CHEMICAL ENGINEERING TRANSACTIONS

VOL. 74, 2019

A publication of
AIDIC
The Italian Association
of Chemical Engineering
Online at www.cetjournal.it

Guest Editors: Sauro Pierucci, Jiří Jaromír Klemeš, Laura Piazza

ISBN 978-88-95608-71-6; **ISSN** 2283-9216

DOI: 10.3303/CET1974101

An Example of Multi-Objective Optimization for Dynamic Processes

Eric S. Fraga

Department of Chemical Engineering, University College London (UCL), Torrington Place, London WC1E 7JE, United Kingdom
e.fraga@ucl.ac.uk

Recent research activity has shown an interest in multi-objective optimization of dynamic processes, both for design and for real-time control; see Liu et al. (2018), Valderrama & Ruiz (2018), and de Sousa Santos et al. (2018) for some recent examples. Many of the methods proposed for these problems are based on converting the multi-objective problem into a single objective through the use of weights. Some works, for instance, Liu et al. (2018), however, consider the multi-objective problem directly but even in these cases, the problem is treated as a single objective problem with other objectives incorporated as constraints. All of these methods may not necessarily generate a satisfactory trade-off or Pareto optimal curve when this trade-off curve is non-convex. In this paper, we illustrate the use of a plant propagation nature inspired algorithm for the solution of design and control problems for dynamic processes. Being population based, it is able to identify an approximation to the Pareto optimal frontier simultaneously. A simple example from the literature is used to demonstrate how this problem can be formulated. The open source, freely available, Fresa application (http://www.ucl.ac.uk/~ucecesf/fresa.html),written in Julia, is used and the discussion concentrates on the problem formulation.
The result shows how a multi-objective approach may lead to better insight into the design or control issues. The illustrated problem was originally formulated as a single objective problem. In doing so, information about alternative designs and how these could affect the performance of the system may be hidden. A multi-objective formulation allows the engineer to make better informed design decisions.

1. Introduction

The design of processes with dynamic behaviour, i.e. not steady state, poses challenges for model based optimization methods. A number of approaches have been presented in the literature and a recent article presents one standard approach for mathematical programming (Biegler, 2018). However, recent research activity has shown an interest in multi-objective optimization of dynamic processes, both for design and for real-time control; see Liu *et al*. (2018), Valderrama & Ruiz (2018), de Sousa Santos *et al*. (2018), and Beck *et al.* (2015) for some recent examples. Many of the methods proposed for these problems are based on converting the multi-objective problem into a single objective through the use of weights. Some works, for instance, Liu *et al*. (2018), however, consider the multi-objective problem directly but even in these cases, the problem is treated as a single objective problem with other objectives incorporated as constraints. All of these methods may not necessarily generate a satisfactory trade-off or Pareto optimal curve when this trade-off curve is non-convex. Generating the Pareto optimal trade-off curve directly may have some benefits in gaining insight into the design trade-offs.

2. A plant propagation algorithm

A wide range of methods exist for multi-objective optimization. Most are population based, such as the NSGA-II (Deb, 2000) genetic algorithm based approach which is able to handle constraint violations and ranks solutions according to dominance properties. Our own approach has also been nature-inspired, specifically developing a *plant propagation algorithm* that mimics how strawberry plants propagate in search of better soil

Paper Received: 15 May 2018; Revised: 27 September 2018; Accepted: 12 January 2019

Please cite this article as: Fraga E., 2019, An Example of Multi-objective Optimization for Dynamic Processes, Chemical Engineering Transactions, 74, 601-606 DOI:10.3303/CET1974101

and nutrients (Salhi and Fraga, 2011), as described below. In the original development of this algorithm, the method was used for solving single objective optimization problems in process design. Subsequently, the algorithm has been applied to multi-objective design problems in the design of integrated off-grid energy systems (Fraga and Amusat, 2016) and the control of a beer fermentation process (Rodman et al., 2018). The latter motivated the research in this paper: to explore how the use of multi-objective optimization could be used to gain insight into the design of dynamic processes.
The multi-objective algorithm is shown in Figure 1.
An implementation of this algorithm, in the Julia, language(http://julialang.org/) has been developed and is available online (http://www.ucl.ac.uk/~ucecesf/fresa.html) . The rest of this paper discusses how this algorithm can be applied to multi-objective dynamic optimization problems.

