Scientific paper

Use of Genetic Algorithm for Fitting Sovova's Mass Transfer Model

Dejan Hrnčič,[1,]* Marjan Mernik[1] and Maša Knez Hrnčič[2]

[1] University of Maribor, Faculty of Electrical Engineering and Computer Science, Programming methodologies laboratory, Smetanova 17, SI-2000 Maribor, Slovenia

[2] University of Maribor, Faculty of Chemistry and Chemical Engineering, Laboratory for separation processes and product design, Smetanova 17, SI-2000 Maribor, Slovenia

** Corresponding author: E-mail: dejan.hrncic@uni-mb.si;*
Tel.: +38622207473, Fax.: +38622207272

Received: 04-02-2010

Abstract

A genetic algorithm with resizable population has been applied to the estimation of parameters for Sovova's mass transfer model. The comparison of results between a genetic algorithm and a global optimizer from the literature shows that a genetic algorithm performs as good as or better than a global optimizer on a given set of problems. Other benefits of the genetic algorithm, for mass transfer modeling, are simplicity, robustness and efficiency.

Keywords: Sovova's mass transfer model, genetic algorithm, parameter estimation.

1. Introduction

The use of supercritical fluids (SCF) as solvents in high pressure processes has environmental, health, safety and chemical benefits, therefore they are sometimes called "Green solvents".[1] Expanded applications are making SCFs the green solvents of the 21st century in numerous processes, like chemical and biochemical reactions, extraction processes, etc.[2] The environmental benefits of the most SCFs, in industrial processes, are in replacement of environmentally far more damaging conventional organic solvents.

Health and safety benefits include the fact that the most important SCFs (supercritical CO_2 and supercritical H_2O) are non-carcinogenic, non-toxic, non-mutagenic, non-flammable and thermodynamically stable. One of the major process benefits has derived from the thermophysical properties of SCFs (high diffusivity, low viscosity, high density and varying dielectric constant) which can be fine tuned by changing the operating pressure and/or temperature.

High pressure and its applications are new tools, nevertheless the extraction of solids with SCF has been already applied on industrial scale processes. The highest capacities are installed for coffee and tea decaffeination, extraction of hop, extraction of spices for production of oleoresins, extraction of bioactive from plants and extraction of oil from degumming residue to obtain highly concentrated and very pure lecithin.[2] Moreover, the extended research for extraction of solids with SCF is in progress. To design an extraction plant and to determine the optimum operating conditions, reliable models are necessary. Therefore, one of the major activities in this field is kinetic modeling. In the literature, several models for supercritical fluid extraction (SFE) can be found.[3–16] Some of them use simple empirical equations that fit to experimental data,[6] others describe extraction rate using a mass transfer coefficient. The description in the second group can be made using the mass transfer coefficient either in the solvent phase[5,9,10] or in the solid phase.[12] Sovova's model, used in this paper, combines both mass transfer coefficients.[17]

In this work, kinetic modeling for SFE curves was used, where the extract mass or yield is represented as a function of time or solvent mass used. The aim of this work was to compare conventional global optimization algorithms with emerging evolutionary algorithms (EAs)[18] on how the mass transfer model fits with the experimental data. The EAs are a population-based stochastic optimization search processes that mimics those biological processes that allow populations of organisms to adapt to their surrounding environments by genetic inheritance and survival of the fittest. The main advantage of the EAs, over the conventional search methods, is that the EAs don't require any additional information about objective functions such as differentiability or continuity. In other words, they do not make any assumptions about the underlying fitness landscape. Hence, the EAs are often used to solve hard real world problems where other heuristic algorithms fail. Many different algorithms such as: genetic algorithm (GA), evolution strategies (ES), evolutionary programming (EP), and genetic programming (GP) belong to the EAs. Recently, many other algorithms such as: ant colony optimization (ACO), differential evolution (DE), particle swarm optimization (PSO), artificial immune systems (AIS), and bees algorithm (BA) can be classified as EAs, too.

The paper is structured as follows. The related work is presented in the second section. In the third section Sovova's mass transfer model is described. The genetic algorithm used for fitting the model is described in depth in the fourth section. In the fifth section the results are presented. The statistical analysis of the results is presented in the sixth section. At the end in the seventh section the conclusion is made. All the parameters of the mass transfer model are described in the notation section.

2. Related Work

Authors use different mathematical models to fit experimental SFE data. Some use mathematical techniques built in common software such as Excel,[19] Matlab,[20] PROC NLIN software of SAS,[21] and adjust them to fit their needs. Others use or implement different algorithms like maximum likelihood method with linear regression,[22] Nelder-Mead simplex method,[23] global optimization algorithm with lexicographical grid method[24] and parallel flows.[25]

Sovova's model for SFE extraction is a commonly used model for SFE of vegetable oils, found in the literature.[26–33] To obtain the curve for the extraction process, model equations are composed from two sets of parameters. In the first set the parameters are dependent from the used material and conditions at which the extraction was made. In the second set the parameters need to be estimated to fit the model curve with the experimental data. The fitting of parameters for Sovova's model was mostly carried out with the use of local optimization tools, although it was also studied using global optimization.[24] A global optimizer[24] uses the lexicographical grid method (LGM) with a local variation algorithm (LVA) to obtain the near global optimum point on which the Nelder-Mead method was used to find the global optimum. The comparison between the global optimization algorithm and the genetic algorithm used in this paper, was made and the results are shown in section 5.

