho_i) \right] = \frac{1}{C_r(\rho_i)} \left[ \left( \frac{\partial C_r}{\partial \rho} - \frac{\partial C_y}{\partial \rho} \right)_{(\rho_i)} y^3 + \frac{\partial C_y}{\partial \rho}(\rho_i) y^2 \right] \tag{11}$$

$$est \left[ \frac{\partial u}{\partial \rho_i}(\rho_i) \right] = \frac{1}{C_r(\rho_i)} \left[ \left( \frac{\partial C_r}{\partial \rho} - \frac{\partial C_y}{\partial \rho} \right)_{(\rho_i)} u^3 + \frac{\partial C_y}{\partial \rho}(\rho_i) u^2 \right] \tag{12}$$

$$\begin{aligned}
&Exp.1: \\
&r_i^1 = r \\
&y_i^1(\rho_i) = T_0(\rho_i) r + S_0(\rho_i) v_i^1 \\
&u_i^1(\rho_i) = S_0(\rho_i) \left[ C_r(\rho_i) r - C_y(\rho_i) v_i^1 \right] \\
&\\
&Exp.2: \\
&r_i^2 = T_0(\rho_i) \\
&y_i^2 = T_0(\rho_i)(r - r_i^1) + S_0(\rho_i) v_i^2) \\
&u_i^2(\rho_i) = S_0(\rho_i) \left[ C_r(\rho_i)(r - y^1(\rho_i)) - C_y(\rho_i) v_i^2 \right] \\
&\\
&Exp.3 \\
&r_i^3 = r \\
&y_i^3(\rho_i) = T_0(\rho_i) r + S_0(\rho_i) v_i^3 \\
&u_i^3(\rho_i) = S_0(\rho_i) \left[ C_r(\rho_i) r - C_y(\rho_i) v_i^3 \right]
\end{aligned} \tag{13}$$

The IFT algorithm, for 2DoF case, consists in a series of tuning cycles. At each iteration $i$, the vector of controller

parameters $\rho_i$ is updated by taking a step, of size $\gamma_i$ in the, possibly modified, direction of an unbiased estimation of the cost gradient. The determination of the later is accomplished by means of (11)-(12) after performing three experiments in closed loop. The first and the third experiments consist simply in gathering of the N sample vector for the output and plant input, $\{y^1, u^1\}$ and $\{y^3, u^3\}$ respectively (13Exp.1 and Exp. 3). In the second experiment 13Exp.2), the only one needing a modification of the nominal loop working conditions, the reference is conformed as $r - y^1$ to obtain the data $\{y^2, u^2\}$. It is to be observed that the estimation of the gradients requires the processing of the experimental data by means of filters that depend only of the known controller structure and the parameters of the current iteration.

$$est\left[\frac{\partial J}{\partial \rho}\right] = \frac{1}{N}\left[\sum_{t=1}^{N} \tilde{y}_t(\rho) est\left[\frac{\partial \tilde{y}_t}{\partial \rho}(\rho)\right] + \lambda \sum_{t=1}^{N} u_t(\rho) est\left[\frac{\partial u_t}{\partial \rho}(\rho)\right]\right] \tag{14}$$

The difference between the estimation (11)-(12) and the true gradients (9)-(10) can be shown, after some manipulations, to depend of the realization of the random process disturbance during the second and third experiments, namely $v_t^2$ and $v_t^3$. It should be stressed that the use of a third experiment in gradient estimation is unavoidable to guarantee the unbiasedness of the estimation by assuring the two factors, $\tilde{y}_t(\rho)$ and $\partial \tilde{y}_t / \partial \rho(\rho)$, to be contaminated by different uncorrelated random variables, $v_t^1$ and $v_t^3$. The IFT procedure gets simplified for 1DoF loops, requiring, in that case, only the first two experiments.