Given: $f(x)$, n_g, n_p, n_r.
Output: z, set representing approximation to Pareto front.
$p \leftarrow$ initial random population of size n_p
for n_g generations **do**
  prune population p, removing **similar** solutions
  $N \leftarrow \text{fitness}(p)$ ▷ rank based
  $\tilde{p} \leftarrow \phi$
  for $i \leftarrow 1, \ldots, n_p$ **do**
    $j \leftarrow \text{select}(p, N)$ ▷ fitness based
    for $k \leftarrow 1, \ldots, n_r \propto N_j$ **do**
      $\tilde{p} \leftarrow \tilde{p} \cup \text{neighbour}(x_j, 1 - N_j)$ ▷ propagation
    end for
  end for
  $p \leftarrow \tilde{p} \cup \{x_i | x_i \in p \wedge f(x_i) \text{ nondominated}\}$ ▷ **elitism**
end for
$z \leftarrow \{x_i | x_i \in p \wedge f(x_i) \text{ nondominated}\}$ ▷ Pareto set

Figure 1. A plant propagation algorithm for multi-objective optimization.

Highlighted in the algorithm are the following terms:

1. Fitness: The Fresa plant propagation algorithm is based on balancing *exploitation* (local search) with *exploration* (global search). The fitness of a solution (a design in the search space) affects the search procedure in two ways: the number of new solutions generated (using random perturbations of the given solution) is proportional to the fitness and the distance or amount of perturbation is inversely proportional to the fitness. The design space around the best solutions is search intensively (exploitation) while poor solutions lead to greater exploration.
2. Selection: The selection mechanism is based on tournament selection of size 2 which provides some selection pressure towards the best solutions but gives all solutions a good chance of propagating.
3. Neighbour: The neighbour generation method uses the fitness to define a new perturbed solution with the amount of perturbation inversely proportional to the fitness. Fresa is written in the Julia language (Bezanson et al., 2017). Julia uses *multiple dispatch* for identifying the function or method to invoke which makes it possible to tailor the behaviour of the Fresa algorithm to problem specific definitions of neighbours. The default implementation includes random perturbations for floating point design variables within bounded domains.

In the following section, a case study will be used to illustrate how Fresa is adapted to a specific problem in the multi-objective design of a dynamic system.

3. Illustrative case study

To illustrate the use of a multi-objective optimization procedure for the optimal design of dynamic processes, we will consider a simple reactor design problem (Čižniar and Latifi, 2005). The aim is to identify the best temperature control profile to achieve the maximum production of the desired product, B, given the following reaction system: [1i]A → B → C. C is an undesired side-product. The first reaction is second order while the second is first orer. The kinetic rate constants are $k_1 = 4000\ e^{-2500/T}$ and $k_2 = 620{,}000\ e^{-5000/T}$ with T in Kelvin. The code for the reactor model, in Julia, is shown in Figure 2.

```
function reactor(x :: Array{Float64,1},
                 p :: TemperatureProfile,
                 t :: Float64
                 ) :: Array{Float64,1}
    T = temperature(p,t)
    k = rates(T)
    return [-k[1] * x[1]^2
            k[1] * x[1]^2 - k[2] * x[2]
            k[2] * x[2]]
end

function simulation(profile :: TemperatureProfile)
    x0 = [1.0, 0.0, 0.0]
    tspan = (0.0,1.0)
    prob = ODEProblem(reactor, x0, tspan, profile)
    DifferentialEquations.solve(prob)
end
```

Figure 2. The reactor model code including the differential equation method available in Julia for simulating the reactions.

