Evaluating Evolutionary Algorithms and Differential Evolution for the Online Optimization of Fermentation Processes

Miguel Rocha[1], José P. Pinto[1], Isabel Rocha[2], and Eugénio C. Ferreira[2]

[1] Departament of Informatics / CCTC - University of Minho
Campus de Gualtar, 4710-057 Braga - Portugal
mrocha@di.uminho.pt, josepedr@gmail.com
[2] IBB - Institute for Biotechnology and Bioengineering
Center of Biological Engineering - University of Minho
Campus de Gualtar, 4710-057 Braga - Portugal
irocha@deb.uminho.pt, ecferreira@deb.uminho.pt

Abstract. Although important contributions have been made in recent years within the field of bioprocess model development and validation, in many cases the utility of even relatively good models for process optimization with current state-of-the-art algorithms (mostly offline approaches) is quite low. The main cause for this is that open-loop fermentations do not compensate for the differences observed between model predictions and real variables, whose consequences can lead to quite undesirable consequences. In this work, the performance of two different algorithms belonging to the main groups of *Evolutionary Algorithms* (EA) and *Differential Evolution* (DE) is compared in the task of online optimisation of fed-batch fermentation processes. The proposed approach enables to obtain results close to the ones predicted initially by the mathematical models of the process, deals well with the noise in state variables and exhibits properties of graceful degradation. When comparing the optimization algorithms, the *DE* seems the best alternative, but its superiority seems to decrease when noisier settings are considered.

Keywords: Fermentation processes, Online optimization, Differential Evolution, Real-valued Evolutionary Algorithms.

1 Introduction

In recent years, many efforts have been devoted to the optimization of processes in bioengineering as a number of valuable products such as recombinant proteins, antibiotics and amino-acids are produced using fermentation techniques. A problem that has received special attention is the dynamic optimization of fed-batch bioreactors. This process has traditionally been conducted on the substrate feed rate as key manipulated variable in operation. The optimization problem is therefore solved before the beginning of the fermentation process (open-loop optimal control) and consists on finding an expression or a sequence of values for

E. Marchiori, J.H. Moore, and J.C. Rajapakse (Eds.): EvoBIO 2007, LNCS 4447, pp. 236–246, 2007.

the feeding rate that maximizes an objective function that represents the process productivity, subject to the constraints represented by a dynamical model.

Several optimization methods have been applied to solve this kind of problem. It has been shown that for relatively simple bioreactor systems, which are expressed in differential equations models, the optimization problem can be solved analytically from the Hamiltonian function by applying the Minimum Principle of Pontryagin [14]. However, in the majority of the cases reported, determination of the optimal feed rate profile has a problem of singular control.

Numerical methods make a distinct approach to dynamic optimization. The gradient algorithms are used to adjust the control trajectories in order to iteratively improve the objective function [3]. In contrast, dynamic programming methods discretize both time and control variables to a predefined number of values. A systematic backward search method in combination with the simulation of the system model equations is used to find the optimal path through the defined grid. However, in order to achieve a global minimum, the computational burden is very high [3].

An alternative comes from the use of algorithms from the *Evolutionary Computation (EC)* field, which have been used in the past to optimize nonlinear problems with a large number of variables. These techniques have been applied with success to the optimization of feeding or temperature trajectories [8][1], and, when compared with traditional methods, usually perform better [12][5].

However, even when the mathematical models used for open-loop optimization are reliable and validated by experimentation, in a real environment several sources of noise can contribute to changes in the observed values of the state variables. These issues are of particular importance when dealing with recombinant high-cell density fermentations, as the process, besides the nonlinearities exhibited, tends to change dramatically upon some events, like induction. Also, it is likely that there exists a time-variance of both yield and kinetic parameters not contemplated in most process models. These scenarios have an important impact on the experimental results that end up being worse than the ones predicted after running the offline optimization.

An alternative to cope with model inaccuracies is the use of online optimization algorithms that periodically generate new solutions as the process is running, making use of the measurement of relevant state variables for update of the internal model. Indeed, unlike the previously stated alternatives where the optimization is conducted prior to the experimental process, in this case, the optimization is performed simultaneously, taking into account values of the state variables measured by sensors within the fermentation process.

