MULTIOBJECTIVE TUNING OF ROBUST GPC CONTROLLERS USING EVOLUTIONARY ALGORITHMS

J. M. Herrero, X. Blasco, M. Martínez and J. Sanchis
Instituto Universitario de Automática e Informática Industrial, Universidad Politécnica de Valencia
Camino de Vera s/n, 46022 Valencia, Spain

Keywords: Multiobjective optimization, Evolutionary algorithms, Predictive control.

Abstract: In this article a procedure to tune robust Generalized Predictive Controllers (GPC) is presented. To tune the controller parameters a multiobjective optimization problem is formulated so the designer can consider conflicting objectives simultaneously without establishing any prior preference. Moreover model uncertainty, represented by a list of possible models, is considered. The multiobjective problem is solved with a specific Evolutionary Algorithm (ev-MOGA). Finally, an application to a non-linear thermal process is presented to illustrate the technique.

1 INTRODUCTION

Generalized predictive control (GPC) (Clarke et al., 1987a) (Clarke et al., 1987b) has been shown to be an effective way of controlling single-input single-output processes. The strategy proposed by GPC is simple to understand and makes good practical sense: predict the behaviour of the output as a function of future control increments and minimize over these increments a cost index. This cost includes the errors between predicted and desired outputs and the control effort. Despite its advantages, tuning GPC methods are based on a linear models, which are usually adjusted around an operating point. When the process operates outside the validity zone of the model (where differences between model and process behaviour increase) poor control performance is obtained since in that case the tuning is suboptimal even close-loop stability could take place.

To avoid that, robust GPC tuning approach is assumed. In this case model uncertainties are taking into account to cover non-modelled dynamics (such as non linearities, high frequency dynamics, and so on) and measurement noise (Reinelt et al., 2002). The simpler the model is the bigger uncertainties are, producing an excess of conservativeness in the tuning result, which give as a result a loss of performance in the close-loop control. Therefore the goal is to achieve robust tunings with good performance at the same time, for instance, minimizing error or control effort. Objectives that are usually in contraposition.

The GPC tuning methodology that is presented tries to achieve that goal by:

- Using non-linear parametric models with uncertainty. The uncertainty is consider by means of a set of models, the Feasible Parameter Set (FPS^*). Although the real process is not known, assume that it lies within the FPS^* (Walter and Piet-Lahanier, 1990).
- Proposing a Multiobjective optimization (MO) GPC tuning approach.

Optimal tuning considers not only a nominal model but the FPS adjusting the controller parameters for the worst case (the most unfavorable model). Moreover, because the tuning method has to consider conflicting objectives, an optimization multiobjective problem is stated where each objective minimizes the maximum cost function for all the models in the uncertainty description.

Multiobjective optimization (MO) techniques present advantages as compared with single objective optimization techniques due to the possibility of giving a solution with different trade-offs among different individual objectives so that the Decision Maker (DM) can select an appropriate final solution.

The presence of multi-modal MO functions and non-convex constrined spaces needs optimizer with good performance. A good choice are stochastic optimizers such as the Evoluationary Algorithms (EAs) (Coello et al., 2002) that can work well with multimodal and non-convex problems, in particular, the al-

M. Herrero J., Blasco X., Martínez M. and Sanchis J. (2009).
MULTIOBJECTIVE TUNING OF ROBUST GPC CONTROLLERS USING EVOLUTIONARY ALGORITHMS.
In *Proceedings of the International Joint Conference on Computational Intelligence*, pages 263-268
DOI: 10.5220/0002269102630268

gorithms used in this work will be the ev-MOGA one (Herrero et al., 2007c; Herrero et al., 2007b).

This paper is organized as follows. Section 2 presents GPC formulation, section 3 introduces tuning procedure proposed in this article, section 4 describes briefly used. Section 5 illustrates the GPC tuning procedure with the example of a thermal process. Finally, some concluding remarks are reported in section 6.