Many other details of the IFT method deserve careful consideration. In Hjalmarsson et al. (1998) is offered, for example, the proof of the convergence of the algorithm to a local minimum of the cost and a discussion of its applicability to the non-linear case, among others important issues. An approximation of the Hessian $R$ according to (15) has been also reported to improve the convergence speed.

$$R = \frac{1}{N}\left[\sum_{t=1}^{N} est\left[\frac{\partial \tilde{y}_t}{\partial \rho}(\rho)\right] est\left[\frac{\partial \tilde{y}_t}{\partial \rho}(\rho)\right]^T + \lambda \sum_{t=1}^{N} est\left[\frac{\partial u_t}{\partial \rho}(\rho)\right] est\left[\frac{\partial u_t}{\partial \rho}(\rho)\right]^T\right] \tag{15}$$

There are many IFT practical experiences reported in the already mentioned papers, which includes the its succesful use at important chemical installations. In addition, the following references offer a incomplete account of the range of applications attempted with with the discussed tuning method: Hamamoto et al. (2003), Mazaeda and Prada (2000), Kissling et al. (2009), Graham et al. (2007), Tay et al. (2006) and McDaid et al. (2010).

The original IFT method has benefited from many relevant theoretical contributions. For example, its application to IMC type controllers and smith predictors (De Bruyne, 2003); its combination with the celebrated relay autotuning method (Ho et al., 2003) and its use to tune a decoupling MIMO 2x2 controller (Gunnarsson et al., 2003). In Hjalmarsson (2005) the IFT algorithm is viewed in the broader context of the relation between identification and control; while in Sala (2007) further insight is given on the subject by putting identification for control, Iterative Feedback Tuning and Virtual Reference Feedback Tuning (VRFT) in a common framework.

## 3. THE SOFTWARE TOOL

The IFT algorithm is particularly suitable for industrial application. It fulfills the natural practical requirements of being a model-free closed-loop procedure which does not demand a very severe modification of the nominal working conditions. In fact, the only 'real' experiment asking for the injection of an artificial reference is the second one. Even more, in the case of tuning for perturbation rejection, very important for the regulatory type of control typically deployed in process industries, the reference at the first experiment is zero, and the adjustment process can be entirely driven by the random perturbations in the loop.

The tuning program here described, IFTtune, has been developed to ease the application of the IFT method in the process industries. Hence, the software implements those theoretical results that have an easy to grasp appeal. Additionally, it has been designed with the intention of being open and configurable, offering at the same time a minimun set of practical safeguards constraining the freedom of the underlying algorithm during the tuning procedure.

The main interface of the program, figure 2, readily offers to the user a general view of was going on in the configured loop. There are windows showing the time evolution of the reference and process variable, on one hand, and of the output of the controller, on the other. The data related to the evolution of the tuning algortihm is shown in a third window, as for example, the functioning state (normal functioning, IFT experiment one, two, or three), the current values of the controller parameters, the optimization step ($\gamma$), and the current and previous values of the performance index ($J$). This last window also provides a convenient entry point for accessing the configuration dialogs and to manually modify such elements as the controllers parameters or the loop reference.

Immediately upon connection, the software starts to monitor the variables and parameters of the target loop with the configured sample frequency. In this so called normal mode, the operator has the oportunity of modifying the mentioned values, as a means of visually assessing the achieved performance. The process of auto-tuning is triggered at operator's will, by clicking the corresponding interface button. The sequence of IFT experiments, then, unfolds, as previously configured for that particular loop. When the current cost is worst than the previous one, the loop returns to normal state, leaving in the controller the last valid values. The operator can abort the tuning process at any moment, and has the choice of keeping the current, already altered parameters of the controller, or of rolling back to the original ones.