In this work, the performance of two different algorithms belonging to the main groups of *Evolutionary Algorithms* (EA) and *Differential Evolution* (DE) is compared in this task of online optimization. These methods were the ones that performed better in offline optimization, in a previous study [6]. Three case studies were taken from literature in order to test the performance of both algorithms. These are used to perform an offline optimization and then a simulation of a real-world fermentation is conducted. The relevant state variables are, in

each case, disturbed by adding a small noise value, at regular periods of time. The behavior of both algorithms is compared, as well as the performance of the initial optimization results given the perturbations considered.

The paper is organized as follows: firstly, the fed-batch fermentation case studies are presented; next, the algorithms of DE and a real-valued EA are described; next, the results of the application of the different algorithms to the case studies are presented; finally, the paper presents a discussion of the results, conclusions and further work.

2 Case Studies: Fed-Batch Fermentation Processes

The case studies used in this work are related to the simulation of fed-batch fermentation processes. In these processes there is an addition of certain nutrients along the process, in order to prevent the accumulation of toxic products, allowing the achievement of higher product concentrations. During this process the system states change considerably, from a low initial to a very high biomass and final product concentrations. This dynamic behavior motivates the development of optimization methods able to find the optimal input feeding trajectories in order to improve the process performance. For the optimization of the process white box mathematical models are developed, based on differential equations that represent the mass balances around the relevant state variables.

2.1 Case Study I

This case study is related to a fed-batch recombinant *Escherichia coli* fermentation process which was previously optimized in [10][11].

The dynamical model can be described by the following equations:

$$\frac{dX}{dt} = (\mu_1 + \mu_2 + \mu_3)X - DX \tag{1}$$

$$\frac{dS}{dt} = (-k_1\mu_1 - k_2\mu_2)X + \frac{F_{in,S}S_{in}}{W} - DS \tag{2}$$

$$\frac{dA}{dt} = (k_3\mu_2 - k_4\mu_3)X - DA \tag{3}$$

$$\frac{dO}{dt} = (-k_5\mu_1 - k_6\mu_2 - k_7\mu_3)X + OTR - DO \tag{4}$$

$$\frac{dC}{dt} = (k_8\mu_1 + k_9\mu_2 + k_{10}\mu_3)X - CTR - DC \tag{5}$$

$$\frac{dW}{dt} \simeq F_{in,S} \tag{6}$$

where X, S, A, O, C represent biomass, glucose, acetate, dissolved oxygen and carbon dioxide concentrations (in g/kg); μ_1 to μ_3 are time variant specific growth rates and k_i are constant yield coefficients. D is the dilution rate, $F_{in,S}$ the substrate feeding rate (in kg/h), W the fermentation weight (in kg), OTR and

CTR are the oxygen and carbon dioxide transfer rates. The kinetic behaviour is given by the specific growth rates μ_1 to μ_3, which are non-linear functions of the state variables. This dependence can be found in [9].

The purpose of the optimization is to determine the feeding rate profile ($F_{in,S}(t)$) that maximizes the productivity of the process, defined as the units of product (recombinant protein) formed per unit of time. In this case, this is usually related with the final biomass obtained. Thus, a *performance index (PI)* is defined by the following expression:

$$PI = \frac{X(T_f)W(T_f) - X(0)W(0)}{T_f} \tag{7}$$

The state variables are initialized with the values: $X(0) = 5$, $S(0) = 0$, $A(0) = 0$, $W(0) = 3$. Due to physical limitations the value of $F_{in,S}(t)$ must be in the range $[0.0; 0.4]$ and $W(t) \leq 5$. The final time (T_f) is set to 25 hours.

2.2 Case Study II

This case study handles a hybridoma reactor described by the equations [12]:

$$\frac{dX_v}{dt} = (\mu - k_d)X_v - \frac{F_1 + F_2}{V}X_v \tag{8}$$

$$\frac{dGlc}{dt} = \frac{F_1}{V}Glc_{in} - \frac{F_1 + F_2}{V}Glc - q_{Glc}X_v \tag{9}$$

$$\frac{dGln}{dt} = \frac{F_2}{V}Gln_{in} - \frac{F_1 + F_2}{V}Gln - q_{Gln}X_v \tag{10}$$

$$\frac{dLac}{dt} = q_{Lac}X_v - \frac{F_1 + F_2}{V}Lac \tag{11}$$

$$\frac{dAmm}{dt} = q_{Amm}X_v - \frac{F_1 + F_2}{V}Amm \tag{12}$$

$$\frac{dMab}{dt} = q_{Mab}X_v - \frac{F_1 + F_2}{V}Mab \tag{13}$$

$$\frac{dV}{dt} = (F_1 + F_2) \tag{14}$$

where the state variables X_v, Glc, Gln, Lac, Amm, Mab, V are the concentrations of viable cells, glucose, glutamine, lactate, ammonia, monoclonal antibodies and culture volume, respectively. The control variables F_1 and F_2 are the volumetric feed rates. The complete kinetic expressions are given in [12].

