Local approximation to complex models for efficient optimisation: application to crystallisation processes
Proceedings of European Congress of Chemical Engineering (ECCE-6)
Copenhagen, 16-20 September 2007

Local approximation to complex models for efficient optimisation: application to crystallisation processes

A. Yang, X. Ou, G. A. Montague, E. B. Martin

School of Chemical Engineering and Advanced Materials, Newcastle University, Newcastle Upon Tyne, UK

Abstract

Recent advances in mathematical modelling have enabled the development of detailed mechanistic models to support the design and analysis of chemical processes or products. However in those cases where the resulting model is highly complex, it can be extremely time consuming to solve the model numerically, thereby limiting its practical applicability. This is especially the case where a complex model is used for the numerical optimisation of a process thereby necessitating the execution of a large number of evaluations of the model. To make such an optimisation task computationally tractable, the idea of approximating complex mechanistic models with statistical models is investigated. In particular, an iterative optimisation procedure is implemented which makes use of a Gaussian process model (GPM) to locally approximate the complex mechanistic model. The GPM is successively updated according to the results obtained from evaluating the original model at locations, in different sub-regions, searched by the optimisation algorithm. The potential of this method in terms of reducing the number of evaluations of the original model, and hence lowering the computational cost, is confirmed through its application to the optimisation of a batch cooling crystallisation process.

Keywords: mathematical modelling, local approximation, statistical modelling, Gaussian process model, simulation-based optimisation

1. Introduction

Recent advances in mathematical modelling have enabled the development of detailed mechanistic models to support the design and analysis of chemical processes or products. However, numerical solutions of complex models can be extremely time consuming. In particular, when a complex model is used for numerical optimisation, this often involves the execution of a large number of evaluations of the model, consequently the potential computational burden can be prohibitive even with today's powerful computing facilities. To make such an optimisation task computationally tractable, it is important to apply appropriate treatments that reduce the complexity of the models within the optimisation framework.

Marquardt (2002) reviewed a number of model order reduction and model simplification methodologies for reducing the complexity of models. Basically the goal of model order reduction is to reduce the size of the model in terms of the number of variables/equations. This approach generally requires detailed knowledge about the composition of the original equation system that forms the model. In contrast, model simplification reduces the complexity of a model by approximating complex functional expressions in the model with simpler functions while preserving the order of the original model. A well-known application of model simplification is in process simulation where a rigorous physical property model is approximated by a simpler empirical or semi-empirical model (e.g. Leesley & Heyen, 1977; Chimowitz et al., 1983). The latter model is only valid within a particular range of process conditions hence requires to be updated when the process conditions change during a simulation run. A similar approach is taken by Bezzo et al. (2005) for approximating rigorous CFD models in a multizonal/CFD modelling framework.

In this paper, interest is on how to efficiently perform optimisation of a chemical process system for which a highly complex model has been developed. In particular, this model is assumed to be implemented in a software code such that the only outcome is its execution, with extremely high computational costs. No other information is directly available from its execution relating to model structure or derivatives. This type of problem has been referred to as the optimization of expensive black-box functions (Jones et al., 1998). To solve such problems, empirical models have been developed based on the data generated by “sampling” (i.e. evaluating) the black-box objective functions during an optimization process. More specifically, a model is often used as a surrogate for the objective function (or one variable in the function) to provide an approximation. This concept is similar to the model simplification approach discussed earlier.

When utilising a surrogate model, the methodology usually takes a form of an iterative procedure. For each iteration a set of training data is obtained (or updated), and a model is established from the training data. A search towards the optimization target is then performed using this model. This procedure continues until a particular criterion is satisfied. By performing the optimisation with a surrogate model as opposed to directly using the original complex model, the number of time-consuming evaluations of the complex model is expected to be reduced, hence improving the computational efficiency of the optimisation.