Each individual loop must have its own IFT algorithm parameters. The configure button launches the dialog box shown in figure 3 where important elements already discussed, such as the number of samples to consider in

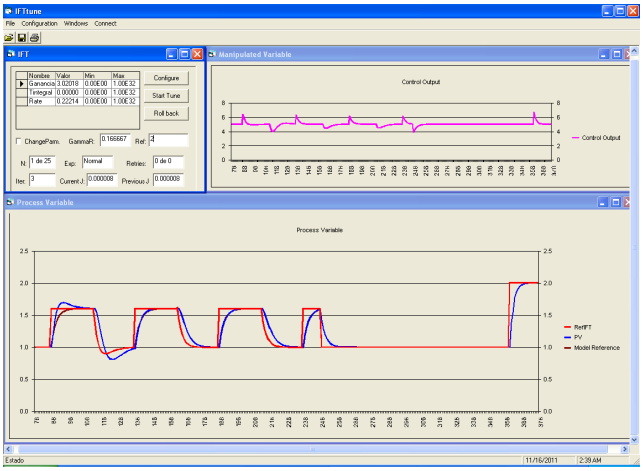

Fig. 2. IFTtune user's interface

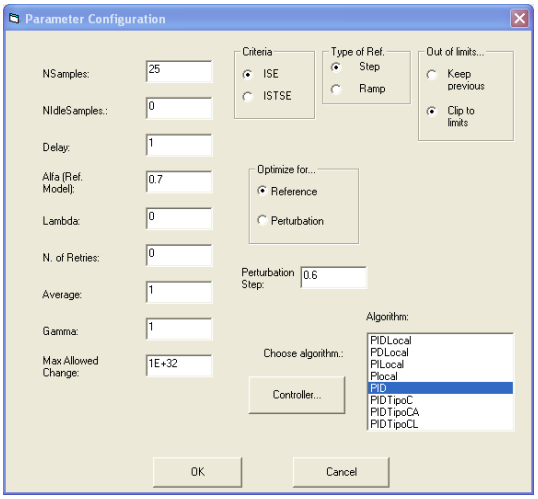


Fig. 3. Algorithm configuration dialog

the experiments ($N$), the weight given to the control signal effort in the cost ($\lambda$), the initial value of the optimization step ($\gamma$) are set. It should be mentioned that the $\gamma$ used at each step is derived from this value by dividing it by the iteration number, as is advised by the above mentioned convergence result that demands that $\sum_i^\infty \gamma_i < \infty$.

The configuration dialog also exhibits other configurable parameters, which have been included to improved robustness and to put in place some basic safeguards. There is, for example, the posibility of retrying the execution of the IFT cycle $i$ a pre-specified number to times (**N. of Retries**), with the parameters of the cycle $i-1$ in spite of obtaining $J_i > J_{i-i}$, as a means of giving some extra chances to the algorithm in the, tipically, very noisy industrial enviroment. The same rationale justifies, in some cases, the convenience of several IFT experiments at step $i$, to calculate, and then take, a more dependable step in the direction of the resulting averaged gradient (**Average**). There is also the posibility of establishing a range of permissible values for each parameters (see figure 4) and a general maximum rate of change for the parameters at each iteration (**Max. Allowed Change**). The violation of any of the hard restrictions can be managed in different ways either by clipping the offending parameter to the correspondig limit or by keeping the last value, as configured in the **Out of limits...** box.

The program uses the modified tuning criterion (16), proposed by Lequin et al. (2003), which differs from (6) in that the original frequency dependent filters $(L_y, L_u)$ are replaced by time dependent weighs $(W_y, W_u)$. These latter parameters allow, for example, to mask out from the optimization cost a user defined first stretch of the transient response. This would prevent the controller parameter optimization search from trying to comply with the arbitrary shape of the desired transient response, to deal instead with the really important task of going from a reference to the next.

$$J(\rho) = \frac{1}{2N}\mathrm{E}\left[\sum_{t=1}^{N}\left(W_y(t)\tilde{y}_t(\rho)\right)^2 + \lambda\sum_{t=1}^{N}\left(W_u(t)u_t\right)^2\right] \quad (16)$$

The mask is also useful for taking into account some possible inherent deadtime of the plant. Note that if the time weighs are constantly increased, proportionally to the sample count, the standard ISTSE (**I**ntegral **S**quare **T**ime per **S**quare **E**rror) criterion can be easily approximated. The above considerations are implemented by appropiately setting the **Delay** numerical parameter and the **Criteria** box in figure 3.