A single feed problem can be obtained by considering $F = F_1 + F_2$. The target of the optimization process, in this case, is to increase the total amount of monoclonal antibodies produced. So, the PI is given by:

$$PI = \int_0^{T_f} -q_{Mab}X_v(t)V(t) \tag{15}$$

Initialization values for the state variables are the following: X_v=$2.0\times10^8 cells/L$, $Glc = 25g/L$, $Gln = 4g/L$, $Lac = 0g/L$, $Amm = 0g/L$, $Mab = 0g/L$, $V = 0.8L$. T_f is $10days$ and the value of $V(t)$ is constrained by $V(t) \leq V_{max}$.

2.3 Case Study III

This system is a fed-batch bioreactor for the production of ethanol by *Saccharomyces cerevisiae*, firstly studied by Chen and Huang [4]. The aim is to find the substrate feed rate profile that maximizes the final amount of ethanol.

The model equations are the following:

$$\frac{dx_1}{dt} = g_1 x_1 - u\frac{x_1}{x_4} \tag{16}$$

$$\frac{dx_2}{dt} = -10 g_1 x_1 + u\frac{150 - x_2}{x_4} \tag{17}$$

$$\frac{dx_3}{dt} = g_2 x_1 - u\frac{x_3}{x_4} \tag{18}$$

$$\frac{dx_4}{dt} = u \tag{19}$$

where x_1, x_2 and x_3 are the cell mass, substrate and ethanol concentrations (g/L), x_4 the volume of the reactor (L) and u) the feeding rate (L/h).

On the other hand, the kinetic variables g_1 and g_2 are given by:

$$g_1 = \frac{0.408}{1 + \frac{x_3}{16}} \frac{x_2}{0.22 + x_2} \tag{20}$$

$$g_2 = \frac{1}{1 + \frac{x_3}{71.5}} \frac{x_2}{0.44 + x_2} \tag{21}$$

The *performance index (PI)* is given by: $PI = x_3(T_f)x_4(T_f)$.

The final time is set to $T_f = 54$ hours, and the initial values for the state variables are the following: $x_1(0) = 1$, $x_2(0) = 150$, $x_3(0) = 0$ and $x_4(0) = 10$. Additionally, there are physical constraints over the variables, namely: $0 \leq x_4(t) \leq 200$ for all time points and the feeding rate $0 \leq u(t) \leq 12$.

3 The Optimization Algorithms

The aim of the offline optimization is to find the feeding trajectory, represented as an array of real-valued variables, that yields the best performance index. Each variable will encode the amount of substrate to be introduced into the bioreactor, in a given time interval, and the solution will be given by the temporal sequence of such values. In this case, the size of the solution will be determined based on the final time of the process (T_f) and the discretization step (d) considered in the numerical simulation of the model, given by the expression: $\frac{T_f}{d}$. To reduce the solution size, feeding values were defined only at certain equally spaced points,

and the remaining values are linearly interpolated. The size of the genome (G) becomes $G = \frac{T_f}{dI} + 1$, where I stands for the number of points within each interpolation interval. The value of d used in the experiments was $d = 0.005$, for all case studies and the value of I is 200, 100 and 500 in case studies I, II and III, respectively.

The evaluation process, for each solution, is achieved by running a numerical simulation of the defined model, giving as input the feeding values in the solution. The numerical simulation is performed using a linearly *implicit-explicit (IMEX) Runge-Kutta* scheme, used for stiff problems, included in package *OdeToJava* [2]. The fitness value is then calculated from the values of the state variables according to the *PI* defined for each case.

3.1 Differential Evolution

Differential Evolution (DE) is a population-based approach to function optimization that generates trial individuals by calculating vector differences between other randomly selected members of the population. In this work, a variant of the *DE* algorithm called *DE/rand/1* was considered that uses a binomial crossover [13]. In this case, the following scheme is followed, in every generation, for each individual i in the population:

1. Randomly select 3 individuals r_1, r_2, r_3 distinct from i;
2. Generate a trial vector based on: $\boldsymbol{t} = \boldsymbol{r}_1 + F \cdot (\boldsymbol{r}_2 - \boldsymbol{r}_3)$
3. Incorporate coordinates of this vector with probability CR;
4. Evaluate the candidate and use it in the new generation if it is at least as good as the current individual.