2 GPC FORMULATION

The GPC formulation with quadratic cost index has been extensively developed in (Clarke et al., 1987a), (Clarke et al., 1987b). Such formulation uses the following CARIMA stochastic model:

$$y(t) = \frac{B(z^{-1})}{A(z^{-1})}u(t-1) + \frac{T(z^{-1})}{\Delta A(z^{-1})}d(t) \qquad (1)$$

where: $u(t)$ and $y(t)$ are the process input and output respectively, $d(t)$ the disturbance (white noise), $T(z^{-1})$ is a polynomial used to filter disturbance and, $B(z^{-1})$ and $A(z^{-1})$ are the polynomial transfer function of the discrete model. A GPC controller is obtained through the optimization of the following cost index and applying Receding Horizon:

$$J(\Delta u) = E[\sum_{i=N_1}^{N_2} \alpha[y(t+i) - r(t)]^2 + \sum_{j=1}^{N_u} \lambda[\Delta u(t+j-1)]^2] \qquad (2)$$

where $N = N_2 - N_1 + 1$ is the prediction horizon, N_u is the control horizon, α is the prediction error weighting factor, λ is the control weighting factor, $r(t)$ is the setpoint, and $[\ \Delta u(t)\ \ \Delta u(t+1)\ \ \cdots\ \ \Delta u(t+N_u-1)\ ]^T$ are the control actions.

Optimizing index (2) and applying Receding Horizon (so that, using only $\Delta u(t)$) the following GPC expression is obtained:

$$u(z) = \frac{T(z^{-1})\left(H_0 r(z) - \frac{S(z^{-1})}{T(z^{-1})}y(z)\right)}{(T(z^{-1}) + R(z^{-1})z^{-1})\Delta}$$

Figure 1 represents implementation of this controller with a block diagram.

3 MULTIOBJECTIVE TUNING OF ROBUST GPC

Let's assume the following model structure:

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), u(t), \theta), \quad \hat{y}(t,\theta) = g(\mathbf{x}(t), u(t), \theta) \quad (3)$$

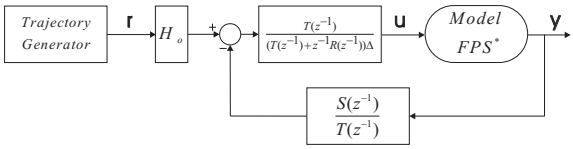

Figure 1: Control structure for GPC. r represents the set point for the output y.

where: $f(.), g(.)$ are the non-linear functions of the model; $\theta \in R^L$ is the vector of unknown model parameters; $\mathbf{x}(t) \in R^n$ is the vector of model states; $u(t)$ is the input process and $\hat{y}(t,\theta)$ the output.

Asume θ_n are the parameters of the nominal model which belong to FPS

$$FPS := \{\theta_1, \ldots, \theta_p\} \qquad (4)$$

that represents the model parameters uncertainty.

The possible controller parameters to tune are $\mathbf{k} = \{N_1, N_2, N_u, \alpha, \lambda, T(z)\}$. To obtain the controller the following MO problem can be formulated:

$$\min_{\mathbf{k}\in D} \mathbf{J}(\mathbf{k}) = \min_{\mathbf{k}\in D}[J_1(\mathbf{k}), J_2(\mathbf{k}), \ldots, J_s(\mathbf{k})] \qquad (5)$$

where $J_i(\mathbf{k})$, $i \in B := [1 \ldots s]$ are the objectives to minimize and $\mathbf{k}$ is a solution inside the solution space D.

Since each objective to minimize has to take into account the model uncertainty FPS^* then

$$J_i(k) = \max_{\theta\in FPS^*} \phi_i \qquad (6)$$

where the cost function ϕ_i is the real objective to minimize for the worst model case belonging to FPS^*. Some typical criteria are: the norm of the control action: $\phi_i = ||u(t)||$, the norm of the rate of change of control action: $\phi_i = ||\Delta u(t)||$, the norm of the error: $\phi_i = ||r(t) - y(t)||$ or the norm of the error weighted with time: $\phi_i = ||t(r(t) - y(t))||$.

Anyway, to solve the MO problem the Pareto optimal set $\mathbf{K}_P$ (solutions where no-one dominates others) must be found. $\mathbf{K}_P$ is unique and normally includes infinite solutions. Hence a set $\mathbf{K}_P^*$ (which is not unique), with a finite number of elements from $\mathbf{K}_P$, should be obtained (see (Coello et al., 2002) for details of MO problems). To obtain $\mathbf{K}_P^*$ a MOEA known as the ev-MOGA algorithm (Herrero et al., 2007c; Herrero et al., 2007b) will be used.