This type of optimisation method can be further classified into *global* and *local* approaches, according to the region in the search domain where the model is established and where the search is conducted at one step in the optimization procedure. A *global method* does not confine the modelling and search steps to a specific sub-region within the entire search domain. After initialization, in which an initial set of data are obtained and a model subsequently established, the location of new point(s) to be explored in the next step is determined via a particular merit function. This function is chosen such that it presents a balance between exploitation and exploration: the former emphasizes the importance of the region in which the optimal point is found by using the surrogate model, while the latter tends to direct the search to the region which has not been sufficiently explored so far, hence increasing the chance of locating the global optimum. Different formulations of such a merit function have been proposed in the literature, see e.g. Torczon & Trosset (1998) and Jones, Schonlau & Welch (1998). Based on the argument that, for complex problems, no single surrogate model can be properly established for the entire domain, *local methods* have been proposed which restrict the modelling and search to within a specific region at each step through an iterative solution process. In addition to the use of a merit function to balance exploitation and exploration, as in the global methods, an additional key issue to be addressed by a local method is the determination of the valid region for executing the local modelling and local search; different strategies have been proposed, see e.g. Conn et al. (1997), Stander (2000), and Bueche et al. (2005).

It is interesting to note, that to date the techniques developed for the type of problems considered here, i.e. the optimization of expensive black-box functions, have received only limited attention from the process engineering community despite the challenges stemming from complex chemical process systems. Meyer et al. (2002) mention the suitability of the method of Jones et al. (1998) for handling computationally highly expensive black-box models compared to their own method for solving optimisation problems with non-factorable constraints. Another brief mentioning of the method of Jones et al. (1998) is in Bindal et al. (2006). The only comprehensive application study known to the authors is that of Wan et al. (2005), where a supply chain management problem is successfully solved by using a simulation-based optimisation procedure which is essentially an implementation of a global method, as classified above, but which incorporates a sequential domain reduction strategy. As for the application of local approximation models, Davis and Ierapetritou (2007) use low-order polynomials to approximate microscopic models of chemical reactors being optimized. Similarly, Schaefer et al. (2005) solve a stirrer optimisation problem based on a CFD model by applying an existing implementation of an algorithm that makes use of quadratic approximation models within a trust region framework. Closely resembling the traditional response surface modelling techniques, the methods used in these two applications are essentially meant to find local optimum only, because the optimisation procedure is mainly driven by exploiting the (lower-order polynomial based) response surface model, and does not explore the less known regions in the solution space (cf. Jones, 2001).

In this paper, the studies are directed towards local surrogate model based methods. The rationale for this is the difficulty of establishing a globally valid approximate model for a complex chemical process system. This issue has been demonstrated previously in model simplification studies in process engineering, e.g. the adoption of local approximates for physical property models or CFD models as mentioned earlier. Furthermore, a methodology will be investigated in detail which addresses both the balance between exploration and exploitation in its sampling strategy and the updating of the local region for searching and modelling at each iteration. Such a method will be capable of not being trapped in (some of) the local optima due to the explorative tendency of the sampling procedure. This approach is not guaranteed to converge to the true global optima due to the use of local models. However, this is usually acceptable as pursuing the absolute global optima is not always meaningful in practical process engineering applications, considering both the high computational cost and the credibility of such an approach when inaccuracies exist in the model on which the optimisation is based on as frequently happens in reality. On the other hand, confining modelling to local regions will ensure the feasibility of building reliable surrogate models with relatively small numbers of evaluations of the complex original model. This will consequently enhance the computational efficiency of the optimisation procedure.

In the rest of the paper, a general procedure for local surrogate model based optimisation methods is first presented in Section 2. An introduction to a particular algorithm called the Gaussian Process Optimisation Procedure (GPOP) and the underlying Gaussian process model that provides the local approximation of the rigorous model in GPOP is given in Section 3. In Section 4, an application of the GPOP algorithm to a process engineering problem namely the optimisation of a crystallisation process is presented. Some concluding remarks are finally given in Section 5.