The GAs were already used in parameter estimation in chemical processes. Wolf and Moros[34] used a GA to estimate the rate constants in oxidizing methane to C_2 hydrocarbons. With a GA, the initial estimates were made, which were further optimized with the Nelder-Mead simplex method. For the heterogeneous catalytic reaction, Park and Froment[35] estimated parameters by combining a GA and the Levenberg-Marquardt method. Balland et al.[36] used a GA, with a local convergence method, to estimate simultaneously the kinetic and energetic parameters on the real and complex chemical system. The estimation of kinetic parameters of multi-component photocatalytic process was done by Wang and Kim[37], also with the combination of a GA and simplex local optimization algorithm. Katare et al.[38] estimated the parameters for propane aromatization on a zeolite catalyst with a hybrid algorithm. All mentioned approaches use the combination of a GA with some local search technique. To our knowledge, none of the papers that use a GA were in the area of extraction models.

3. Experimental

3. 1. Sovova's Model

An extraction curve is a plot of total mass of oil extracted vs the time or total mass of solvent used. Sovova's model is designed to describe the extraction curve of vegetable oils. It extends Lack's plug-flow model and assumes that the part of extractable material is easily accessible to the solvent, due to the breaking of the cell structures, which contains the solute, during the milling of the raw material. The other fraction of the solute remains inside the cell structures that were not broken by milling, so its contact with the solvent is more difficult.[17]

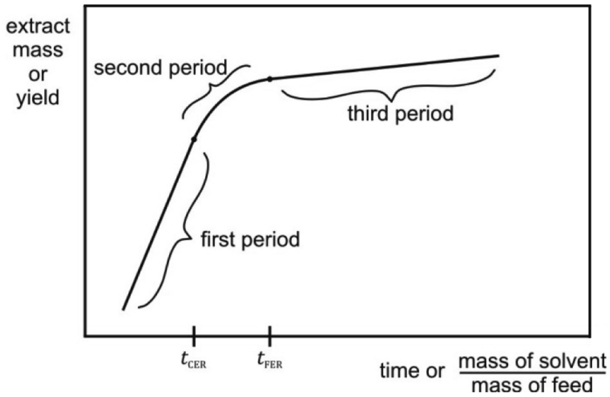

Figure 1: Extraction curve separated into three parts/periods.

The whole extraction curve consists of three periods (Figure 1). In the first period, the easily accessible solute is linear with a slope close to the value of oil solubility in solvent (CO_2). In the second, transition period, the rate of extraction drops rapidly and continues with a third period where the extraction is almost linear but with much a smaller extraction rate then in the first period.

Equations (1)–(4) show Sovova's model that describes the extraction curve as a function of time. Martínez and Martínez[24] have used the global optimization method, combined with the Nelder-Mead method, to find the optimal fit. They have optimized the parameters Z, t_{CER} and t_{FER}. The parameter W was calculated, because a correlation exists between W and t_{CER}. To prove that the GA is able to find the near optimal or optimal solution to this problem, the comparison was made and is presented in section 5.

$$y(t)=\begin{cases} Q_{\mathrm{CO_2}}y_{\mathrm{r}}\left[1\text{-}\exp\left(\text{-}Z\right)\right]t & \text{for } 0\le t\le t_{\mathrm{CER}} \\ Q_{\mathrm{CO_2}}y_{\mathrm{r}}\left[t\text{-}t_{\mathrm{CER}}\exp\left(z_{\mathrm{w}}\text{-}Z\right)\right] & \text{for } t_{\mathrm{CER}}<t\le t_{\mathrm{FER}} \\ m_{\mathrm{SI}}\left[x_0-\dfrac{y_{\mathrm{r}}}{W}\ln\left\{1+\left[\exp\left(\dfrac{Wx_0}{y_{\mathrm{r}}}\right)-1\right]\exp\left[\dfrac{WQ_{\mathrm{CO_2}}}{m_{\mathrm{SI}}}\left(t_{\mathrm{CER}}-t\right)\right]\dfrac{x_{\mathrm{k}}}{x_0}\right\}\right] & \text{for } t>t_{\mathrm{FER}} \end{cases} \qquad (1)$$

The axial coordinate z_{w} in the second extraction period is calculated using equation (2).

$$z_{\mathrm{w}}=\frac{Zy_{\mathrm{r}}}{Wx_0}\ln\left[\frac{x_0\exp\left[WQ_{\mathrm{CO_2}}\left(t-t_{\mathrm{CER}}\right)/m_{\mathrm{SI}}\right]-x_{\mathrm{k}}}{x_0-x_{\mathrm{k}}}\right] \qquad (2)$$

Times t_{CER} and t_{FER} represent the transition between the fast- and slow-extraction periods and are calculated using equations (3) and (4).

$$t_{\mathrm{CER}}=\frac{\left(x_0-x_{\mathrm{k}}\right)m_{\mathrm{SI}}}{y_{\mathrm{r}}ZQ_{\mathrm{CO_2}}} \qquad (3)$$

$$t_{\mathrm{FER}}=t_{\mathrm{CER}}+\frac{m_{\mathrm{SI}}}{WQ_{\mathrm{CO_2}}}\ln\left[\frac{x_{\mathrm{k}}+\left(x_0-x_{\mathrm{k}}\right)\exp\left(Wx_0/y_{\mathrm{r}}\right)}{x_0}\right] \qquad (4)$$

Parameters Z and W are directly proportional to mass transfer coefficients by equations (5) and (6).

$$Z=k_{\mathrm{f}}a_0\rho\Big/\left[\frac{Q_{\mathrm{CO_2}}}{m_{\mathrm{SI}}}\left(1-\varepsilon\right)\rho_{\mathrm{s}}\right]=\frac{F\ m_{\mathrm{SI}}}{Q_{\mathrm{CO_2}}} \qquad (5)$$

$$W=k_{\mathrm{s}}a_0\Big/\left[\frac{Q_{\mathrm{CO_2}}}{m_{\mathrm{SI}}}\left(1-\varepsilon\right)\right]=\frac{S\ m_{\mathrm{SI}}}{Q_{\mathrm{CO_2}}} \qquad (6)$$

The values k_{f} and k_{s} are mass transfer coefficients in CO_2 and solid phase, respectively. Value a_0 represents the specific interfacial area, ρ is the density of the solvent phase, ρ_{s} is the density of the material and ε represents the void fraction in bed.