Finally a unique solution $\mathbf{k}^*$ of the Pareto optimal set $\mathbf{K}_P^*$ has to be selected. The selection procedure is based on designer preferences and can differs depending on design needs. Since all Pareto optimal points are non-dominated any selection made will be always optimal.

4 EV-MOGA ALGORITHM

ev-MOGA (previously called ε-MOGA, (Herrero et al., 2007b; Herrero et al., 2007c)) is an elitist multiobjective evolutionary algorithm based on the concept of ε-dominance (Laumanns et al., 2002). A completed and detailed version of ev-MOGA algorithm is developed in (Herrero, 2006) where the performance of the algorithm is tested by facing up to classical benchmarks for MO. It obtains an ε-Pareto set, $\mathbf{K}_P^*$, that converges towards the Pareto optimal set $\mathbf{K}_P$ in a distributed manner around Pareto front $\mathbf{J}(\mathbf{K}_P)$, with limited memory resources. Next a brief description of the ev-MOGA algorithm is presented.

ev-MOGA adjusts the limits of the Pareto front $\mathbf{J}(\mathbf{K}_P^*)$ dynamically and prevents the solutions belonging to the ends of the front from being lost. For this reason, the objective space is split up into a fixed number of boxes n_box_i, for each dimension i, so that this grid preserves the diversity of $\mathbf{J}(\mathbf{K}_P^*)$ since one box can be occupied by only one solution. This fact prevents that the algorithm converges towards just one point or area inside the objective space (see Fig. 2).

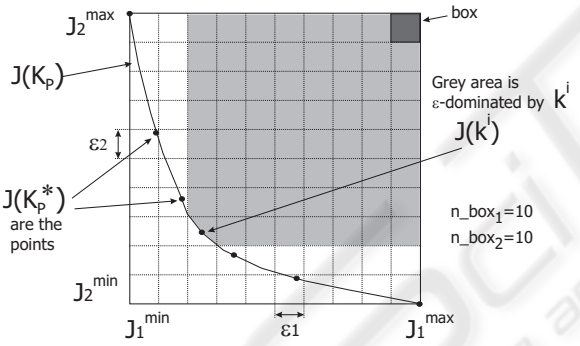

Figure 2: The concept of ε-dominance. ε-Pareto Front $\mathbf{J}(\mathbf{K}_P^*)$ in a two-dimensional problem. J_1^{min}, J_2^{min}, J_1^{max}, J_2^{max}, Pareto front limits; ε_1, ε_2 box widths; and n_box_1, n_box_2, number of boxes for each dimension.

The algorithm is composed of three populations: The main population $P(t)$ which explores the searching space D during the algorithm iterations (t). Its Population size is $Nind_P$; the archive $A(t)$ which stores the solution $\mathbf{K}_P^*$. Its size $Nind_A$ can vary but it will never be higher than

$$Nind\_max\_A = \frac{\prod_{i=1}^{s} n\_box_i + 1}{n\_box_{max} + 1} \qquad (7)$$

where $n\_box_{max} = \max([n\_box_1, \ldots, n\_box_s])$ and the auxiliary population $G(t)$. Its size is $Nind_G$, which must be an even number.

The pseudocode of the ev-MOEA algorithm is given by

```
1.  t:=0; A(t):=∅;
P(t):=ini_random(D)
2.  eval(P(t))
3.  A(t):=store_ini(P(t),A(t))
4.  while t<t_max {
5.       G(t):=create(P(t),A(t))
6.       eval(G(t))
7.       A(t+1):=store(G(t),A(t))
8.      P(t+1):=update(G(t),P(t))
9.       t:=t+1 }
```

The main steps of the algorithm are detailed as follows:

Steps 2 and 6. Function **eval** calculates function value (Equation (5)) for each individual in $P(t)$ (step 2) and $G(t)$ (step 6).

Step 3. Function **store**$_{ini}$ checks individuals of $P(t)$ that might be included in the archive $A(t)$ as follows:

1. Non-dominated $P(t)$ individuals are detected, $\mathbf{K}_{ND}$.
2. Pareto front limits J_i^{max} and J_i^{min} are calculated from $\mathbf{J}(k), \forall \mathbf{k} \in \mathbf{K}_{ND}$.
3. Individuals in $\mathbf{K}_{ND}$ are analyzed, one by one, and those that are not ε-dominated by individuals in $A(t)$, will be included in $A(t)$.

Step 5. Function **create** creates individual of $G(t)$ by using linear recombination technique and random mutation with Gaussian distribution.

