AN ALGORITHM FOR THE USE OF SURROGATE MODELS IN MODULAR FLOWSHEET OPTIMIZATION.

José A. Caballero

Dept. Chemical Engineering, University of Alicante, E-03080, Alicante, Spain

Ignacio E. Grossmann

Dept. Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213

Abstract

In this work a methodology is presented for the rigorous optimization of nonlinear programming problems in which the objective function and (or) some constraints are represented by noisy implicit black box functions. The special application considered is the optimization of modular process simulators in which the derivatives are not available and some units operations introduce noise preventing the calculation of accurate derivatives. The black box modules are substituted by metamodels based on a kriging interpolation that assumes that the errors are not independent but a function of the independent variables. A Kriging metamodel uses a non Euclidean measure of distance that avoid sensitivity to the units of measure. It includes adjustable parameters that weight the importance of each variable getting a good model representation, and it allows to calculate errors that can be used to establish stopping criteria and provide a solid base to deal with 'possible infeasibility' due to inaccuracies in the metamodel representation of objective function and constraints. The algorithm continues with a refining stage and successive bound contraction in the domain of independent variables with or without kriging recalibration until an acceptable accuracy in the kriging metamodel is obtained. The procedure is illustrated with several examples.

Submitted to be considered for its publication in AIChE Journal

Introduction

There is a growing interest in designing and optimizing process and products using complex mathematical models. Computer models facilitate the exploration of alternatives. However, although hardware and software are continuously increasing in computational power, there is also a growing demand in models with increased accuracy of the mathematical description. Due to the complexity of these computer models, a large number of specialized software has appeared to solve a wide variety of problems. The architecture of most of these programs is modular in order to use tailored numerical methods developed or adapted to each particular problem. In most situations, the user only can view a 'black box model' with limited access to the original code.

Using optimization algorithms with these "black box models" is a challenging problem for at least two reasons. First, some of those models can require significant CPU computation time (e.g. a computer fluid dynamics model could take several hours of CPU time). Second, even in the case in which the CPU time is not excessive, derivatives for gradient based algorithms cannot be accurately estimated because most of these black box models introduce noise. This noise can be due to different reasons: small sensitivity of some variables, termination criteria in the algorithms or even rounding of some values in the final solution of the original model (i.e. rounding the diameter of a column to physically meaningful values).

In cases in which it is not practical to calculate the model at each iteration of an optimization algorithm, a good approach may be to use the original model as a source of 'computational experiments' that produce data points in the same way as if we had performed a physical experiment. With these data we can use a simpler model that involves explicit functions. These new models are referred as surrogate, reduced order or metamodels [1].

The Response Surface Methodology (RSM) to approximate functions has a long history at least in three areas: Geology, Global Optimization, and Statistics [2]. In Geology, the approach is called 'kriging' and is based on the pioneering works by G.G. Krige[3] who was the first in using functions to approximate the underground concentration of valuable mineral using a statistical methodology. His work formed the base of a new entire field now known as geostatistics[4, 5]. In global optimization this approach is called 'Bayesian Optimization'. The first important contribution was due to H. Kushner[6], and developed by many authors[7-10] In statistics, the initial interest relied on approximating difficult integrals or other difficult functions.

At the end of the 1980s, a new research interest appeared in computer science and statistics for applying kriging techniques to deterministic computer experiments[11, 12]. This new methodology was named Design and Analysis of Computer Experiments (DACE) following the title of the paper that first introduced it. More than a specific algorithm, DACE refers to a set of tools for modeling and optimizing a complex system. The idea, as mentioned above, is very intuitive. Data obtained from computer experiments generated from complex models are fitted using some metamodel, and at the same time statistical parameters are generated that allow determining how good the metamodel is. There is a secondary benefit, because it is possible to select those variables that most influence the behaviour of the system. The new estimations or optimization of the system is performed using the surrogate model instead of the original one. This methodology is especially good at modelling the non-linear function often appearing in engineering[13].

RSM methods can be differentiated in two ways: if they are non-interpolating (i.e. least squared error of some predetermined functional form) or interpolating (pass through all points). Jones[13] showed that non interpolating surfaces, such as quadratic surfaces, can be unreliable because they do not capture the shape of the function. He showed that it is usually better to use surfaces that interpolate the data with linear combinations of 'basis functions'. Although the work of Jones mainly focused in global optimization, he showed that the interpolating approach is much more reliable showing some examples in which quadratic fitting could not even locate a local minimum.