In this paper, parameters Z, W and x_{k} were chosen to be optimized with GA, similar to parameters F, S and x_{k} fitted in literature.[25]

3. 2. The genetic Algorithm for Sovova's Model

GAs belong to EAs, which use mechanisms inspired by biological evolution, such as mutation, crossover, and selection. The main differences between GA and other well known instances of EAs (ES, EP, and GP) are in the representation of solutions (bit vector, real vector, finite automata, computer program), type of search operators (crossover, mutation), type of selection (stochastic, deterministic), order of operators (selection, crossover, mutation), and how control parameters (*pop_size*, *max_gen*, p_{c}, p_{m}) are handled.

In order to obtain optimal solutions, the search process of the EA is leveraged by two important aspects: exploration and exploitation. Exploration visits entirely new regions of a search space to discover a promising offspring while exploitation utilizes the information from previously visited regions to determine potentially profitable regions to be visited next. To be successful, any search algorithm needs to find a good balance between exploration and exploitation. Many researchers believe that EAs are

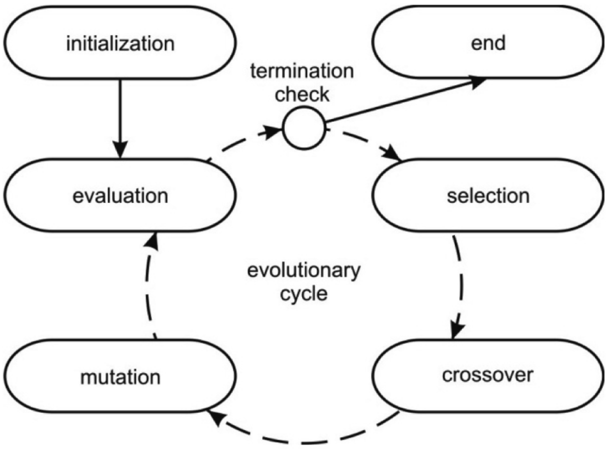

Figure 2: Steps of genetic algorithm.

effective because of a good ratio between the exploration and exploitation. However, how and when to control and balance exploration and exploitation in the search process to obtain even better fitness results, and/or convergence faster, are still an on-going research in EAs.[39–42]

Figure 2 shows the basic steps of the GA. In the beginning, the initial population is often generated randomly. The size of the population, the number of individuals, is defined by the user before the run, with parameter *pop_size*, that might change during the evolution process. In the parameter estimation technique, each individual represents a set of model parameters, which need to be estimated. The model equation(s) can therefore be computed for every individual and the result represents its fitness.

The basic steps of the GA are selection, crossover, mutation, and evaluation. These steps are often problem specific and hence explained in more details later. Every iteration in the evolutionary cycle (Figure 2) is also called a generation. The user, with a *max_gen* parameter, defines the number of generations.

When to stop the GA is checked in the termination check step. There are many possibilities for how long to iterate the GA steps. The algorithm can stop when the result is good enough (within the defined error range), after a certain number of fitness evaluations, or as in our case after a certain number of generations (*max_gen*). The first option is rarely used since the solution is usually not known in advance.

One of the most important parts of GAs is the representation of the individual. The performance and goodness of the fit is often based on the representation. Wolf and Moros[34] transformed each model parameter in the individual into floating point codes, which subdivide information on mantissa, exponent and a sign of exponent. Park and Froment[35] used the binary representation of the parameter set. Balland et al.[36] used number coding where the decimal value between 0 and 1 was converted to an ASCII character chain. The coding of the individual, in this work, is the same as in literature[37,38] and uses the vector of floating points. Each model parameter is represented by a float value. Because the parameters Z, W and x_k were chosen to be estimated, the individual is represented with a vector [Z, W, x_k].

The size of population during the algorithm run can remain the same or it can change. If the offspring generated by a crossover or mutation replaces the parent, the size of the population does not change. Otherwise, as in our case, newly generated individuals can be inserted into the population, which increases its size and leaves the parent individuals untouched. This is useful if the parent has better fitness value then the generated offspring.

3. 2. 1. Selection

It is decided, through the selection, which individuals will survive and make up the population in the next generation. Many different techniques exist to make the selection of the individuals.[18] The most popular are proportional, ranking and tournament selections. The tournament selection was used in this paper. In tournament selection, the algorithm randomly selects *k* individuals from the population and the best one (with the best fitness value) is selected into the next population. Because the population size is changing, it is important that this process is repeated *pop_size* times to reduce the size of population to one that was defined by the user. Both parameters, *k* and *pop_size*, are algorithm parameters defined by the user.

3. 2. 2. Evaluation

The fitness value, for every individual in the population, is calculated with the evaluation method and it is used in the selection process. In this paper, for every individual, the evaluation process calculates how good Sovova's model fits the experimental data with given parameter values. To calculate the fitting of the model to the experimental data, the merit function (7) was used. Value $y(t_i)$ is calculated using equation (1) and represents the calculated amount of an extract at time t_i, while $y_{obs}(t_i)$ represents the experimentally obtained amount of extract.

$$\phi(y, y_{obs}) = \sum_{i=1}^{m} \frac{\left| y(t_i) - y_{obs}(t_i) \right|}{y_{obs}(t_i)} \tag{7}$$

3. 2. 3. Crossover

The crossover operator is often used to explore the search space.[40] This operator is used to mix the "genetic" material between two or more individuals. Generally, the crossover uses two individuals as parents and generates two new individuals or offspring. The two basic crossover mechanisms are the n-point crossover and uniform crossover.[18] The *n*-point crossover cuts the parents of length *L* into *n* + 1 segments, based on randomly selected crossover points. The first offspring is created with the use of odd segments from the first parent and even segments from the second parent and vice versa for the second offspring. In the uniform crossover the decision for every element for the first offspring is made randomly, if it will be taken from first or the second parent. The second offspring is then generated using opposite decisions. Mostly the 1- or 2- point crossover is used.