Although the original work addressed this problem as a single objective optimization based design problem, we treat this as a multi-objective optimization problem by simultaneously wishing to minimize the production of C. This will allow us to investigate the trade-offs between maximizing the desired product while minimizing undersired products. An industrial application that could benefit from a multi-objective consideration, for example, would be the design of a gasification reactor in a power plant (Sofia et al., 2014).

```
struct TemperatureProfile
    T0 :: Float64              # initial temperature
    Tf :: Float64              # final temperature
    t1 :: Float64              # time point
    a :: Float64               # coefficients of quadratics
    b :: Float64
    c :: Float64
    d :: Float64
    e :: Float64
    f :: Float64
    function TemperatureProfile(T0, Tf, t1)
        a = T0
        b = 0
        c = (Tf-T0)/t1
        d = (Tf*t1 - T0) / (t1 - 1)
        e = (2*T0 - 2*Tf) / (t1 - 1)
        f = (Tf - T0) / (t1 - 1)
        new(T0, Tf, t1, a, b, c, d, e, f)
    end
end
```

Figure 3. Code for defining a temperature profile given three arguments: the initial temperature, T0, the final temperature, Tf, and the interior point in the time domain.

3.1 The search space

The key adaptation of Fresa for this problem is the definition of the search space. This space is defined implicitly by the specification of the design variables. The design variables are the temperature control profile values through the operation of the reactor. Two approaches could be considered: 1) the use of discrete points in the time domain with temperature values defined at each of those points, with an associated interpolating procedure to determine the temperature throughout the domain; 2) the use of a continuous representation for the temperature profile. The first approach is straightforward and requires the *a priori* selection of control points in the time domain (although an adaptive or evolutionary procedure for these points could also be used (Rodman et al., 2018)). The second approach gives more control over the shape of the temperature profile and therefore enables the engineer to provide insight directly into the definition of the search space. We will consider the second case as a result.

```
function temperature(p :: TemperatureProfile,
                     t :: Float64) :: Float64
    if t <= p.t1
        T = p.a + p.b*t + p.c*t^2
    else
        T = p.d + p.e*t + p.f*t^2
    end
    return T
end
```

Figure 4. Code for determining the temperature at any time t, using the piecewise quadratic representation of a temperature profile as defined in Figure 3.

The temperature profile will be defined by the following specifications:
1. The slope of the profile will be 0 at the start time.
2. A time point in the interior of the domain will be chosen and the profile will consist of two quadratic functions, one for the profile on either side of this time point. The quadratics will have the same slope at that point.
3. The slope of the profile will be 0 at the end time as well.

Two quadratic functions imply the need to define 3 times 2 coefficients. However, the conditions above reduce the degrees of freedom to 3: the initial temperature, the time point in the interior, and the final temperature. The coefficients are then a function of the conditions specified above.

The code for defining a new data type which defines the search space, in this case the possible temperature profiles, is shown in Figure 3. The actual temperature at any time t is obtained by evaluating the function shown in Figure 4. This is used by the reactor model directly, as shown in Figure 2.

3.1 Fitness and selection

For single objective problems, the fitness of solutions can be a straightforward function of the value of the desired objective. For multi-objective problems, there are different fitness mappings that could be considered. For instance, NSGA-II (Deb, 2000) uses the concept of dominance in the objective function space to assign each solution a rank. All solutions that are non-dominated in the whole population are given the same rank and hence the same fitness. The next rank, and hence lower fitness, is given to those solutions that would be non-dominated were the first ranked solutions removed from the population. And so on. Although this procedure is effective, it can have a significant impact on computation as the number of objectives and the population size grow due to an $O(n^3)$ growth rate, where n is the population size.

We use, instead, a faster method based on the individual ranking of each solution with respect to each objective function. This procedure has been described in another paper (Fraga and Amusat, 2016) and uses a Hadamard multiplication to combine the individual rankings into a single ranking. The advantage of this fitness method, besides the computational cost, is the emphasis it gives to solutions that give the best results for the individual objective functions. This has the effect of ensuring that the Pareto trade-off curve identified is as broad as possible, at the possible expense of gaps in the middle regions of this curve.

4. Results

Invoking Fresa, with a population size of 20, 5 runners maximum, 100 generations, and using the Hadamard fitness function, we obtain a population of solutions at the end shown in Figure 4. The axes in this plot are the concentrations of C (y-axis) and B (x-axis), the two criteria. We wish to maximize B and minimize C; therefore, the most desirable solutions would be those that lie in the bottom right corner. The non-dominated solutions, i.e. those that approximate the Pareto frontier, are all the points along the bottom branch of the curve until the right-most point. The results demonstrate that Fresa is able to identify the trade-off between these objectives. The fitness function (Fraga and Amusat, 2016) places an emphasis on solutions which are best for each individual criterion and this figure shows an increased density of solutions at the two extrema: the left end of the bottom branch and the point on the right where the direction of the points changes.