With interpolating methods it is possible to differentiate between fixed basis functions (i.e. linear, cubic or thin-plate splins) and basis functions with adjustable parameters (kriging). Furthermore, kriging has a statistical interpretation that allows the construction of estimations or the error in the interpolator, which can be crucial in the development of an accurate optimization algorithm. Due to these adjustable parameters kriging interpolation tends to produce the better results[13, 14].

In this work, we develop an algorithm based on fitting response surfaces –using a kriging metamodel- for the optimization of constrained-noise black box models. The system we are dealing with have some characteristics intermediate between classical Bayesian Optimization, pure DACE systems and very noisy geological models. The functions can be considered deterministic, but they introduce some noise. Here we are dealing with mixed systems in which a part can be represented by a metamodel and the rest with an accurate model. Besides, an important characteristic is that we deal with constrained problems in which the metamodel can represent either the objective function or some constraints (or both simultaneously). A typical case is the optimization of process flowsheets using modular simulators in which some units are represented by a metamodel. In these systems it is possible to include external constraints and even the result of some calculations could be constraints to the model.

In the rest of the paper we first present an overview of kriging. Then an algorithm based on successive region refinement is introduced. Later we will show the special characteristics of process flowsheets as a typical example of constrained light-noise black box models, with some examples to illustrate the effect of noise in these systems, and the main considerations to take into account when kriging is used.

Overview of kriging interpolation.

When we evaluate a deterministic function in a set of given points we assume that the true function y(x) is approximated by a function *f(x)* with some error;

$$y(x) = f(x) + \varepsilon \qquad (1)$$

Most metamodel techniques assume that the errors (ε) are independent and identically distributed (with a normal distribution) $\varepsilon \sim N(0,\sigma^2)\forall x$. However, the errors in the predicted values are usually not independent, but they are a function of *x*. In other words, it is expected that if the predicted value $f(x_i)$ is far from the true value $y(x_i)$ then the predicted value of a point near x_i, $(x_i + h)$ will also have a predicted value $f(x_i + h)$ far from the true value $y(x_i + h)$.

The kriging fitting approach is comprised of two parts; a polynomial term and a departure from that polynomial:

$$y(x) = f(x) + Z(x) \tag{2}$$

where Z is a stochastic Gaussian process, that represent the uncertainty about the mean of $y(x)$ with expected value zero $E\left(Z(x)\right)=0$ and covariance for two points $\mathbf{x}_i, \mathbf{x}_j$: $\mathrm{cov}\left(Z(\mathbf{x}_i), Z(\mathbf{x}_j)\right)=\sigma^2\mathbf{R}(\mathbf{x}_i, \mathbf{x}_j)$. Here σ^2 is a scale factor known as process variance that can be tuned to the data and $\mathbf{R}(\mathbf{x}_i, \mathbf{x}_j)$ is the spatial correlation function (SCF). The choice of SCF determines how the model fits the data. There are many choices for the SCF, but the most common used in kriging models is the exponential function:

$$\mathbf{R}\left(\mathbf{x}_i, \mathbf{x}_j\right) = \exp\left(-\sum_{l=1}^{d} \theta_l \left|\mathbf{x}_{i,l} - \mathbf{x}_{j,l}\right|^{P_l} \right) \qquad \theta_l \geq 0; \quad 0 \leq P_l \leq 2 \tag{3}$$

This SCF has the property that if $\mathbf{x}_i = \mathbf{x}_j$, then the correlation is one. The correlation tends to zero as the difference between both points increases. In other words, the influence of the sampled data point on the point to be predicted becomes weaker as their distance increases. The value of θ_l indicates how fast the correlation goes to zero as we move in a l^{th} coordinate direction. Parameters P_l determine the smoothness of the function. Therefore, in engineering smooth functions this parameter is usually fixed to 2 in all coordinate directions.

A polynomial choice for equation 2 has the form:

$$f(x) = \mu + \mu_1 x + \mu_2 x^2 \tag{4}$$

In kriging fitting, when a function is smooth, the degree of the polynomial *f(x)* does not affect significantly the resulting metamodel fit because *Z(x)* captures the most significant behaviour of the function[1]. This is an important advantage of kriging models. Usually a simple constant term (μ) is enough for a good prediction.