Because the individuals in this paper are of the short length, the 1-point crossover has been used. For every individual in the population a random number between 0 and 1 is generated. If the number is lower than the crossover probability (p_c), defined by the user, then this individual is selected for crossover as the first parent. The second parent is then randomly selected from the population. After both parents are known, the crossover position needs to be determined. The crossover position in 1-point

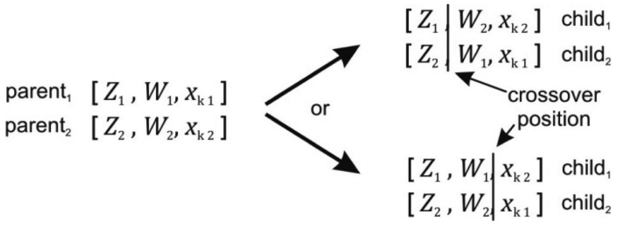

Figure 3: Possible crossovers for vectors $[Z_1, W_1, x_{k1}]$ and $[Z_2, W_2, x_{k2}]$.

crossover is randomly chosen value between 1 and *L*-1. After the position is selected, the new individuals are generated based on rules written earlier in this subsection. Figure 3 depicts this process graphically. As a result, two new individuals are made and inserted into the population.

3. 2. 4. Mutation

The mutation operator is often used to exploit the search space.[40] The mutation operator is used to change a part of the "genetic" material of the individual. In the real-valued representation, the basic element or gene is a floating number. Mutation is therefore implemented as it changes the value (gene) of the individual. For every gene, in the individual, the random number between 0 and 1 is generated. If this number is lower than the probability of mutation (p_m) defined by the user, the gene is mutated. The proposed GA uses equation (8a) or (8b) for mutation.

$$v_n = v + r^3 * \left(v_{\max} - v\right) \tag{8a}$$

$$v_n = v - r^3 * \left(v - v_{\min}\right) \tag{8b}$$

Where v_n represents value that replaces the old value v, r is the random value between 0 and 1, $v_{\max}$ is the maximum value and $v_{\min}$ is the minimum value, both defined by the user for every parameter separately. To achieve that a smaller value change occurred more often, the cube on the random value was used. With such designed selection, crossover, and mutation operators we believe that good diversity of population is obtained. Diversity is one of the important factors that influence exploration and exploitation of EAs.[43]

The boundaries of the model parameters estimated (Z, t_{CER} and t_{FER}) in the global optimizer[24] were studied in detail and limited to values:

$$t_{\mathrm{CER}}^{\min} \le t_{\mathrm{CER}} \le t_{\mathrm{m}},$$

$$Z^{\min} \le Z \le \frac{m_{\mathrm{SI}} x_0}{t_{\mathrm{CER}} y_{\mathrm{r}} Q_{\mathrm{CO_2}}},$$

$$t_{\mathrm{CER}}\left(1+Z\right) \le t_{\mathrm{FER}} \le t_{\mathrm{CER}} + \frac{m_{\mathrm{SI}} x_0}{Q_{\mathrm{CO_2}} y_{\mathrm{r}}}.$$

In contrast to the global approach, the boundaries of parameters estimated in the GA, were not dependent of each other and for parameters *Z* and *W* the minimum value was 0 and the maximum was 10, while the boundaries for parameter x_k were between 0 and x_0 (which fits with the description of x_k).

4. Results and Discussion

The comparison of the GA with the global optimization technique described in the literature[24] was made on the same data. The data describes supercritical fluid extraction with CO_2 of vetiver roots at 40 °C and 200 bars. The first three problems represent a small pilot scale and the next three a large production scales. The values for initial mass of extractable material $x_0 = 0.0619$ and extract solubility $y_r = 0.06$ are equal for all problems.

Firstly, the influence of different control parameters values on results of the GA was measured.[41] The parameters chosen for tuning were population size (*pop_size*), probability of crossover (p_c) and probability of mutation (p_m). For each parameter, three different values were chosen:

- *pop_size*: 1000, 5000, 10000
- p_c: 0.1, 0.2, 0.3
- p_m: 0.05, 0.1, 0.2

The data used, for fitting, is described as problem 1 in literature.[24] The flow rate was 0.85 kg/min and m_{total} was 3.53 kg. The results of GA control parameter tuning are summarized in Table 1. For every set of control parameters, the GA was run for 100 times and Table 1 contains an average, the best and the worst solution found as well as the standard deviation of results. The results are shown as a percentage of average absolute relative deviation (AARD), which is a commonly used measure in extraction modeling and shows the deviation of the model from experimental data. The AARD is calculated using equation 9.

$$\mathrm{AARD} = \frac{100}{\mathrm{m}} \sum_{\mathrm{i=1}}^{\mathrm{m}} \frac{\left|y\left(t_{\mathrm{i}}\right) - y_{\mathrm{obs}}\left(t_{\mathrm{i}}\right)\right|}{y_{\mathrm{obs}}\left(t_{\mathrm{i}}\right)} \quad (\%) \tag{9}$$

Comparison was made between AARD calculated from values for model parameters taken from literature[24] (obtained with the global optimizer) and AARD calculated from values obtained with the proposed genetic algorithm on the same model (Equations (1)–(4)).