Step 7. Function **store** checks, which individuals in $G(t)$ must be included in $A(t)$ on the basis of their location in the objective space. Only individuals which are not ε-dominated by any individual from $A(t)$ will be included (if its box is occupied by an individual not ε-dominated too, then the individual lying farthest away from the centre box will be eliminated). Individuals from $A(t)$ which are ε-dominated by individual of $G(t)$ will be eliminated. Also this function updates the limits J_i^{max}, J_i^{min} of the Pareto front if it is necessary.

Step 8. Function **update** updates $P(t)$ with individuals from $G(t)$. Every individual $\mathbf{k}^G$ from $G(t)$ is compared with an individual $\mathbf{k}^P$ randomly selected from $P(t)$. If $\mathbf{k}^G$ dominates $\mathbf{k}^P$ then $\mathbf{k}^G$ replaces $\mathbf{k}^P$. $\mathbf{k}^P$ will not be included in $P(t)$ if there is no individual in $P(t)$ dominated by $\mathbf{k}^G$.

Finally, individuals from $A(t)$ compound the MO problem solution $\mathbf{K}_P^*$.

5 ROBUST GPC TUNING FOR A THERMAL PROCESS

A scale furnace with a resistance placed inside is considered. A fan continuously introduces air from outside (air circulation) while energy is supplied by an actuator controlled by voltage. Taking into account heat transfer phenomena (conduction, convection and radiation) the dynamics of the resistance temperature can be modelled by

$$\dot{x}(t) = \frac{\left(\theta_1 u(t)^2 - \theta_2 \left(x(t) - T_a(t)\right) - \frac{\theta_3 (273+x(t))^4}{100^4}\right)}{1000}, \tag{8}$$

$$\hat{y}(t) = x(t), \tag{9}$$

where: $\dot{x}(t)$ is the model state; $u(t)$ is the input voltage with rank 0 - 100 (%); $\hat{y}(t)$ is the resistance temperature (^oC) (model output); $T_a(t)$ is the air temperature (^oC) and $\theta = [\theta_1, \theta_2, \theta_3]^T$ are the model parameters.

To obtain the (FPS^*), which characterize the model uncertainty, the robust identification method presented in (Herrero et al., 2007a) was applied. The FPS^* is discrete characterization of the parameter set which keeps the model predictions error bounded for certain norms and bounds.

In this example ∞-norm and absolute norm are simultaneously used to determine the FPS^*. Bounds are selected in order to hold the FPS^* models predictions errors lower than 2^oC and their average values lower than 0.8^oC.

The resulting FPS^* contains 304 models (for more details see (Herrero et al., 2007c)).

The nominal model $\theta_n = [0.0776, 4.52, 0.176] \in FPS^*$ is linearized in the $[y,u] = [56.1, 50]$ point and converted to discrete time with $T_s = 10$ sample time obtaining the following model $B(z^{-1}) = 0.0758z^{-1}$ and $A(z^{-1}) = 1 - 0.9533z^{-1}$. The following GPC parameters are fixed $N_1 = N_u = \alpha = 1$, $T(z^{-1}) = A(z^{-1})$ whilst N_2 and λ will be tuned. Therefore the searching space is defined by $N_2 \in [5, 6, \ldots 100]$ and $\lambda \in [0.1 \ldots 100]$. The functions selected are the followings:

$$\phi_1 = \frac{||r(t) - y(t)||_1}{N}, \phi_2 = ||\Delta u(t)||_1 .$$

with $r(t) = [r(0), r(1 \cdot Ts) \ldots r(N \cdot T_s)]$ and $N = 250$ is the number of samples.

Then the MO problem to solve is the following:

$$\min_{k \in D}[J_1(k), J_2(k)] = \min_{k \in D}[\max_{\theta \in FPS^*} \phi_1, \max_{\theta \in FPS^*} \phi_2]$$

To solve this MO problem the algorithm ev-MOGA is used. The parameters of the ev-MOGA algorithm were set to: $Nind_G = 4$; $Nind_P = 100$; $t_{max} = 1000$ (resulting in 4100 evaluations of $J_1(k)$ and $J_2(k)$) and $n\_box_1 = n\_box_2 = 200$. The algorithm was run 10 times.