3.2 Real-Valued EA

In this work, a real-valued *EA* was adopted that provided good results in previous work [11][6]. A brief presentation of the algorithm is given in this section. A more detailed description can be found in those references. The *EA* uses real-valued representations and a number of mutation and crossover operators to create new solutions, namely:

- *Random Mutation*, which replaces one gene by a new randomly generated value, within the allowed range [7];
- *Gaussian Mutation*, which adds to a given gene a value taken from a Gaussian distribution, with a zero mean.
- the standard *Two-Point crossover*;
- the *Arithmetical* crossover [7];
- the *Sum* crossover, where the offspring genes denote the sum or the subtraction of the genes in the parents.

The mutation operators are applied to a number of genes randomly generated between 1 and N (value of N was 10 in the experiments). In each generation

half of the population is kept from the previous generation and the remaining is replaced by the new individuals created using the above reproduction operators. Selection is performed by using a ranking of the individuals in the population and applying a roulette wheel scheme.

4 Online Optimization

During the fermentation process, some of the state variables can be measured, but its values are scarcely used for closed-loop optimization purposes, and are rather employed to evaluate qualitatively the performance of the process. However, it is possible to develop dynamic optimization algorithms capable of timely reacting to this new knowledge generated by updating the corresponding internal model and generating new solutions.

EAs and *DE* are promising approaches to this real-time optimization task, since they keep a population of solutions that can be easily adapted to perform re-optimization. Indeed, a population of solutions previously obtained can be evaluated under the new scenario and better adapted solutions can be created through the use of reproduction operators. The fact that a set of solutions is kept, and not only the best solution, makes a faster adaptation to new conditions possible, while taking advantage of previous optimisation efforts.

In this work, an online optimization strategy based on *EAs* and *DE* is proposed, working in two stages: before the fermentation process starts, an offline optimization is conducted with the algorithms described in the previous sections. After this preliminary optimization, online optimization algorithms use information gathered by measuring the value of relevant state variables to improve the PI during the real fermentation process. These algorithms react by updating its internal model and reaching a new solution, that is available to be sent back to the fermentation monitoring software.

The version of the *EA*/*DE* used to perform online optimization is similar to the ones described before. When new information regarding the state variables is received, the following steps are followed by both *EA* and *DE*:

1. a starting point (in time) is determined for the re-optimized solution, by adding the time label of the received data with the predicted time necessary to compute a new solution (since it is impossible to reach and therefore apply a solution before that time).
2. the last available population is adapted by removing from the genome of each individual the genes that encode feeding values for elapsed time periods.
3. half of the individuals in the population are replaced by new randomly generated solutions (these individuals are chosen randomly, although the best individual is always kept). This helps in maintaining genetic diversity, a specially needed feature for the optimization in changing landscapes.
4. the internal model of the fermentation used by the *EA*/*DE* is updated with the new information available from the real process and each of the individuals is re-evaluated taking this new knowledge into consideration.

5. the normal process of the *DE* or *EA* proceeds for a given number of iterations.
6. the best solution obtained is sent to the fermentation process and can be used in the real process.

5 Experiments

5.1 Setup

In this study, and given time and physical constraints, real fermentations were not conducted and instead these were replaced by simulating the fermentation process and adding noise to state variables. This process is implemented by considering two interacting components: an *optimizer*, that implements the optimization algorithm (*DE* or *EA*), and a *noise simulator (NS)*, that simulates the real fermentation process adding noise to the state variables.

This is performed by considering that there is a deviation between the model prediction and the behaviour of the process due to inaccuracies of the model. Therefore, for each sampling time, the state variables that represent the real process are obtained from the simulated variables by adding white noise. These new values of the state variables would originate a major deviation of the process from its optimal behavior, which had been defined during offline optimization. To partially compensate for this deviation, the new values. of the state variables will be used by the optimization algorithm to calculate a new feeding profile.