To estimate the values of $\mu, \sigma^2, \theta_l, P_l$, we maximize the likelihood of the observed data **y**.

$$L(x, \sigma, \mu) = \frac{1}{\left(2\pi\sigma^2\right)^{n/2} \left|\mathbf{R}\right|^{1/2}} \exp\left(\frac{-(\mathbf{y} - \mathbf{1}\mu)^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{1}\mu)}{2\sigma^2} \right) \tag{5}$$

In equation 5 **y** is the nx1 vector of observed responses, **1** is an nx1 vector of ones (a vector of unit elements) and n is the number of sampled points.

From a practical point of view, it is more convenient to maximize the logarithm of the likelihood function

$$-\frac{n}{2} ln(2\pi) - \frac{n}{2} ln(\sigma^2) - \frac{1}{2} Ln\left(\left|\mathbf{R}\right|\right) - \frac{-(\mathbf{y} - \mathbf{1}\mu)^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{1}\mu)}{2\sigma^2} \tag{6}$$

Differentiating equation 6 with respect to σ^2 and μ and equating it to zero, and after some algebra we get the optimal values for μ and σ^2:

$$\hat{\mu} = \frac{\mathbf{1}^T \mathbf{R}^{-1} \mathbf{y}}{\mathbf{1}^T \mathbf{R}^{-1} \mathbf{1}} \tag{7}$$

$$\hat{\sigma}^2 = \frac{(\mathbf{y} - \mathbf{1}\hat{\mu})^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{1}\hat{\mu})}{n} \tag{8}$$

Substituting equations (7) and (8) in (6) and neglecting the constant terms, the maximization of the concentrated log-likelihood function is given by:

$$\max_{\theta_l, P_l} \; -\frac{n}{2}\ln(\hat{\sigma}^2) - \frac{1}{2}\ln\left(|\mathbf{R}|\right) \tag{9}$$

To interpolate a new point x^{new} we add the point $\left(x^{new}, y^{new}\right)$ to the data and compute the augmented likelihood function keeping all the parameters at the previous calculated values. With all the parameters constant, the log-likelihood function is only a function of y^{new}. Therefore, The predicted value for y^{new} will be the value that maximizes the augmented likelihood function. The final predictor of the kriging method is given by equation 10. A detailed derivation can be found in Sasena[15]:

$$\hat{y}\left(x^{new}\right) = \hat{\mu} + \mathbf{r}^T \mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{1}\hat{\mu}\right) \tag{10}$$

where **r** is the nx1 vector of correlations $R(x^{new}, x_i)$ between the point to be correlated and the sample design points.

We are more confident in the prediction if the new point is near a sampled point –the error drops to zero in all sampled points-, and at the same time we are more confident if the augmented log-likelihood drops off rapidly as one moves away from the optimal value of y^{new}. It is possible to derive the mean-squared error of the predictor, which yields:

$$s^2(x^{new}) = \hat{\sigma}^2\left(1 - \mathbf{r}^T \mathbf{R}^{-1}\mathbf{r} + \frac{(1 - \mathbf{r}^T \mathbf{R}^{-1}\mathbf{r})^2}{\mathbf{1}^T \mathbf{R}^{-1}\mathbf{1}}\right) \tag{11}$$

There are two important differences between kriging and other basis function methods that make the kriging to usually outperform those methods. First, the other methods usually do not have parameters in their basis functions, or if there is any parameter, it is rarely optimized. Second, most of the methods use a Euclidean norm which makes them sensitive to the units of measurements. Kriging, however, capture all those effects in the θ's parameters through a non-Euclidean norm.

As an example of the quality of kriging interpolation we use the 'peaks' function (a sample function in Matlab©[16]). Figures 1and 2 shows a graphical comparison using the 33 points market

as dots. The fitted surface using only those 33 points is very close to the original one (at simple sight it is difficult to differentiate them).

In highly noisy systems a non-interpolated approach is usually preferred in order to avoid oscillations of the fitted surface when it is forced to go through all data points. To allow kriging to smooth the data, an additional parameter is introduced in the SCF (equation 3). The kriging model takes the form[15],

$$y(x) = f(x) + Z(x) + E(x)$$

$$\mathbf{R}\left(\mathbf{x}_i, \mathbf{x}_j\right) = \prod_l Nu_l \exp\left(-\theta_l \left|\mathbf{x}_{i,l} - \mathbf{x}_{j,l}\right|^{P_l}\right) \qquad (12)$$

$$\theta_l \geq 0; \quad 0 \leq P_l \leq 2 \quad 0 \leq Nu_l \leq 1$$

with a new adjustable parameter, Nu, that can be defined globally (a single parameter for all dimensions) or locally to smooth differently in each coordinate direction. An example of non-interpolating kriging model is shown in Figure 3.