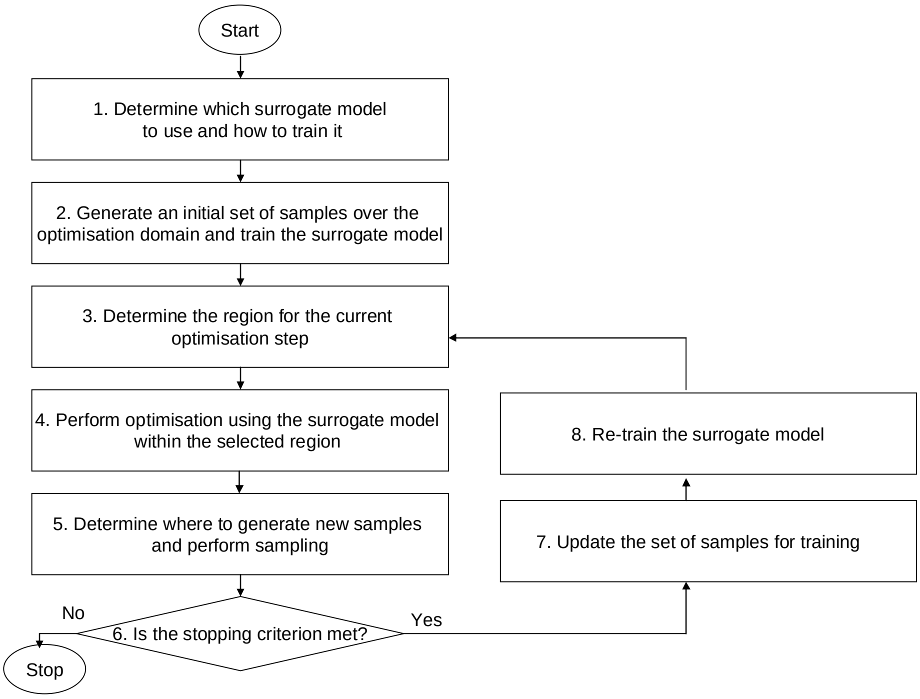
Start
1. Determine which surrogate model to use and how to train it
2. Generate an initial set of samples over the optimisation domain and train the surrogate model
3. Determine the region for the current optimisation step
4. Perform optimisation using the surrogate model within the selected region
5. Determine where to generate new samples and perform sampling
6. Is the stopping criterion met?
No
Yes
Stop
7. Update the set of samples for training
8. Re-train the surrogate model

Figure 1. General procedure for local surrogate model based optimization methods.

2. General Procedure of Local Surrogate Model-Based Optimisation Methods

A typical procedure for local surrogate model based methods for the optimisation of computationally expensive black-box objective functions is shown in Figure 1. The approach starts with the determination of the form of the surrogate model and the algorithm for training the surrogate model given a set of samples (Step 1). Following this the surrogate model is initialized by samples drawn from a number of evaluations of the original complex model (Step 2). This step is then followed by an iterative procedure that focuses on the determination of the current search region (Step 3), the optimisation (i.e. searching) within this current region (Step 4), the updating of the set of samples by incorporating the results of the new evaluations of the original model within the current region (Steps 5 and 7), and finally the re-training of the surrogate model based on the updated set of samples (Step 8). During this iterative procedure, a stopping criterion is assessed (Step 6) to determine if the procedure should continue.

While most local surrogate model based methods follow such a computational procedure, they differ in how these steps are actually carried out. Particularly, the major difference lies in the selection of the surrogate model (Step 1), the determination of the local region (Step 3), the selection of locations for a new round of sampling (Step 5), and finally the updating of the set of samples for training (Step 7).

3. Gaussian Process Models and the GPOP Algorithm

The Gaussian Process Optimization Procedure (GPOP), the algorithm studied in detail in this work, employs a Gaussian process model as the surrogate model. This is described in Section 2.1 with other features of GPOP being presented in Section 2.2.

3.1. The Gaussian process model

A Gaussian process model (Williams & Rasmussen, 1996) is applied to fulfil the task of locally approximating a function with an unknown form based on a set of samples obtained by evaluating the function. As a statistical model, it is selected because it can not only approximate arbitrary functions but can also directly provide an uncertainty measure for its predictions by means of a standard derivation. As will be shown later in Section 3.2, this uncertainty measure plays an important role in the GPOP algorithm.