As already explained, the original IFT derivation assumes the tuning process to be mainly driven by the random perturbations in the loop. But it is also reasonable to assume, that the convergence of the algorithm would be impaired if the signal to noise ratio in the system is not high enough as to provide adequate excitation of the process and a suitably informative set of data. The above considerations were put forward in Huusom et al. (2009), where the idea of injecting an appropriate probe signal either at the reference ($r_p$) or at the plant input ($u_p$), as shown in figure 5. The conditions to be met by the injected probes, in order to still obtain an unbiased estimate of the gradients, were found to be the following: the $r_p$ signal should be present at the first and third experiment and they should be equal ($r_p^1 = r_p^3$) while it should be absent in the second experiment ($r_p^2 = 0$). The plant entrance probe, on the other hand, should be injected only in the first experiment ($u_p^2 = u_p^3 = 0$).

This basic idea has been implemented in the tool. There is the possibility of specifying the size for a step in the reference (**Perturbation step**) or for an artificial probe impulse signal at another configured input into the plant, typically the controller output (**Optimize for ...** box). In favouring simplicity, the form of the perturbation probe has been fixed as mentioned. The idea of the original paper in the sense of optimally designing the spectrum of these signals would require the use of a simplified model of the plant thus reducing the original appeal of the IFT method. For the reference probe, there is the possibility of choosing a ramp instead of a step (**Type of Ref.** box), an option put forward very early (Hjalmarsson et al., 1998) to avoid the cancellation of a possibly existing fixed integral action by a zero moving in the adjustable part of the controller. The parameter specified by the field **NIdleSamples** has beed added to represent the number of samples to be left between one experiment and the next to give enough time so that the perturbed loop signals return to a similar stationary state.

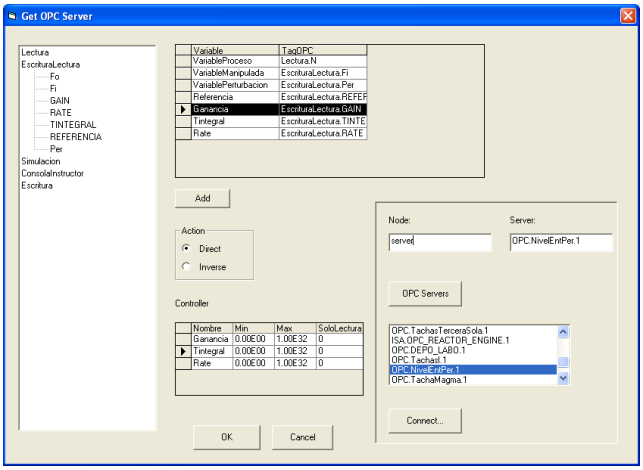


Fig. 4. OPC variables configuration dialog

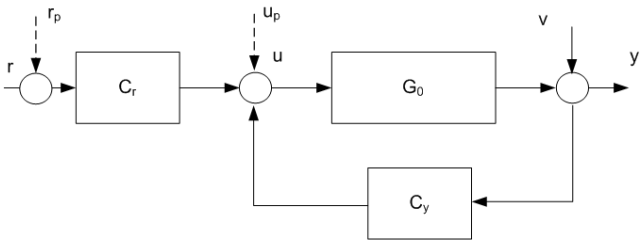


Fig. 5. IFT tuning with excitation probes

The configuration parameter **Alfa (Ref. Model)** allows to choose the $\alpha$ of the response of a first the order model (17) whose output is to be taken as the desired closed loop trajectory $y_d$.