Kriging Metamodels in Modular Chemical Process Simulators

Modular Chemical Process Simulators are widely used tools used by chemical process industries. This is because modern chemical process simulators include state of the art models for each of the unit operations. However, if we try to use these models to optimize a a process there are some important difficulties:

1. The most efficient optimizers are 'gradient based' -Most of them are based on more or less sophisticated versions or Newton method, SQP, GRG[17] -. If the simulator works like a 'black box' model –that is the most common situation-, automatic differentiation techniques cannot be used to accurately estimate derivatives and the usual approach consists of using finite difference approximations. Unfortunately, the 'black box' systems introduce some noise. The source of that noise can be a lack of sensitivity of some variables with respect to the design variables together with the stopping criteria in the intermediate calculations inside the modules. In some cases, truncation and rounding of decimals to provide the final user with physically meaningful values, can complicate even more the situation. In this case a detailed sensitivity analysis must be performed in order to determine the optimal perturbation length of each variable and the adequate approximation procedure[1, 18].

2. Even though the process simulators are very robust, it is not uncommon that some unit operations (i.e. distillation columns) do not easily converge for a set of design variables –assuming that there is a feasible solution for that set of specifications-. This problem can usually be solved by modifying initial guesses or by successive approximations following, for example, a continuation method. In gradient based algorithms steps are usually not too large which contributes to mitigate this problem.

3. Direct search algorithms –derivative free- do not have the problems mentioned in previous points (in general it is not difficult to modify direct search algorithms to circumvent the lack of convergence in a given point). But these algorithms have two drawbacks: 1. The number of function evaluations tends to be large, which is important if the CPU time is significant. 2. They are usually designed to solve unconstrained

problems. Several approaches have been developed to deal with constrains. Penalty methods in which the constrains are moved to the objective function and transformed in an unconstrained problem, multiobjective approaches, variable reduction, removing some variables by explicit or implicit elimination in equality constraints, restoration methods in which if a point violates some constraints there is a especial restoration step, special codification for constraints (i.e. genetic algorithms with especial codifications), etc. (Michalewicz[19, 20] has published a review of these techniques). Although, some of these methods may show reasonable performance, in general they require a large number of function evaluations that greatly increases in constrained problems, and most of them also include parameters that must be tuned for each particular problem.

Due to these difficulties some chemical process simulators have developed equation based versions, that have proved to be very useful in optimization, but often at the price of losing the robustness of the numerical methods especially developed for some unit operations. In general, as the complexity of the mathematical model increases the modular architecture tends to be more robust.

In order to develop a reliable metamodel optimization algorithm for process simulators we must take into account the following facts:

1. In general the CPU time per run is not too large (in comparison with CFD models that could take hours per run), but considerably longer than a simple function and constraints evaluation in an equation based approach. Hence, we can perform a moderate number of flowsheet evaluations without increasing too much the CPU time in the optimization.

2. Flowsheets introduce noise in some situations that is not negligible and must be taken into account.

3. In general, the lower the dimensionality of the metamodel, the higher its accuracy. A good approach is to use a metamodel for each unit operation (or reduced group of unit operations) to get metamodels with low dimensionality. Furthermore, if there are some units that do not generate noise and can be evaluated very quickly, a metamodel is not needed in that unit and the optimization can be performed using a mixed approach (i.e. in HYSYS.Plant©[21] a vapour-liquid flash almost does not introduce noise with the default parameters).

An algorithm for constrained optimization using kriging metamodels

Taking in mind all the considerations in the previous sections, it is possible to develop an algorithm for optimizing a flowsheet using a kriging metamodel approach that guarantees a local minimum within a pre-specified tolerance. The next paragraphs provide a comprehensive description of the main steps with comments about cautions, weaknesses and robustness of each of them. Figure 4 shows a block diagram of the algorithm. While the algorithm is described for optimizing process flowsheets, it can be used for any black box model with or without noise.

1. Given a flowsheet to optimize, identify the unit operations that introduce noise or are very CPU time consuming in their simulation. These will be the units to be substituted by a metamodel, in our case a kriging model. The rest can be considered as accurate implicit models without any special treatment. If possible, group units to keep the dimensionality

(independent variables) as low as possible. One should take into account that problems with a large number of independent variables could require an important number of sample points and the fitting surface step could become the limiting step – usually no more that 9-10 dimensions-. Besides, the fitted surface in large dimensionality problems could be very inaccurate. Bounding the variables as much as possible, will increase the accuracy of the metamodel.