The principle of a Gaussian process model is as follows. Consider N data points, $\mathbf{X}_N=\{\mathbf{x}_i\}$, i=1,2,…,N, $\mathbf{x}_i \in \mathbb{R}^n$ and the corresponding values of a function f, $\mathbf{t}_N=\{t_i\}$,i=1,2,…,N evaluated at these data points. The Gaussian process model treats $\mathbf{t}_N$ as a particular sample of a multivariate Gaussian distribution with zero mean and a covariance matrix denoted as $\mathbf{C}_N$. For a new data point $\mathbf{x}_{N+1}$, its function value $\mathbf{t}_{N+1}$ has the following conditional probability density:

$$p(t_{N+1} \mid \mathbf{X}_{N+1}, \mathbf{t}_N) \propto \exp(-\frac{1}{2}\frac{(t_{N+1}-\widehat{t}_{N+1})^2}{\sigma^2_{t_{N+1}}}) \qquad (1)$$

where the mean and the variance are given by

$$\begin{aligned} \widehat{t}_{N+1} &= \mathbf{k}^T\mathbf{C}_N^{-1}\mathbf{t}_N, \\ \sigma^2_{t_{N+1}} &= \kappa - \mathbf{k}^T\mathbf{C}_N^{-1}\mathbf{k}. \end{aligned} \qquad (2)$$

$\mathbf{C}_N$, **k**, and κ represent the covariance matrix, covariance vector, and variance, respectively. $\mathbf{C}_N$ is determined by a covariance function *C* which, when applied in the GPOP algorithm, takes the following form

$$C(\mathbf{x}_p, \mathbf{x}_q) = \theta_1 \exp(-\frac{1}{2}\sum_{i=1}^{n}\frac{(x_{p,i} - x_{q,i})^2}{r_i^2}) + \theta_2 + \delta_{p,q}\theta_3. \qquad (3)$$

The parameters in the covariance function *C*, namely θ_1, θ_2, θ_3, and r_i, i=1,2,..,n are often referred to as hyperparameters which can be estimated, for example by applying the maximum likelihood method to the known samples. The covariance vector **k** and the variance κ in Eq. (2) can be expressed as

$$\begin{aligned} k_i &= C(\mathbf{x}_i, \mathbf{x}_{N+1}), \qquad i = 1,2,...,N, \\ \kappa &= C(\mathbf{x}_{N+1}, \mathbf{x}_{N+1}). \end{aligned} \qquad (4)$$

3.2. The GPOP algorithm

The GPOP (Gaussian Process Optimisation Procedure) algorithm is one of the local surrogate model based methods for the optimisation of computationally expensive black-box functions (Bueche et al, 2005). In addition to using a Gaussian process model as the surrogate model, the other key features of the GPOP algorithm are as follows.

(i) Use of a hypercube that surrounds the current best solution as the local region for searching (Step 3). This hypercube is defined by

$$\mathbf{x}^{best} - \frac{\mathbf{d}}{2} \le \mathbf{x} \le \mathbf{x}^{best} + \frac{\mathbf{d}}{2}, \ \mathbf{x}, \mathbf{d} \in \mathbb{R}^{\mathrm{n}}$$

$$d_i = \max_c(x_{c,i}) - \min_c(x_{c,i}), \qquad i = 1,2,...,n; \qquad c = 1,2,...,N_C$$

where n is the number of decision variables, and N_C is a prefixed number that denotes the number of points closest to the current best solution that are to be included in the set of samples for training, as discussed below.

(ii) Use of a merit functions to determine where to generate new samples (Steps 4 and 5). The underlying idea is the same as that utilised in other methods for the optimisation of expensive black-box functions, that is to balance (a) the exploitation of the current GP model (valid for a particular local region) for finding the most promising candidate with respect to the objective function concerned, with (b) the exploration of less known parts of the optimisation space. Specifically, the GPOP algorithm utilises the following set of merit functions (for a minimisation problem):

$$f_m(\mathbf{x}) = \hat{t}(\mathbf{x}) - \alpha\sigma_t(\mathbf{x}), \qquad \alpha = 0,1,2,4$$

where $\hat{t}$ is the value of the objective function as predicted by the GP model, σ_t is the standard deviation of this prediction, α is the scaling factor that reflects the balance mentioned above. Four levels of α were chosen and empirically proven to be effective by the original authors of the GPOP algorithm. For each of these merit function, optimisation is performed using the GP model, and the solution is one point where a new sample is to be generated. This implies that four new samples are generated for each iteration.