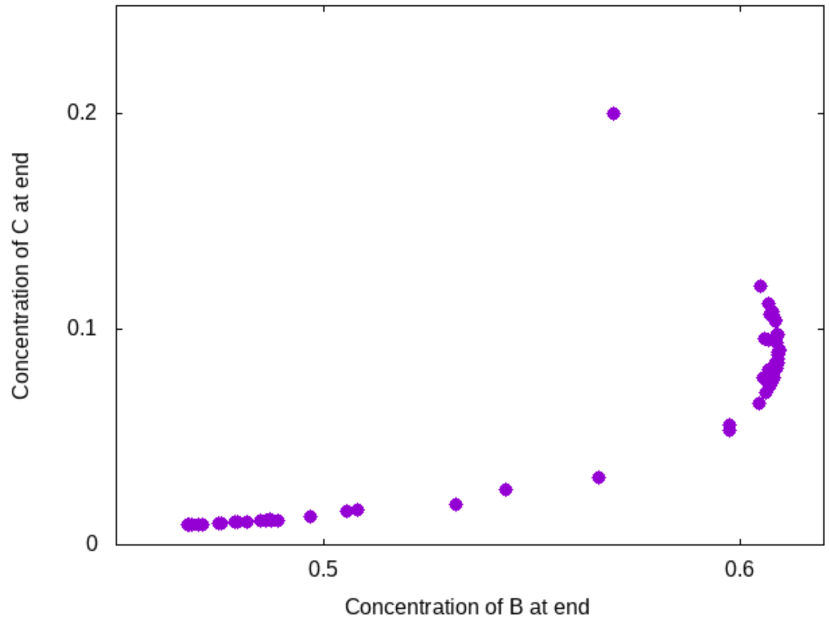

Figure 4. Plot of all solutions in the final population. The set of non-dominated points are those along the bottom of the set of points from the left up and right towards the extreme right-most point.

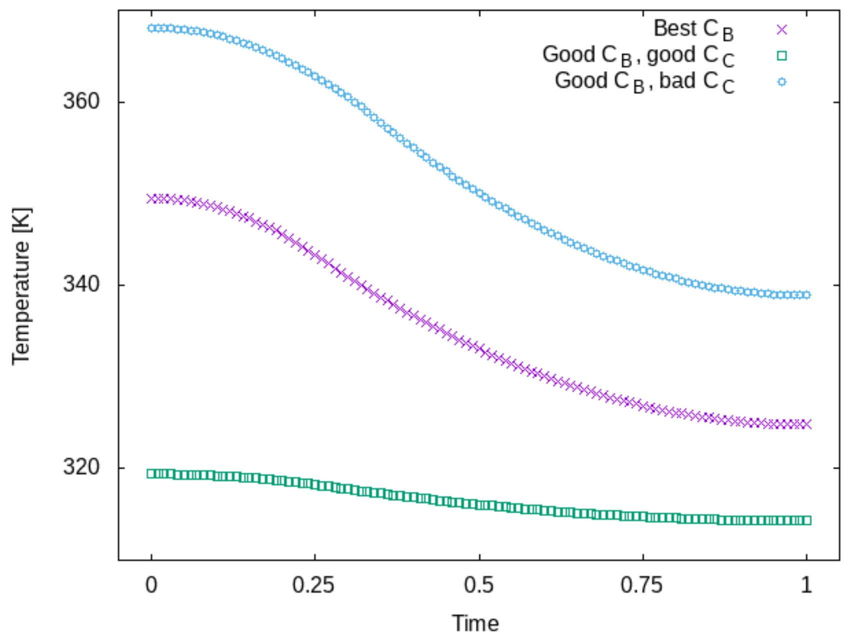

Figure 5. Plot of three selected temperature profiles: 1) best C_B solution which obtains highest concentration of B at the end and represents the solution at the rightmost point in the plot of all design points in the final population; 2) solution with good final C_B value and good C_C value, essentially a reasonable compromise between the two objective functions; and, 3) a solution with similar C_B value but much worse value of C_C.