2. It is important to have at least an estimation of the noise introduced by the dependent variables (especially if the noise introduced is important). In flowsheet optimization, a simple way of doing that consists of simulating the flowsheet for a fixed set of independent variables, starting from different initial points. Note that the accuracy of the final optimum point cannot be higher than the noise, but should be as close as possible.

3. Select the domain of the independent variables for the sampling, and select an initial confidence domain for those independent variables. Here there are two options: a) select a hypercube than coincides with the limits of the independent variables (initially we select the full dominion space), or b) select a sub-region included in the original dominion space. The first approach is better if the initial kriging captures all the basic trends in the model, and it has the advantage that it reaches the optimum faster because in the first iteration it can locate a near optimal region. The second approach is adequate if the dimensionality of the model is large or the kriging cannot capture the main trends in the model (i.e. large predicted errors in kriging). The major drawback with this second approach is that the first movements lie in the limits of the hypercube that must be updated until a near optimal region is located, and the algorithm enters in the refinement stage (In next points a detailed explanation of this case is included).

   Whatever the initial sampling region selected, a key point is the sampling procedure. Two aspects must be taken into account: 1. The sampling procedure, including the distribution of points, and 2. The number of points. It is not the objective of this paper to provide a detailed description of sampling techniques, but to highlight the influence of a correct sampling over the final quality of the kriging.

   A correct sample must cover all the space defined by the domain of the independent variables, However, simple Monte Carlo methods can result in large error bounds (confidence intervals) and variance. Variance reduction techniques are statistical procedures designed to reduce the variance in the Monte Carlo estimates[22]. Latin Hypercube sampling[23, 24], Hammersley[25], Halton o Sobol[26] sequences or infill sample procedures (ISP) are examples of variance reduction techniques. In most applications, the actual relationship between successive points in a sample has no physical significance. Hence the randomness for approximating a distribution is not critical[22]. Moreover, the error of approximating a distribution by a finite sample depends more on the uniformity of the distribution than on its randomness. Therefore, the sampling must be done to preserve the uniformity of the distribution and avoid 'correlation'. Methods based on Halton, Niederreiter, or similar sequences tend to produce good results for systems in which there is no further information. But all these series are generated based on prime numbers that must be carefully chosen in order to avoid correlation[22]. Infill sampling criteria, i.e. select a set of points that maximize the minimum distance between them, assures a good distribution – the 33 points in Figure 1 were generated using this approach-. However, some care must be taken in large dimensional problems because most of the points tend to be placed in the edges of the hypercube (to maximize the distance). Also, in slightly noisy systems the sample points must be chosen separated enough to assure a good kriging model (in other case a non-interpolate approach must be used). Recall that one of the motivations to use a

surrogate model is that accurate derivatives cannot be obtained in the original one, but this does not mean that the sampled value was not accurate enough to get a good metamodel.

The number of sampling points is also important. If in one dimension we use N points to get the same 'fill in' in two dimensions we would need N^2 points and in general N^k points in a k^{th} dimensional space. We cannot increase the number of points for two reasons: the time to run the flowsheet for all the points can be very large, and surface fitting could be a cumbersome problem mainly due to the NxN matrix inversion at each iteration of the algorithm. Fortunately, the smaller the distance between the bounds of the hypercube the higher the accuracy of the metamodel. Therefore, in the worst case, initially only a metamodel that captures the main trends of the model is needed and successive refinement (contraction and movement of the hypercube) allows us to get a solution as accurate as desired.

4. Fit all the surfaces using kriging and validate the model. Once all the variables (surfaces) have been estimated by kriging, it is important to validate the metamodel. In the 'peaks' function we assessed the validity of the kriging by comparing the contour plot of the matemodel with the true contours. A possible, more practical, validation consists of selecting a few additional points and a 'test' sample, and compare actual and predicted values on this small sample. But there is a better procedure called 'cross validation' that allows us to asses the accuracy of the model without extra sampling. The idea in cross validation is to leave out one observation and predict it back using the n-1 remaining points[2]. If the number of points is not too small and there are not too many outliers, a single point has not too much influence in the kriging and then it is not needed to re-estimate the kriging parameters. This procedure is repeated for all the points. In addition to the cross validated prediction of point $\mathbf{x}^i$, we also get a cross validated standard error of the prediction. These two values can be used to compute a confidence interval using the mean prediction plus or minus three standard errors (approximately 99.7 % confident that the interpolated point lies in that interval). However, instead of actually deriving???? confidence intervals, it is more convenient to compute the number of standard errors that the actual value is above or below the predicted value. If the model is valid, the value should be in the interval [-3,+3].