(iii) Updating the set of samples for training (Step 7). Each time new samples are generated, the set of samples to be used for retraining the GP model is updated by making selections among all the samples available so far. The selection rule is such that the N_C points closest to the current best solution (in terms of Euclidian distance) and N_R, the most recently evaluated points are included, hence forming an updated training data set that includes in total $N_C + N_R$ points. The values of N_C and N_R can be problem specific but are usually proportional to the dimension (i.e. the number of decision variables) of the optimisation problem.

In addition to the above features of GPOP, for a practical application, an appropriate stopping criterion is required. In this work, the optimisation procedure is stopped when the size of the current local region, denoted by the largest diameter of the hypercube, is smaller than a preset value termed “minimum diameter”, or when no sampling points different from the existing ones are identified by the algorithm (essentially Step 5 in Figure 1).

4. Case Study: Optimisation of a Crystallisation Process

The GPOP algorithm is applied for the optimisation of a crystallisation process. The optimisation problem is described in Section 4.1, with the results presented in Section 4.2.

4.1. Problem description

The optimisation of the temperature profile of a batch cooling crystallisation process for the potash alum -water system is considered. In particular, it is assumed that a batch may involve one or more periods where the temperature increases (i.e. heating) in addition to the cooling periods. This optimisation problem is formulated as follows:

$$\begin{aligned}
&\max_{T(t)} L_a(t_{end}) + w\sum_i \Delta T_i \\
&s.t. \\
&T(0) = T_{start}, \\
&T(t_{end}) = T_{end}, \\
&T_{end} < T(t) < T_{start}, \ for \quad 0 < t < t_{end}, \\
&Y(t_{end}) \geq Y_{linear}(t_{end}),
\end{aligned}$$

where L_a is the (weight-) average product size which is essentially determined by the crystallization process model discussed below and ΔT_i is the temperature change during the *i-th* cooling period. *w* is a weighting factor that determines the degree to which a low ending temperature of a cooling period is penalized. This serves as a approximate means by which to prevent excessive temperature oscillations. *T(t)* represents the temperature profile, T_{start} and T_{end} are the pre-specified temperatures at the start point and end point of the batch respectively, *t* is time and t_{end} is the pre-specified length of the batch. *Y* is the yield of crystals and Y_{linear} is the yield attained from a linear cooling profile for the same process. To solve this problem, the temperature profile is parameterised into a piece-wise linear approximation:

$$T(t) = \begin{cases} T_0 \equiv T_{start}, & t = 0; \\ T_{i-1} + r_i(t - t_{i-1}), & (i-1)\Delta t < t \leq i\Delta t, \quad i = 1,2,\dots,N; \end{cases}$$

where r_i is the rate of change of the temperature for the *ith* piece, *N* is the total number of pieces and $\Delta t = t_{end}/N$. The parametric settings applied here are t_{end} = 11 hours; T_{start} = 40.0'C; T_{end} = 20 'C; $Y_{linear}(t_{end})$ = 0.125; *N*=11; *w*=0.25. This formulation results in 10 independent variables requiring to be optimised.

The solution of the above optimisation problem relies on a mechanistic model of the crystallisation process, which is developed on the basis of the kinetics of nucleation, (one-dimensional) crystal growth, and dissolution and the population balance of the crystals. Details of the model can be found in Yang, Montague and Martin (2006). Briefly, it is a dynamic model in a form of a set of partial differential-algebraic equations. There are two factors that make the numerical solution of the model computationally expensive. Firstly, when the process switches from positive supersaturation to negative supersaturation as a result of heating, the kinetics of nucleation and growth in the model is replaced by that of dissolution, which is also accompanied by a change in boundary conditions for the population balance equation. Such a switch incurs significant discontinuities to which the numerical integrator is often sensitive and hence slows down the solution speed. The same effect also exists when the system is switched from negative to positive supersaturation as a result of re-cooling. Secondly, the growth of crystals is modelled considering the growth rate dispersion (GRD) phenomenon, which introduces an additional dimension of distribution into the model and consequently leads to the multiplication of the size of the equation systems after discretisation, requiring even longer time to get the model solved.