Fig. 3 shows the best Pareto front and set obtained. Notice that the Pareto front is disjoint, the same as the Pareto optimal set. The better characterization of the Pareto front is needed the larger n_box_i has to be used. ev-MOGA algorithm captures the extremes of the Pareto front, and thus $\mathbf{K}_P^*$ will contain the optimal solutions k^{J_i} of each J_i considered on an individual basis.

Analyzing the Pareto front (see Fig. 3), the solutions corresponding to higher values of λ (bottom right area of the Pareto Front) produce bigger control error as it is expected. Otherwise, the solutions corresponding to lower values of λ and bigger values of N_2 (top left area) produce lower control error in exchange for bigger control effort.

Therefore, taking into account the Pareto front and set obtained, the following compromise solution $\mathbf{k}^*$ has been selected $\mathbf{k}^* = [12, 6.552] \Rightarrow \mathbf{J}(\mathbf{k}^*) = [3.0885, 123.1767]$.

Fig. 4 shows the envelop generated by the compromise controller k^* for the output $y(t)$ and input $u(t)$ when all the models of the FPS^* are considered.

6 CONCLUSIONS

A methodology, based on Evolutionary Algorithms, has been developed to tuning robust GPCs from an MO point of view. The methodology presents the following features:

- Assuming parametric uncertainty, all kind of processes can be considered.
- Since a non-linear models set have been considered, low uncertainties are produced by the robust identification process (a difference that a liner model with interval parametric uncertainty were considered) and therefore less conservativeness is produced.
- Any kind of design objectives can be used simultaneously to tune the GPC controller resulting in a MO Problem. Thanks to the ev-MOGA algorithm would be possible to characterize all kind of Pareto fronts in a well-distributed manner with bounded memory resources.

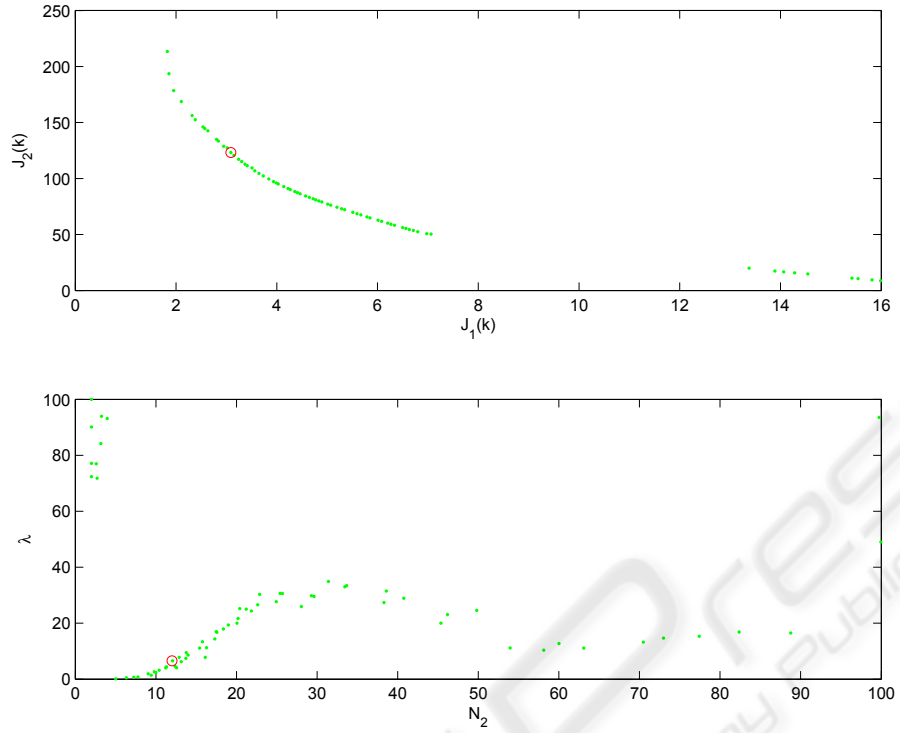

Figure 3: Top: the ε-Pareto front $\mathbf{J}(\mathbf{K}_P^*)$. Bottom: the Pareto optimal set $\mathbf{K}_P^*$. (*) Compromise solution obtained, $\mathbf{k}^*$ and $\mathbf{J}(\mathbf{k}^*)$.