The following sequence of events takes place:

1. an offline optimization is performed by the *optimizer* and its results are passed on to the *NS*, used to compute the predicted values of the state variables. The *optimizer* stops and waits for new information.
2. the variable t, which stores the simulated time in the *NS* is set to $t = 0$.
3. while $t < T_f$ (where T_f denotes the final time of the fermentation process) the following steps are executed:
   (a) the values of all state variables at time t are disturbed by the *NS* by adding/ subtracting noise, given by the original value multiplied by a value taken from an uniform distribution with range $[0, U]$. The new values of the state variables are sent to the *optimizer*.
   (b) the *optimizer* receives this information and runs the steps for online optimization listed in the previous section. The best solution reached is sent to the *NS* that updates its model accordingly.
   (c) the *NS* updates $t = t + \Delta t$

Each run for the initial optimization is stopped after $100,000$ function evaluations. For the *DE*, the population size is set to 20, F was set to 0.5, CR to 0.6. In terms of the real-valued EA, the population size is set to 200. The value of Δt was set to 1 (h.) in case study I, 0.5 (d.) in II and 2 (h.) in III.

5.2 Results

The results will be presented in terms of the mean of the *PI* values obtained in 30 runs, as well as 95% confidence intervals. The Tables 1, 2 and 3 show the results of the algorithms obtained on case studies I, II and III, respectively. In every case, the first column represents the parameter U of the uniform representation used to generate noise (an increase in this parameter implies noisier setups). The next two columns show the results for the *DE* and the *EA* during offline optimization; columns 4 and 5 show the results obtained for the same algorithms, but applying the noise disturbances without changing the solutions of offline optimization (simulating the case where there are discrepancies between model predictions and real processes but without intervention of online optimizers) and, finally, the last two columns show the results obtained by the online optimization.

Table 1. Results obtained by the *DE* and *EAs* in case study I

U	Initial optim.		Initial+noise		Online opt.	
	DE	**EA**	**DE**	**EA**	**DE**	**EA**
0.01	9.47 ± 0.00	8.85 ± 0.04	4.67 ± 0.70	4.79 ± 0.73	9.11 ± 0.14	8.72 ± 0.14
0.02	9.47 ± 0.00	8.83 ± 0.05	4.41 ± 0.75	4.69 ± 0.78	8.80 ± 0.24	8.53 ± 0.25
0.03	9.47 ± 0.00	8.81 ± 0.05	4.20 ± 0.76	4.35 ± 0.81	8.47 ± 0.34	8.17 ± 0.35

Table 2. Results obtained by the *DE* and *EAs* in case study II

U	Initial optim.		Initial+noise		Online opt.	
	DE	**EA**	**DE**	**EA**	**DE**	**EA**
0.01	394.7 ± 0.2	386.3 ± 0.8	371.7 ± 8.5	367.9 ± 7.1	386.2 ± 4.8	379.8 ± 3.8
0.02	394.7 ± 0.2	385.2 ± 0.7	353.9 ± 14.9	351.2 ± 12.3	374.1 ± 9.2	371.8 ± 8.3
0.03	394.7 ± 0.2	386.1 ± 0.9	330.0 ± 23.5	343.0 ± 15.4	364.5 ± 13.0	367.6 ± 11.0

Table 3. Results obtained by the *DE* and *EAs* in case study III

U	Initial optim.		Initial+noise		Online opt.	
	DE	**EA**	**DE**	**EA**	**DE**	**EA**
0.01	20405 ± 4	20374 ± 9	20097 ± 133	20236 ± 108	20421 ± 115	20408 ± 119
0.02	20407 ± 3	20379 ± 7	19832 ± 305	19986 ± 244	20404 ± 243	20392 ± 242
0.03	20405 ± 5	20376 ± 9	19711 ± 357	19938 ± 393	20282 ± 317	20236 ± 335

The first conclusion to draw from the results in that, in every case study, even a low level of noise is enough to clearly disturb the results, although that effect is clearly more visible in case study I. In fact, the levels of noise studied are certainly within the range of the differences observed between model predictions and experimental results in biotechnological processes. However, the consequences in terms of process performance when an open-loop fermentation

(without online optimization) is performed are quite extreme, implying that in many cases the utility of even relatively good models for process optimization with current state-of-the-art optimization techniques (mostly offline approaches) is quite low. Therefore, the results obtained with online optimization strategies indicate that the reward obtained in terms of process productivity is probably enough to justify its implementation and the corresponding costs.

In fact, the results obtained for all 3 case studies are quite close to the ones predicted by offline optimization without added noise, thus implying that the optimization scheme is robust to the levels of noise studied in this work. Furthermore, the degradation of the results that is caused by the increase of U is quite graceful, as an increase in U does not cause dramatic effects in the PI.