$$y_d = \frac{(1-\alpha)z^{-Delay}}{1-\alpha z^{-1}} r \tag{17}$$

Loop variables and controller parameter are accessed via OPC (OLE for Process Control) (Iwanitz and Lange, 2002) a mature communication standard which is widely adopted in process industries. The figure 4 shows the way IFTtune matches the internal reference of the involved variables with their corresponding OPC assigned identification tags. The OPC server, the source of plant data complying with the protocol, needs to be specified and should be accesible either locally or in a reachable network node. Many OPC servers have browsing capabilities allowing their clients to navigate the tags in the namespace which is typically organized as a tree data arrangement.

The IFT method can be applied to reduced order controllers of arbitrary structure. The IFTtune program allows the definiton of different controllers which, fundamentally, means the ability of defining their associated filters (11-12). The required definiton is implemented appealing to a pair of related database tables as shown in figure 6. The **controller** table has fields for specifying the name of the particular controller and for deciding if it is 2DoF or not. The normal software use implies the existence of a field comissioned controller which is accessed via OPC; but it also gives the posibility of locally defining the loop controller. In the latter case, the **local** field should be set and then the numerator and denominator of the filters defining the controller should be found in

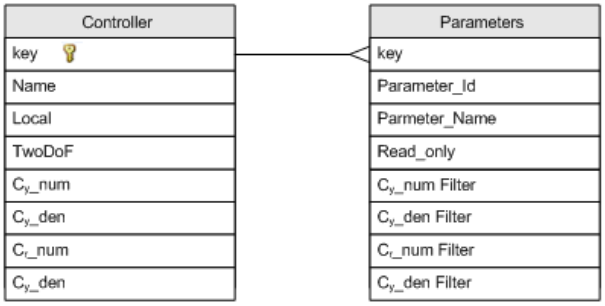


Fig. 6. Controller - parameters definition database tables

the fields (**Cynum**, **Cyden**) and (**Crnum**, **Crden**) as mathematical expressions involving the parameters. This expressions follow a format that can be parsed in real time by the program. For each defined controller, there is a set of records of the **parameters** table, one for each parameter, defining the numerator and denominator of the needed filters. This table has additional fields for identifying the parameters by **name** and for declaring some parameters to be **read-only** and thus not modifiable by the tuning procedure. The controller-parameters database ships with the definiton of several controllers and their associated filters, including several PID versions but it can be easily extended with others as needed. The software distribution also includes several continuous time simulated examples, which are installed as local OPC servers, for off-line training purposes.

As observed from the previous description, there are many configuration data pertaining to each loop. But in any case, many of them remain unchanged after been initially set. The tool provides the mechanisms for saving the configuration of each in a dedicated database table.

In figure 2 the tuning procedure for a textbook parallel PID controller in a simulated level control problem is shown. The local optimum is obtaining in just three iterations. The tuning procedure is driven by a reference step and the desired output has been obtained with an $\alpha = 0.7$. Each iteration consists of two experiment as corresponds to a 1DoF controller.

## 4. CONCLUSIONS

The tuning tool here presented could be improved in several ways. It would be useful, for example, that the process of auto-tuning should be triggered automatically as the result of the loop performance diagnosis (Ghraizi et al., 2007).

On the other hand, the toolbox of possible algorithms to choose from could be augmented, favouring those model-free procedures which are easily applicable in the industry. There would much to be gained by counting with an implementation of the relay method, like for example, the recently improved version discussed in Hang et al. (2002), or by counting with other interesting alternatives like the ones represented by, the already mentioned, correlation based tuning and VRFT methods.

ACKNOWLEDGEMENTS

The research leading to these results have received funding from the Ministry of Science and Innovation of Spain under the project MICINN DPI2009-12806.

REFERENCES

Aström, K. (1988). *Automatic tuning of PID controllers*. ISA.