Using the solution of the global optimizer[24] for the first problem, the AARD value is 2.09499%. The GA was able to find the solution, which was better than one obtained with the global optimizer, regardless of the control parameters values used (fifth column). We also measured the worst solution from 100 runs (sixth column in Table 1). It can be seen that the worst solutions of the GA are getting closer to the best runs solution with increases in the values of all three control parameters.

The results from Table 1 show that the worst solutions have lower values (are therefore better) than that of the global optimizer, in all cases where the probability of mutation is 0.2. Furthermore the average AARD (fourth column) is falling with increasing control parameter values. However, because the GA is a stochastic and population based technique, the computational time used also increases with the growth of each control parameter value. Therefore we have to make a compromise between the quality of the average result and the time spent for computation. According to the results, given from Table 1, the algorithm gave better results than the global optimizer in all runs, with the use of higher control parameter values. This makes it robust and less dependent of control parameter settings (if they are high enough).

Table 1: Tuning of control parameter of the GA.

pop_size	p_c	p_m	Average AARD	Best AARD	Worst AARD	St. Dev.
1000	0.1	0.05	2.10032	2.09152	2.33522	2.61E-02
1000	0.1	0.1	2.09329	2.09152	2.12256	3.99E-03
1000	0.1	0.2	2.09184	2.09151	2.09348	3.93E-04
1000	0.2	0.05	2.10127	2.09152	2.33437	2.91E-02
1000	0.2	0.1	2.09445	2.09152	2.26253	1.70E-02
1000	0.2	0.2	2.09183	2.09151	2.09309	3.27E-04
1000	0.3	0.05	2.09954	2.09152	2.34141	2.62E-02
1000	0.3	0.1	2.09299	2.09152	2.12326	3.37E-03
1000	0.3	0.2	2.09179	2.09152	2.09327	2.74E-04
5000	0.1	0.05	2.09211	2.09151	2.09919	1.02E-03
5000	0.1	0.1	2.09171	2.09151	2.09910	7.66E-04
5000	**0.1**	**0.2**	**2.09154**	**2.09151**	**2.09178**	**4.40E-05**
5000	0.2	0.05	2.09217	2.09151	2.09528	8.60E-04
5000	0.2	0.1	2.09161	2.09151	2.09243	1.33E-04
5000	0.2	0.2	2.09156	2.09151	2.09174	4.79E-05
5000	0.3	0.05	2.09215	2.09152	2.09783	9.75E-04
5000	0.3	0.1	2.09159	2.09151	2.09212	1.09E-04
5000	0.3	0.2	2.09154	2.09151	2.09175	4.08E-05
10000	0.1	0.05	2.09171	2.09151	2.09318	2.88E-04
10000	0.1	0.1	2.09155	2.09151	2.09177	5.07E-05
10000	0.1	0.2	2.09153	2.09151	2.09164	1.71E-05
10000	0.2	0.05	2.09171	2.09151	2.09363	3.41E-04
10000	0.2	0.1	2.09154	2.09151	2.09173	3.79E-05
10000	0.2	0.2	2.09152	2.09151	2.09159	1.41E-05
10000	0.3	0.05	2.09167	2.09151	2.09275	2.20E-04
10000	0.3	0.1	2.09155	2.09151	2.09174	5.08E-05
10000	0.3	0.2	2.09153	2.09151	2.09158	1.39E-05

Another interesting test, commonly used in different EAs, is the convergence to the solution. Figure 4 shows the convergence of GA for different population sizes (1000, 5000, 10000) with the probability of crossover 0.1 and the probability of mutation 0.2. The plotted lines represent how the fitness value of the best individual in population is changing through generations.

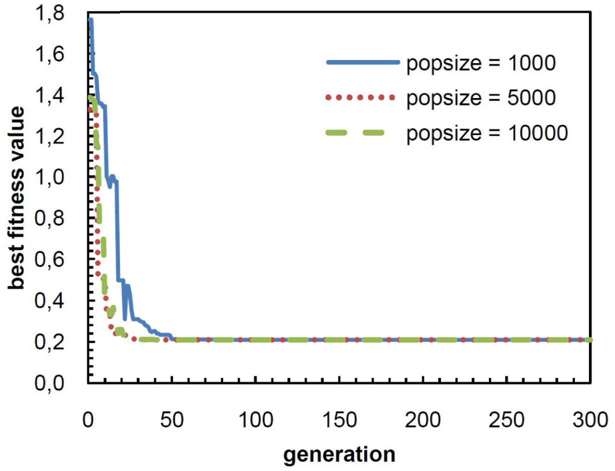

Figure 4: Convergence of GA run for different population sizes (p_c = 0.1, p_m = 0.2).

It can be seen, from Figure 4, that the algorithm converged quickly. In all three cases, it was near final solution after about 30 to 50 generations. After that, only small improvements of fitness value were performed. There was not a big difference in convergence between *pop_size* 5000 and 10000.

Based on the results of parameter tuning and algorithm convergence, the next parameter values of the

GA were used for all six problems described in the literature:[24]

– population size (*pop_size*): 5000
– maximum number of generation (*max_gen*): 300
– probability of crossover (p_c): 0.1
– probability of mutation (p_m): 0.2
– tournament size (k): 2

For every problem, the algorithm was run for 100 times. Commonly, the results obtained from more GA runs were averaged and showed the expected value (with standard deviation) if the user made a random run afterwards. In this case, average value of results was not of much help, because the comparison of estimated model parameters values between the GA and global optimizer was also needed. Therefore, the best and worst solution with belonging estimations of model parameters values were compared with model parameter values obtained with the global optimizer.[24] This comparison is shown in Table 2.