4.2. Results

The mechanistic model is formulated and solved by means of gPROMS (Process Systems Enterprise Ltd, 2004). The GPOP algorithm was implemented using MATLAB. Accessing the model solution from MATLAB is implemented using go-MATLAB developed by Process Engineering Enterprise.

The optimisation was first carried out on the mechanistic model using a Genetic Algorithm (GA). A GA was selected to avoid being trapped in a local optimum too quickly. In particular, the GA implementation by Houck, Joines, and Kay (1995) has been employed with their default algorithmic parameters. The same optimisation problem was then solved by applying the GPOP algorithm, where both the maximum likelihood based training of the Gaussian process model (i.e. the surrogate model) and the optimisation using the surrogate model, at each iteration in the procedure, were carried out using the aforementioned GA programme. The algorithmic parameters for GPOP were set such that $N_C = N_R = 10*n$, where n is the number of input variables. In this case study, n=10. The stopping criterion, the "minimum diameter" that indicates the critical size of the local search region is set to be 1.0.

When applied to solve the crystallisation optimisation problem, both the GA and GPOP yield results that were sensitive to the initial set of samples selected to start the optimisation. To achieve a fair comparison, five different sets of initialisation were generated, each comprising 100 samples. These were generated by evaluating the original complex model, following the application of Latin hypercube designs provided in the MATLAB statistics toolbox to obtain uniformly distributed samples. Each set of initialisations was then used for the execution of the GA and GPOP algorithms.

Table 1. Results of comparative studies on GPOP and GA.

No. of comparative study	**Number of original complex model evaluations**		
	GA	GPOP	Savings brought by GPOP
1	3170	328	90%
2	1390	496	64%
3	1135	276	76%
4	1069	840	20%
5	4153	588	86%

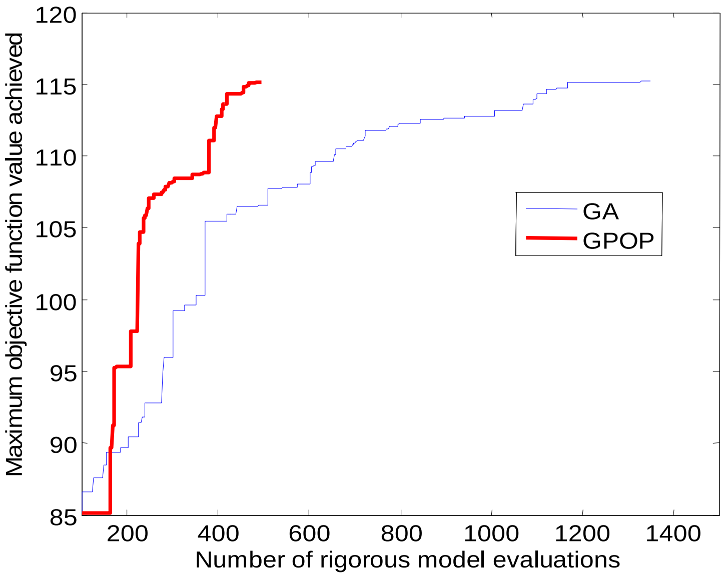

Figure 2. Comparison of the performance of GA and GPOP.

The results in terms of the number of original model evaluations (including those for initialisation) required by the GA and GPOP algorithms for achieving the same value of the objective function are shown in Table 1. It can be seen that except for Set 4 where the two algorithms require a comparable number of original model evaluations, the GPOP algorithm results in significant savings (up to 90%). The average savings attained through the implementation of the GPOP algorithm in comparison with the GA is 67%. Figure 2 shows the intermediate results for Case 2.