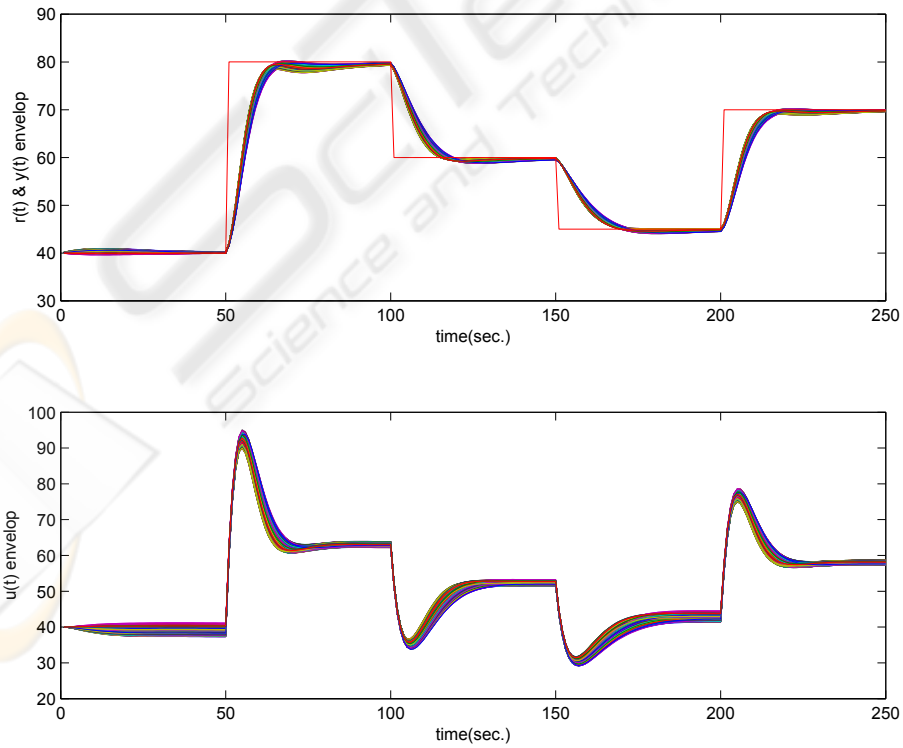

Figure 4: Top: Set point trajectory generated $r(t)$ and the envelop of the outputs $y(t)$ when the controller k^* is applied to FPS^* models. Bottom: the envelop of the control action produces the envelop of the outputs.

ACKNOWLEDGEMENTS

Partially funded by GVPRE/2008/326 and DPI2008-02133/DPI.

REFERENCES

Clarke, D., Mohtadi, C., and Tuffs, P. (1987)a). Generalized predictive control. Part I. *Automatica*, 23(2):137–148.

Clarke, D., Mohtadi, C., and Tuffs, P. (1987b). Generalized predictive control. Part II. extensions and. *Automatica*, 23 (2):149–160.

Coello, C., Veldhuizen, D., and Lamont, G. (2002). *Evolutionary algorithms for solving multi-objective problems*. Kluwer Academic Publischers.

Herrero, J. (2006). *Identificación robusta de sistemas no lineales mediante algoritmos evolutivos [Robust identification of non-linear systems using evolutionary algorithms.]*. PhD thesis, Universidad Politécnica de Valencia, Valencia, Spain.

Herrero, J., Blasco, X., Martínez, M., and Salcedo, J. (2007a). Non-linear robust identification: Application to a thermal. *Lecture Notes in Computer in Computer Science*, 4527:457–466.

Herrero, J., Martínez, M., Ramos, C., and Sanchis, J. (2007b). Non-linear robust identification of a greenhouse model using multi-objective evolutionary algorithms. *Biosystems agriculture*, 16 (5):515–530.

Herrero, J., Martínez, M., Sanchis, J., and Blasco, X. (2007c). Well-distributed pareto front by using the. *Lecture Notes in Computer in Computer Science*, 4507:292–299.

Laumanns, M., Thiele, L., Deb, K., and Zitzler, E. (2002). Combining convergence and diversity in evolutionary multi-objective. *Evolutionary computation*, 10 (3):263–282.

Reinelt, W., Garulli, A., and Ljung, L. (2002). Comparing different approaches to model error modelling in robust identification. *Automatica*, 38 (2):787–803.

Walter, E. and Piet-Lahanier, H. (1990). Estimation of parameter bounds from bounded-error data: A survey. *Mathematics and computers in Simulation*, 32:449–468.