Results from the GA are in all cases better then the results from model parameters values obtained with the global optimizer. However, if the comparison between parameter values (Z, W, t_{CER} and t_{FER}) is done, it can be seen that they are in most cases almost identical. This means that both algorithms found the same optimum in almost the same search space location. It can be concluded that GA found a reasonable solution with a physical meaning without studying the connections between model parameters that was done in literature.[24] The upper and lower bounda-

Table 2: Comparison of results obtained with global optimizer and genetic algorithm for fitting Sovova's model to experimental data. The fitting was measured using AARD.

Extraction conditions	Global optimizer	Best run of GA*	Worst run of GA*
Problem 1: $Q_{CO_2} = 0.85$, $m_{total} = 3.53$	AARD = 2.09499 $Z = 0.083382$, $W = 0.012494$, $t_{CER} = 24.372$, $t_{FER} = 26.411$	AARD = 2.09151 $Z = 0.0833816$, $W = 0.0124254$, $x_k = 0.0307196$, $t_{CER}* = 24.3743120$, $t_{FER}* = 26.4131456$	AARD = 2.09178 $Z = 0.0833820$, $W = 0.0124398$, $x_k = 0.0307221$, $t_{CER}* = 24.3722408$, $t_{FER}* = 26.4109189$
Problem 2: $Q_{CO_2} = 0.85$, $m_{total} = 3.53$	AARD = 1.95147 $Z = 0.104925$, $W = 0.009490$, $t_{CER} = 18.766$, $t_{FER} = 20.740$	AARD = 1.91780 $Z = 0.0938798$, $W = 0.0074551$, $x_k = 0.0298954$, $t_{CER}* = 22.2208750$, $t_{FER}* = 24.3108404$	AARD = 1.91955 $Z = 0.0940318$, $W = 0.0074529$, $x_k = 0.0298973$, $t_{CER}* = 22.1836385$, $t_{FER}* = 24.2734789$
Problem 3: $Q_{CO_2} = 6.3$, $m_{total} = 26.2$	AARD = 1.62209 $Z = 0.228515$, $W = 0.056783$, $t_{CER} = 13.627$, $t_{FER} = 16.762$	AARD = 1.59072 $Z = 0.2285151$, $W = 0.0552940$, $x_k = 0.0141545$, $t_{CER}* = 13.6377716$, $t_{FER}* = 16.7743242$	AARD = 1.59073 $Z = 0.2285151$, $W = 0.0552929$, $x_k = 0.0141544$, $t_{CER}* = 13.6378002$, $t_{FER}* = 16.7743588$
Problem 4: $Q_{CO_2} = 17$, $m_{total} = 71.68$	AARD = 1.60608 $Z = 0.226704$, $W = 0.097600$, $t_{CER} = 14.269$, $t_{FER} = 17.537$	AARD = 1.56185 $Z = 0.2267038$, $W = 0.0943289$, $x_k = 0.0129419$, $t_{CER}* = 14.2915706$, $t_{FER}* = 17.5638633$	AARD = 1.56249 $Z = 0.2267038$, $W = 0.0939244$, $x_k = 0.0129218$, $t_{CER}* = 14.2974381$, $t_{FER}* = 17.5708875$
Problem 5: $Q_{CO_2} = 17$, $m_{total} = 72.26$	AARD = 1.94700 $Z = 0.259076$, $W = 0.087644$, $t_{CER} = 12.182$, $t_{FER} = 15.371$	AARD = 1.84725 $Z = 0.2590756$, $W = 0.0816559$, $x_k = 0.0140226$, $t_{CER}* = 12.3287263$, $t_{FER}* = 15.5528071$	AARD = 1.84727 $Z = 0.2590756$, $W = 0.0816523$, $x_k = 0.0140224$, $t_{CER}* = 12.3287778$, $t_{FER}* = 15.5528703$
Problem 6: $Q_{CO_2} = 6.3$, $m_{total} = 73.1$	AARD = 3.10782 $Z = 0.283353$, $W = 0.220752$, $t_{CER} = 24.083$, $t_{FER} = 31.207$	AARD = 3.07182 $Z = 0.2833528$, $W = 0.2174639$, $x_k = 0.0243632$, $t_{CER}* = 24.1252429$, $t_{FER}* = 31.2576854$	AARD = 3.07291 $Z = 0.2833527$, $W = 0.2168423$, $x_k = 0.0243396$, $t_{CER}* = 24.1404193$, $t_{FER}* = 31.2762118$

* Values for t_{CER} and t_{FER} parameters were calculated from obtained Z, W and x_k using equations (3) and (4).

ries of parameter values were wider, however this did not affect the GA effectiveness. Therefore the GA is also more robust on wider parameter value boundaries, which also means that it is successful on search with larger search space. With high confidence, we can conclude that a random run of the GA with given control parameter values, will give at least as good results as the global optimizer.

Because of almost the same estimated model parameters values, from both algorithms, the extraction curves from the model are the same except for the second problem, where the GA found better fitting with different model parameter values. Figure 5 shows both extraction curves and experimental points for the second problem.

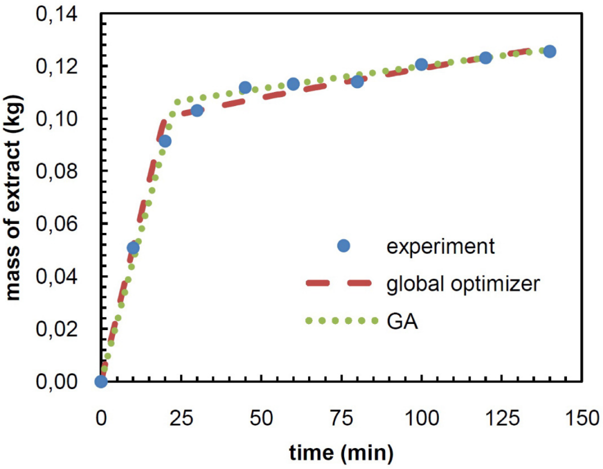

Figure 5: Difference between the extraction curve of the model with estimated parameters of global optimizer and estimated parameters of GA for the second problem.