Aström, K. and Hägglund, T. (1984). Automatic tuning of simple regulators on phase and amplitude margins. *Automatica*, 20, 645–651.

Campi, M., Lecchini, A., and Savaresi, S. (2002). Virtual reference feedback tuning: a direct method for the design of feedback controllers. *Automatica*, 38, 1337–1346.

De Bruyne, F. (2003). Iterative feedback tuning for internal model controllers. *Control Engineering Practice*, 11, 1043–1048.

Gevers, M. (2002). A decade of progress in iterative process control design: from theory to practice. *Journal of Process Control*, 12, 519 – 531.

Ghraizi, R., Prada, C., Martnez, E., Alonso, F., and Gonzalez, J. (2007). Performance monitoring of industrial controllers based on the predictibility of controller behaviour. *Computers and Chemical Engineering*, 31, 477 – 486.

Graham, A., Young, A., and Sie, S. (2007). Rapid tuning of controllers by ift for profile cutting machines. *Mechatronics*, 17, 121–128.

Gunnarsson, S., Collignon, V., and Rousseaux, O. (2003). Tuning of a decoupling controller for a 2x2 system using iterative feedback tuning. *Control Engineering Practice*, 11, 1035–1041.

Hamamoto, K., Fukuda, T., and Sugie, T. (2003). Iterative feedback tunning of controllers for a two-mass-spring system with friction. *Control Engineering Practice*, 11, 1061 – 1068.

Hang, C., Aström, K., and Wang, Q. (2002). Relay feedback auto-tuning of process controllers: a tutorial review. *Journal of Process Control*, 12, 143–162.

Hjalmarsson, H. (2005). From experiment design to closed loop control. *Automatica*, 41, 393 – 438.

Hjalmarsson, H., Gevers, M., Gunnarsson, S., and Lequin, O. (1998). Iterative feedback tuning: theory and applications. *Control Systems, IEEE*, 18, 26–41.

Ho, W., Hong, Y., Hansson, A., Hjalmarsson, H., and Deng, J. (2003). Relay auto-tuning of PID controllers using iterative feedback tuning. *Automatica*, 39, 149–157.

Huusom, J., Poulsen, N., and Jorgensen, S. (2009). Improving convergence of Iterative Feedback Tuning. *Journal of Process Control*, 19, 570–578.

Iwanitz, F. and Lange, J. (2002). *OPC : fundamentals, implementation and application*. Hüthig.

Karimi, A., Miskovic, L., and Bonvin, D. (2003). Iterative correlation-based controller tuning with application to a magnetic suspension system. *Control Engineering Practice*, 11, 1069 – 1078.

Kissling, S., Blanc, P., Myszkorowski, P., and Vaclavik, I. (2009). Application of iterative feedback tuning to speed and position control of a servo drive. *Control Engineering Practice*, 17, 834 – 840.

Lequin, O., Gevers, M., Mossberg, M., and Bosmans, E. (2003). Iterative feedback tuning of pid parameters: comparison with classical tuning rules. *Control Engineering Practice*, 11, 1023 – 1033.

Mazaeda, R. and Prada, C. (2000). Iterative feedback tuning of a pid in a pilot plant. In *Digital Control: Past, Present and Future of PID Control*.

McDaid, A., Aw, K., Sie, S., and Haemmerle, E. (2010). Gain scheduled control of ipmc actuactors with 'model free' iterative feedback tuning. *Sensors and Actuactors A: Physical*, 164, 137–147.

O'Dwyer, A. (2006). *Handbook of PI and PID Controller Tuning Rules*. Imperial College Press.

Sala, A. (2007). Integrating virtual reference feedback tuning into a unified closed-loop identification framework. *Automatica*, 43, 178–173.

Tay, A., Ho, W., J., D., and Lok, B. (2006). Control of photoresist film thickness: Iterative feedback tuning approach. *Computers and Chemical Engineering*, 30, 572–579.