A comparison of the results obtained by both optimization algorithms show that DE seems to be more effective than the EAs. The difference is very clear when offline optimization is performed, but decreases when the level of noise increases. In fact, the differences for $U = 0.02$ and 0.03 are not significant from a statistical perspective and in case study II, the EA even outperforms the DE for $U = 0.03$. Nevertheless, if an alternative has to be chosen the DE still has an advantage, since it shows the best results (mean) in almost all scenarios.

6 Conclusions and Further Work

In this work, the task of optimizing feed profiles for fed-batch fermentation problems was approached by proposing optimization algorithms, such as EAs and DE, that are able to implement online optimization strategies, i.e., to perform the optimization simultaneously with the real process. The proposed approach was validated by conducting a number of experiments that used a noise simulator to emulate the differences between the values predicted by the mathematical model and the real values in the fermentation project. The results of the experiments show that even small differences lead to important disruptions in the behavior that was predicted by offline optimization.

The proposed approach to online optimization deals well with the noise and exhibits properties of graceful degradation. When comparing the optimization algorithms, the DE seems the best alternative, but its superiority that seems to decrease when noisier settings are considered.

In future work, the priority is to validate these results by implementing this approach with a real fed-batch fermentation process. Furthermore, other case studies will be tested and distinct optimization algorithms will be taken into account.

Acknowledgment

This work was supported by the Portuguese Foundation for Science and Technology under project POSC/EIA/59899/2004, partially funded by FEDER.

References

1. P. Angelov and R. Guthke. A Genetic-Algorithm-based Approach to Optimization of Bioprocesses Described by Fuzzy Rules. *Bioprocess Engin.*, 16:299–303, 1997.
2. Ascher, Ruuth, and Spiteri. Implicit-explicit runge-kutta methods for time-dependent partial differential equations. *Applied Numerical Mathematics*, 25:151–167, 1997.
3. A.E. Bryson and Y.C. Ho. *Applied Optimal Control - Optimization, Estimation and Control*. Hemisphere Publication Company, New York, 1975.
4. C.T. Chen and C. Hwang. Optimal Control Computation for Differential-algebraic Process Systems with General Constraints. *Chemical Engineering Communications*, 97:9–26, 1990.
5. J.P. Chiou and F.S. Wang. Hybrid Method of Evolutionary Algorithms for Static and Dynamic Optimization Problems with Application to a Fed-batch Fermentation Process. *Computers & Chemical Engineering*, 23:1277–1291, 1999.
6. R. Mendes, I. Rocha, E. Ferreira, and M. Rocha. A comparison of algorithms for the optimization of fermentation processes. In *2006 IEEE Congress on Evolutionary Computation*, pages 7371–7378, Vancouver, BC, Canada, jul 2006.
7. Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, USA, third edition, 1996.
8. H. Moriyama and K. Shimizu. On-line Optimization of Culture Temperature for Ethanol Fermentation Using a Genetic Algorithm. *Journal Chemical Technology Biotechnology*, 66:217–222, 1996.
9. I. Rocha. *Model-based strategies for computer-aided operation of recombinant E. coli fermentation*. PhD thesis, Universidade do Minho, 2003.
10. I. Rocha and E.C. Ferreira. On-line Simultaneous Monitoring of Glucose and Acetate with FIA During High Cell Density Fermentation of Recombinant E. coli. *Analytica Chimica Acta*, 462(2):293–304, 2002.
11. M. Rocha, J. Neves, I. Rocha, and E. Ferreira. Evolutionary algorithms for optimal control in fed-batch fermentation processes. In G.Raidl et al., editor, *Proceedings of the Workshop on Evolutionary Bioinformatics - EvoWorkshops 2004, LNCS 3005*, pages pp.84–93. Springer, 2004.
12. J.A. Roubos, G. van Straten, and A.J. van Boxtel. An Evolutionary Strategy for Fed-batch Bioreactor Optimization: Concepts and Performance. *Journal of Biotechnology*, 67:173–187, 1999.
13. R. Storn and K. Price. Differential Evolution - a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *Journal of Global Optimization*, 11:341–359, 1997.
14. V. van Breusegem and G. Bastin. Optimal Control of Biomass Growth in a Mixed Culture. *Biotechnology and Bioengineering*, 35:349–355, 1990.