The computational time, used in the global optimizer, is described with equation $t = 10^{-6} n_1 n_2 n_3 n_{obs}$ (s), where n_1, n_2 and n_3 is the number of grid points for each model parameter and n_{obs} represents the number of experimental points. In case, where n_1, n_2 and n_3 are 100, this means about 10 seconds for data with 10 experimental points. The average time for the run of the GA for the problem with 10 experimental points was 5 seconds. For the GA, the Intel processor with 2.66 GHz was used and the algorithm was written in Java. Although the GA is a population based technique, the time used for computation was better than the time mentioned in the literature for the global optimizer.

5. Statistics

The statistical analysis of the result obtained with GA and global optimizer was made with SPSS Statistics software. One sample t-test was used for comparison of the results. This test was found to be the most appropriate because the results from several GA runs represent one sample that needs to be compared to the already known mean result from the global optimizer. The following hypotheses were defined:

H_{null}: There is no significant difference between results of the GA and results from the global optimization algorithm.

H_{alt}: There is a significant difference between results of the GA and results from the global optimization algorithm.

With parameter tuning, we have determined the control parameters for the GA that give good enough solutions. Although parameter tuning is time consuming and the user without any information about GA efficiency for different control parameter values might use values that do not give good results. Therefore we have also made one sample t-test for the GA run with the worst results obtained, where the control parameter values were lower. The results of the statistical test are shown in the first row of Table 3.

By observing the data from the significance column of Table 3, which is the most important column, it can be concluded that there is not a significant difference in mean value between the results of the global optimizer and the GA with control parameter values 1000 for population size, 0.1 for probability of crossover and 0.05 probability of mutation. The significant level was in this case exceeded ($\alpha > 0.01$). Therefore, in this case, the null hypotheses H_{null} was accepted. Although, in the second case (see Table 3, second row), where the control parameter values of the GA were tuned (pop_size = 5000, $p_c = 0.1$, $p_m = 0.2$) the significant level was not reached ($\alpha < = 0.01$) and this shows a significant difference between mean values of both results. This is expected while all results from 100 runs are lower than the result from global optimizer. In this case the null hypotheses H_{null} is rejected and the alternative hypotheses H_{alt} is accepted.

The same statistical test was performed on control parameter values, presented in Table 1. In all cases, where the mutation value is higher than 0.2, the results from GA are significantly better than results from global optimizer.

Table 3: One sample t-test for comparison of results from GA and global optimizer. Test value is the AARD value of global optimizer for the first problem: 2.09499%

GA control parameters			t	df	Sig. (two tailed)	Mean difference	99% Confidence Interval of the Difference	
pop_size	p_c	p_m					Lower	Upper
1000	0.1	0.05	2.041	99	0.044	0.00533373	–0.0015287	0.0121962
5000	0.1	0.2	–784.487	99	0.000	–0.00344807	–0.0034568	–0.0034393

6. Conclusions

The aim of this work was to check how the GAs perform by fitting the SFE extraction data with Sovova's mass transfer model. The comparison of results obtained with the GA was made with the global optimizer found in literature. Presented results show that the GA algorithm proved successful on all tested problems and that the results were significantly better than results obtained with global optimizer.

Comparing the time needed for computation, the GA proved to be faster than the global optimizer, although the search space was larger, because the boundaries for model parameters fitted were wider. The GAs are also easy to implement and can be used for different optimization problems. They are robust and capable of obtaining good solutions for small problems, as in this case as well as solving much harder problems as in the literature.[44] The algorithm implemented in this case could easily be used, with minor changes in implementation, for parameter estimation of any other model (even with a higher number of model parameters and therefore larger search space).

7. Acknowledgement

The authors are grateful to Prof. Dr. Rafiqul Gani for his helpful comments.

Notation

$y(t)$	mass of extract (kg)
Q_{CO_2}	solvent mass flow rate (kg/min)
y_r	extract solubility in the solvent at given pressure and temperature
Z	model parameter related to convection in the fluid phase of the extraction bed
W	model parameter related to diffusion in the solid phase of the extraction bed
t	time (min)
t_{CER}	end of the first extraction period (min)
t_{FER}	end of the second extraction period (min)
m_{SI}	mass of non-extractable material (extract-free) (kg)
m_{total}	mass of material (extract + extract-free) (kg)
x_0	initial mass of extractable material, relative to mass of non-extractable material
x_k	initial mass of extractable material in intact cells, related to mass of non-extractable material