5. Perform the optimization of the flowsheet substituting the selected units by their metamodel. If in the constraints no variable appears from the metamodel then no special consideration is needed. However, if we are dealing with a constrained problem like the next one in which the constraints are calculated through a metamodel,

$$\begin{aligned} &\min: \ f(x) \\ &s.t. \ g(x) \pm \varepsilon_1(x) \le 0 \\ &\quad h(x) \pm \varepsilon_2(x) = 0 \end{aligned} \tag{13}$$

In equation 13 the error introduced by the metamodel can produce infeasible solutions even if the actual model is feasible. There are at least two ways for dealing with that problem. The first one consists of explicitly introducing the error in the model formulation:

$$\begin{aligned} &\min: \ f(x) \\ &s.t. \ g(x) - \varepsilon_1(x) \le 0 \\ &\quad h(x) - \varepsilon_2(x) \le 0 \\ &\quad h(x) + \varepsilon_2(x) \ge 0 \end{aligned} \tag{14}$$

The major drawback of the formulation in (14) is that it tends to underestimate the value of the objective function. The reason is that the error is usually a bound of the worst possible situation, but in most of the evaluated points this is not the case. (i.e. if the error is estimated as 3σ -99.7% confidence- we are overestimating the error in the 99.7% of the evaluated points).

Another alternative consists of removing the errors from model formulation in equation 13. If a problem is found to be infeasible we can follow two approaches: 1. solve a problem like that in equation 14 or 2. Simply consider that the problem is 'possibly feasible' if the infeasibilities for each equation are inside the errors estimated by the kriging metamodel. Note, however, that in this second case it is necessary to verify the feasibility (i.e. increasing the accuracy of kriging by adding new points or reducing the sampling hypercube).

As we refine the model by successive reducing the size of the sampled hypercube, the accuracy of the surrogate model increases and we can approach the actual solution.

6. Refinement without updating kriging parameters. It is possible to improve the solution obtained in point 5 by adding this new point to the kriging model. Like in the cross validation, a new point has little influence over the kriging parameters. Therefore, it is not needed to re-calculate them. With the new point added we reoptimize the model. This re-optimization is expected to be very fast because the initial point should be near the optimal solution. The procedure is repeated until two successive results are inside a pre-specified tolerance. Note, however, that the new points added must be separated enough to avoid 'ill conditioning' in the correlation matrix.

   It is also possible to add a large number of new points to the kriging without re-optimizing the parameters. Since kriging is an interpolating procedure, all the new points are exact points (zero error) and it is possible to further reduce the error in the interpolation. The cost is that we have to sample in the new points and invert an NxN matrix (N is the number of sampled points) and deal with this large matrix each time we want to interpolate new data which can considerably slow the optimization.

7. As will be mentioned in the section 'algorithm convergence,' the refinement step presented in point 6, even reoptimizing the kriging parameters each time we add a new point, does not necessarily guarantee convergence to a local optimum. Jones et al,[13] presents an example of how this procedure can fail in a one dimensional problem. It is also necessary to force the gradient of the surface to match the gradient of the true function, for example adding new sample points in the neighborhood of the tentative solution. Furthermore, the termination criterion based on no improvement in two consecutive iterations does not guarantee even a local optima of the true model. Therefore, we propose to reevaluate the kriging in successive contraction or moving steps:

   Depending on where the solution of the previous step is located, we take different actions that redefine the domain of independent variables. Figure 5 helps to clarify the actions taken:

   - If the optimal solution obtained in steps 5 (or 6) is an internal point to the original hypercube, then select a contraction factor and reduce the size of hypercube. This new hypercube is center on the optimal solution of previous point. (Figure 5a)

- If the optimal solution is in the limit of the hypercube, then do not reduce the size of the hypercube, simply move it to do that this last point be the center of the hypercube. (Figure 5b)

- The limits of the hypercube can go further than the bounds of the variables, although the sampling is always performed inside those bounds. Therefore a contraction step is only performed if the last point is in the boundary of the hypercube, but not if it is only in the limit of the domain of a variable. Figure 5c and 5d will help to clarify this point.

Go back to point 4 adding to the new sample the optimal point obtained in step 6.