Figure 5 presents three temperature profiles selected from the population at the end. All have the expected shapes given the definition of the search space; each profile has a zero slope at start and also at the end and each exhibits a smooth change in temperature throughout the time domain. The profiles presented all have a higher initial temperature at the beginning than at the end. However, the search space includes solutions with the opposite behaviour, ones where the temperature is higher at the end than at the start but these solutions do not have good objective function values and so are not present in significant numbers in the final population. It is further interesting to note that the two solutions which obtain good values for the final concentration of B but have significantly different values for the final concentration of C have such different temperature profiles.

4. Conclusions

This paper has presented the use of the Fresa system for multi-objective optimization applied to a problem in design with dynamic behaviour. Fresa is easy to tailor to new problems by supporting the definition of problem specific data structures to represent the space of desired solutions. This enables the engineer to incorporate insight into the definition of the search space which has two benefits: only solutions that have the qualitative or quantitative features the engineer would want will be presented and the reduction of the search space to only those solutions of interest will reduce the computation would want will be presented and the reduction of the search space to only those solutions of interest will reduce the computational requirements.

Fresa supports the simultaneous consideration of multiple objectives. For engineering, a multi-objective fitness function that gives emphasis to extrema for each objective ensures that the final population has a good representation of solutions that are likely to be of interest to the engineer. The resulting solutions can help gain further insight. In the case of the dynamic operation of a reactor, we observe that solutions which exhibit similar quality solutions for one objective (e.g. maximization of the concentration of B) can exhibit different outcomes for other objectives (e.g. minimization of the concentration of C). Such insight would be difficult to obtain through the use of single objective optimization methods.

The Fresa system is available for download from the author’s web pages.

References

Beck J., Friedrich D., Brandani S., Fraga E.S., 2015, Multi-objective optimisation using surrogate models for the design of VPSA systems, Computers & Chemical Engineering, 82, 318–329.

Bezanson J., Edelman A., Karpinski S., Shah, V. B., 2017, Julia: A Fresh Approach to Numerical Computing. SIAM Review, 59, 65–98.

Biegler L.T., 2018, Advanced optimization strategies for integrated dynamic process operations, Computers & Chemical Engineering, 114, 3–13.

Čižniar M.F., Salhi D., Latifi M., 2005, A MATLAB package for orthogonal collocations on finite elements in dynamic optimisation, Proceedings of the 15th int. Conference process control ’05.

De Sousa Santos L., Ferreira de Souza K.M., Ribeiro Bandeira M., Ruiz Ahón V.R., Cunha Peixoto F., Martinez Prata, D. 2018. Dynamic optimization of a continuous gas lift process using a mesh refining sequential method. Journal of Petroleum Science and Engineering 165, 161–170.

Deb K., 2000, An efficient constraint handling method for genetic algorithms, Comput. Methods Appl. Mech. Engng., 186, 311–338.

Fraga E.S., Amusat O.O., 2016, Understanding the impact of constraints: A rank based fitness function for evolutionary methods, in: P.M. Pardalos, A. Zhigljavsky and J. Zilinskas, eds., Advances in stochastic and deterministic global optimization, Springer, 243–254.

Liu C., Gong Z., Teo K.L., Loxton R., Feng, E., 2018, Bi-objective dynamic optimization of a nonlinear time-delay system in microbial batch process, Optimization Letters, 12, 1249–1264.

Rodman A.D., Fraga E.S., Gerogiorgis D., 2018, On the application of a nature-inspired stochastic evolutionary algorithm to constrained multi-objective beer fermentation optimisation, Computers & Chemical Engineering, 108, 448–459.

Salhi A., Fraga E.S., 2011, Nature-inspired optimisation approaches and the new plant propagation algorithm, Proceedings of ICEMATH 2011, the international conference on numerical analysis and optimization, K2, 1–8.

Sofia D, Coca Llano P., Giuliano A., Iborra Hernández M., García Peña F., Barletta D., 2014, Co-gasification of coal–petcoke and biomass in the Puertollano IGCC power plant, Chemical Engineering Research and Design, 92, 1428-1440.

Valderrama F., Ruiz F., 2018, An optimal control approach to steam distillation of essential oils from aromatic plants, Computers & Chemical Engineering, 117, 25–31.