8. References

1. Ž. Knez, *Chem. Eng. Res. Des.* **2004**, *82*, 1541–1548.
2. Ž. Knez, M. Škerget, M. Knez Hrnčič, in: S. Rizvi (Ed.): Separation, extraction and concentration processes in the food, beverage and nutraceutical industries, Woodhead Publishing Limited, Cambridge, **2010** (In Press), Chapter: Extraction using supercritical fluids.
3. G. Brunner, *Ber. Bunsen. Phys. Chem.* **1984**, *88*, 887–891.
4. M. J. Cocero, J. Garcia, *J. Supercrit. Fluid.* **2001**, *20*, 245–255.
5. M. Cygnarowicz-Provost, D. J. O'Brien, R. J. Maxwell, J. W. Hampson, *J. Supercrit. Fluid.* **1992**, *5*, 24–30.
6. M. M. Esquível, M. G. Bernardo-Gil, M. B. King, *J. Supercrit. Fluid.* **1999**, *16*, 43–58.
7. S. R. S. Ferreira, M. A. A. Meireles, *J. Supercrit. Fluid.* **2002**, *54*, 263–269.
8. M. Goto, J. M. Smith, B. J. McCoy, *Ind. Eng. Chem. Res.* **1990**, *29*, 282–289.
9. E. A. Lack, Kriterien zur Auslegung von Anlagen für die Hochdruckextraktion von Naturstoffen, Ph.D. thesis, TU Graz, **1985**.
10. A. K. K. Lee, N. R. Bulley, M. Fattori, A. Meisen, *J. Am. Oil Chem. Soc.* **1986**, *63*, 921–925.
11. R. Montero-Vázquez, J. Cota-Galán, A. Roselló-Segado, F. J. González-Vila, *Chemosphere* **2003**, *53*, 789–793.
12. A. V. Pekhov, G. K. Goncharenko, *Maslozhirovaja promyshlennost'* **1968**, *34*, 26–29.
13. S. Quispe-Condori, D. Sánchez, M. A. Foglio, P. T. V. Rosa, C. Zetzl, G. Brunner, M. A. A. Meireles, *J. Supercrit. Fluid.* **2005**, *36*, 40–48.
14. E. Reverchon, G. Donsi, L.S. Osséo, *Ind. Eng. Chem. Res.* **1993**, *32*, 2721–2726.
15. H. Sovova, *J. Supercrit. Fluid.* **2005**, *33*, 35–52.
16. M. Škerget, Ž. Knez, *Comput. Chem. Eng.* **2001**, *25*, 879–886.
17. H. Sovova, *Chem. Eng. Sci.* **1994**, *49*, 409–414.
18. Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Third Edition, Springer-Verlag, New York, **1996**.
19. A. Berna, A. Tárrega, M. Blasco, S. Subirats, *J. Supercrit. Fluid.* **2000**, *18*, 227–237.
20. L. M. A. S. Campos, E. M. Z. Michielin, L. Danielski, S. R. S. Ferreira, *J. Supercrit. Fluid.* **2005**, *34*, 163–170.
21. N. P. Povh, M. O. M. Marques, M. A. A. Meireles, *J. Supercrit. Fluid.* **2001**, *21*, 245–256.
22. E. M. B. D. Sousa, J. Martínez, O. Chiavone-Filho, P. T. V. Rosa, T. Domingos, M. A. A. Meireles, *J. Food Eng.* **2005**, *69*, 325–333.
23. J. A. Nelder, R. Mead, *Comput. J.* **1965**, *7*, 308–313.
24. J. Martinez, J. M. Martinez, *Comput. Chem. Eng.* **2008**, *32*, 1735–1745.
25. H. Sovova, *Chem. Eng. Sci.* **1994**, *49*, 415–420.
26. O. Bensebia, D. Barth, B. Bensebia, *J. Supercrit. Fluid.* **2009**, *49*, 161–166.
27. M. G. Bernardo-Gil, M. Casquilho, M. M. Esquivel, *J. Supercrit. Fluid.* **2009**, *49*, 32–36.
28. E. Cassel, R. M. F. Vargas, G. W. Brun, *Nat. Prod. Commun.* **2008**, *3*, 373–378.
29. X. J. Han, L. M. Cheng, R. Zhang, *J. Food Eng.* **2009**, *92*, 370–376.

30. E. Langa, J. Cacho, A. M. F. Palavra, *J. Supercrit. Fluid.* **2009**, *49*, 37–44.
31. E. Langa, G. Della Porta, A. M. F. Palavra, *J. Supercrit. Fluid.* **2009**, *49*, 174–181.
32. B. Nagy, B. Simandi, *J. Supercrit. Fluid.* **2008**, *46*, 293–298.
33. E. Talansier, M. E. M. Braga, P. T. V. Rosa, *J. Supercrit. Fluid.* **2008**, *47*, 200–208.
34. D. Wolf, R. Moros, *Chem. Eng. Sci.* **1997**, *52*, 1189–1199.
35. T. Y. Park, G. F. Froment, *Comput. Chem. Eng.* **1998**, *22*, S103–S110.
36. L. Balland, L. Estel, J. M. Cosmao, N. Mouhab, *Chemometr. Intell. Lab.* **2000**, *50*, 121–135.
37. L. Wang, C. N. Kim, *Korean J. Chem. Eng.* **2001**, *18*, 652–661.
38. S. Katare, A. Bhan, J. M. Caruthers, W. N. Delgass, V. Venkatasubramanian, *Comput. Chem. Eng.* **2004**, *28*, 2569–2581.
39. J. Brest, S. Greiner, B. Boškovič, M. Mernik, V. Žumer, *IEEE T. Evolut. Comput.* **2006**, *10*, 646–657.
40. A. E. Eiben, C. A. Schippers, *Fund. Inform.* **1998**, *35*, 1–16.
41. A. E. Eiben, R. Hinterding, Z. Michalewicz, *IEEE T. Evolut. Comput.* **1999**, *3*, 124–141.
42. S. H. Liu, M. Mernik, B. Bryant, *Int. J. Knowl. Base. Intell. Eng. Syst.* **2009**, *13*, 185–206.
43. N. Chaiyaratana, T. Piroonratana, N. Sangkawelert, *J. Heuristics* **2007**, *13*, 1–34.
44. C. Grosan, A. Abraham, H. Ishibuchi, Hybrid Evolutionary Algorithms, Springer, Berlin, **2007**.

Povzetek

V članku predstavimo genetski algoritem s spremenljivo populacijo za določitev parametrov modela prenosa snovi Sovova. Primerjava rezultatov med genetskim algoritmom in globalnim optimizacijskim algoritmom povzetim iz literature pokaže, da so rezultati genetskega algoritma prav tako dobri ali boljši od rezultatov globalnega optimizacijskega algoritma na dani množici problemov. Ostale prednosti genetskega algoritma za modeliranje krivulje prenosa snovi so preprostost, robustnost in učinkovitost.