In each contraction step a new set of sample points is generated. Previous samples are discarded. Recall that we are looking for a local optimum, and points far away from this optimum do not provide valuable information and considerably complicate the re-optimization of kriging parameters. However, to obtain sampling points near the optimum, and then decreasing the error, the best point in previous iteration is included as a sampled point in the new one.

The procedure stops when the size of the search region (last hypercube) is small enough to be confident in the final result. Note that this does not necessarily mean a large number of contraction steps. i.e. if we use the metamodel for the' peaks' function showed in Figure 1, we do not need any contraction step. Eventually if in two consecutive iterations we get the same values of the independent variables with no improvement, we cannot guarantee an optimum in the actual model if the predicted errors in the surrogate model are greater than the tolerance. But if this situation occurs it is possible to force a large contraction to verify if this is a local optimum or not.

If the number of functions defined by black box relationships is large, the kriging generation can become the limiting step. The CPU time will depend on the number of independent variables in each kriging and on the number of sampled points. However, the time to generate the kriging models increases linearly with the number of black box functions (the time to calibrate two kriging metamodels that depend on the same number of independent variables and sampled points is similar). Therefore, in systems in which the sampling is very time consuming, the kriging calibration will have no important effect. If the sampling is relatively fast, the kriging calibration will be the most time consuming step, but for a moderate number of independent variables the time is not too large and we obtain robust models.

Convergence of the Algorithm

Once a kriging metamodel is generated, the simple substitution of this model by the actual one does not necessarily guarantee a local optimumFurthermore, a refinement procedure (as described in point 6 of previous section) that consist of solving the problem and successively update the kriging with the last obtained solution until there is no improvement in two consecutive iterations, does not guarantee a local optimum. As was mentioned, Jones et al,[13] presented an example of how this procedure can fail in a one dimensional problem.

Biegler et al.[27] proved that a necessary condition for an appropriate simplified model for optimization is that the gradients of the simplified and rigorous models be the same at the optimum. This, however, implies nothing about convergence to KKT points inherent in the simple model that may be absent in the rigorous model. Biegler et al.[27] also proved that a sufficient condition for an appropriate simplified model is that it matches the gradients of the rigorous model at all points. Therefore, in order to ensure a local optimum it is not enough to

add a new point but to force the gradient of the surface to match the gradient of the true function.

Local convergence can be guaranteed, however, if the gradient matching is combined with a trust region approach. Alexandrov et al[28], developed an algorithm that uses a correction factor to force gradient matching between the surface and function at the current iterate, in the next iteration the surface is optimized within a trust region around the last point. Basically they substituted the second order Taylor approximation by a response surface within the trust region. Because the response surface is usually more accurate than the Taylor approximation larger steps are taken to the optimal solution. They proved that this approach converges to a stationary point of the function.

If the generation of the response surface is fast, the Alexandrov´s algorithm would be perfect for our flowsheet optimization using a kriging metamodel. But this is only the case if the dimensionality is not large (3 or 4 dimensions at most) and the number of points is relatively small. Therefore, we have proposed an approach that starts from a global approach (although maybe not be very accurate) and successive contraction and movement the searching region. The final result is a region in which we are confident enough in the metamodel and the gradient information obtained from it, and therefore we can guarantee that the local optimum of the metamodel coincides with the local optimum of the true function. The last step of the proposed algorithm is equivalent to a step in the Alexandrov's approach.

In noisy systems it is not possible to verify if the gradient of the metamodel matches the gradient of the true function. In this case the stopping criteria is based on the assumption that if in two successive major iterations (at least one contraction must be performed) the optimal solution is the same, we would expect that the gradients also match the 'true gradients'. This is only a heuristic based on the observation that as the domain reduces, the accuracy of the kriging increases, and also the accuracy of the derivative information extracted from the kriging. However, this is only a heuristic –two iterations could eventually produce the same gradients and to be different from the actual ones-. Fortunately, if the statistical parameters obtained from the kriging predict a good metamodel, previous situation is not likely to be produced.

Davis and Ierapetritou[29] recently proposed an algorithm in which kriging is only used in the first step, as a global picture of the system, and then a non-interpolating approach combined with SQP algorithm is used to refine the search (although they constrained to single implicit function). This approach has proved to be robust, but it lacks the nice statistical predictions of the kriging that can be used as an accurate stopping criteria. Besides, the sampling is constrained to feasible points, which can be problematic in highly constrained problems. Instead in our algorithm we reevaluate the kriging approach in successive contraction or moving steps. We maintain a robust –statistical meaning- approach, which is very useful for dealing with feasibility in constraints and as stopping criteria, at the extra cost of the kriging re-evaluation.