5. Concluding Remarks

Reducing the complexity of mathematical models is important for realising the practical application of complex models in optimisation-based process engineering tasks. In this research, a local surrogate model based method for the optimisation of computationally expensive black-box objective functions has been studied. In particular, the GPOP algorithm reported in the recent literature, which makes use of Gaussian process models, to approximate complex mechanistic mathematical models has been implemented and applied for the optimisation of a batch cooling crystallisation process. The case study has demonstrated that this algorithm can bring about significant savings in terms of the number of evaluations of the original complex model, in comparison with algorithms that make use of the original model directly without employing a surrogate model, such as the GA as used in the case study.

The optimisation problem tackled in the case study was not just a test case but rather a real problem in the area of the modelling and optimisation of crystallisation processes. Of the various surrogate model-based optimisation methods reported in the literature, the GPOP algorithm was selected because conceptually this algorithm offers a desirable trade-off between locating global optima and ensuring computational efficiency as discussed in Section 1. However, a detailed comparison between this and other methods by means of comprehensive test cases is still ongoing. Besides, the mechanistic model in this case study is deterministic, hence the simulation results based on this model are not noisy. In future work, it would be interesting to investigate how an algorithm like GPOP can be applied (possibly with adaptation) to address noisy simulation based optimisation.

Acknowledgements

This work is financially supported by the EPSRC grant GR/R64407/01 “Vertical Integration of Product Development and Manufacturing”.

References

Bezzo, F., Macchietto, S., and Pantelides, C. C. (2005). *AIChE Journal*, 51, 1169-1177.

Bindal, A., Ierapetritou M.G., Balakrishnan, S., Armaou, B, Makeev, A. G., Kevrekidis, I.G. (2006). *Chemical Engineering Science*, 61, 779 – 793.

Bueche, D., Schraudolph, N.N., Koumoutsakos, P. (2005). *IEEE Transactions on Systems, Man, and Cybernetics,* 35, 183 – 194.

Chimowitz, E. H., Anderson, T. F. , Macchietto, S. and Stutzman, L. F. (1983). *Ind. Eng. Chem. Proc. Des. Dev.*, 22, 217–225.

Conn, A. R., Scheinberg, K., Toint. P. L. (1997). *Mathematical Programming*, 79, 397–414.

Davis, E., Ierapetritou, M. (2007). *Comput. Chem. Eng.*, 31, 466–476.

Houck, C., Joines, J., Kay, M. (1995). *A Genetic Algorithm for Function Optimization: A Matlab Implementation.* NCSU-IE TR 95-09.

Jones, D. R. (2001). *Journal of Global Optimization*, 21, 345-383.

Jones, D. R., Schonlau, M., Welch, W. J. (1998). *Journal of Global Optimization*, 13, 455-492.

Leesley, M. E. and Heyen G. (1977). *Comput. Chem. Eng.,* 1, 109–112.

Marquardt, W. (2002). *AIChE Symp. Ser. 326,* 98, 12–42.

Meyer, C.A., Floudas, C.A., Neumaier, A. (2002). *Ind. Eng. Chem. Res.* 41, 6413-6424.

Process Systems Enterprise Ltd. *gPROMS Advanced User Guide*, Process Systems Enterprise ltd., 2004.

Schaefer, M., Karasoezen, B., Uludag, Y., Yapıcı, K., Ugur, O. (2005). *Computers and Chemical Engineering* 30, 183–190.

Stander, N. (2000), *Optimization of Nonlinear Dynamic Problems using Successive Linear Approximations,* AIAA Paper 2000-4798.

Torczon, V. and Trosset, M.W. (1998). Using approximations to accelerate engineering design optimization. In*: the Proceedings of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, St. Louis, Missouri, September 2-4, 1998

Wan, X., Pekny, J.F., Reklaitis, G.V. (2005). *Comput. Chem. Eng.*, 29, 1317–1328.

Williams, C. K. I., Rasmussen, C. E. (1996). In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo (Eds.), *Advances in Neural Information Processing Systems 8*, 514-520. The MIT Press, Cambridge, MA.

Yang, A., Montague, G., Martin, E. (2006). *Temperature cycling in batch cooling crystallisation: a model-based study*. Technical Report, School of Chemical Engineering and Advanced Materials, Newcastle University, UK.