The kriging model is able of reproduce the actual model, and their gradients, not only in the optimum, but also in the domain of the last hypercube generated after successive contractions. Therefore, the sufficient condition mentioned above can also be ensured at least in a confidence region around the optimum.

Example 1.

This example is presented to illustrate the performance of the algorithm for a function with 10 variables (in the limit of the recommended number of variables) in which we can control the noise to evaluate the performance of the model. Originally the model was in equation[30] form. In

order to simulate the behavior of a noise black-box system the objective function and the fourth constrain that is active in the solution were substituted by implicit equations with noise artificially introduced. The model is as follows:

$$
\begin{aligned}
&\min\ f(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}) \\
&s.t.\ -105 + 4x_1 + 5x_2 - 3x_7 + 9x_8 \le 0 \\
&\quad 10x_1 - 8x_2 - 17x_7 + 2x_8 \le 0 \\
&\quad -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \le 0 \\
&\quad z(x_1, x_2, x_3, x_4) \le 0 \\
&\quad 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \le 0 \\
&\quad x_1^2 + 2(x_2 - 2)^2 - 2x_1x_2 + 14x_5 - 6x_6 \le 0 \\
&\quad 0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6 - 30 \le 0 \\
&\quad -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \le 0 \\
&\qquad 0 \le x_i \le 10 \quad i = 1....10
\end{aligned}
$$

$$
\begin{aligned}
f &= x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^4 + (x_5 - 3)^2 + \\
&\quad 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45 + \mathrm{N}(0;\ 0.01) \\
z &= 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 + \mathrm{N}(0;\ 0.01)
\end{aligned}
\tag{15}
$$

Where $N(\mu, \sigma)$ is a normally distributed random variable with mean μ and standard deviation (σ).

The optimal solution if the noise is removed is f = 24.1384; x^T = (2.1811, 2.3676, 8.8263, 5.3579, 0.9916, 1.4304, 1.3245, 9.8234, 8.2900, 8.3680)

In general, it would be necessary to perform an analysis to determine the noise, if it is independent or not of the variables, and which is the minimum perturbation in a variable that is significant. Of course, we know that information a priori, but some Monte Carlo experiments would verify that the errors are independent, with a standard deviation around 0.01 in both cases (f and z), that differences around 0.01 are needed to get significant differences in both (f and z) when only one variable is modified. Therefore, the final error is expected to be larger. Also, the synergetic effect in the error of objective function and in the constraint is important, in this case both errors are independent but their interaction has higher impact that the simple sum of them. The value of the Lagrange multiplier in the original problem (without noise) is 0.305, which means that an error in the constraint of 0.01 can produce an error in the objective function around 0.07 to be added to the objective function own error. Fortunately this is not always the case.

The kriging metamodel was generated using 107 and 55 points for objective function and constraint respectively. All the NLP sub-problems were solved using SNOPT. Table 1 shows a summary of the main steps in the algorithm.

The last 2 iterations of the algorithm are performed to ensure that the kriging metamodel has enough accuracy. There is an important interval of values for the variables with values of objective function inside the error. Therefore in two consecutive iterations they are equal (below a tolerance) only by chance. Hence, the only valid stopping criterion is based on the errors provided by the kriging. In process flowheeting, as we will see in next examples, the error introduced by the noise is small enough to avoid this situation.

Note that the generation of kriging metamodels consumes an important part of the total CPU time, therefore is important try to balance the accuracy of the metamodel (number of sampled points) with the number of times the kriging should be re-evaluated.

The solution obtained using the kriging metamodel was [2.15, 2.45, 8.83, 5.32, 1.02, 1.46, 1.26, 9.77, 8.21, 8.40] and objective function 24.19, that is close the solution without noise.

Example 2. Separation of a mixture of hydrocarbons using a distillation column.

Consider an example that illustrates the behavior of a modular process simulator when combined with a gradient based optimizer. Assume we want to determine the minimum heat load needed in the reboiler of a distillation column to separate an iso-molar mixture of butanes and pentanes (n-C4 i-C4, n-C5, i-C5), simulated in HYSYS.Plant[21]. We want to recover at least 99% of the butanes fed with at least the 0.99 in combined molar fraction. (model M1) To keep the model as simple as possible, let us assume that we are using a column with 28 theoretical trays (feed in tray 14) and that the pressure in the column remains constant at 450 kPa.

$$\begin{aligned} &\min\ f = Q \\ &s.t \left.\begin{aligned} &\sum_i x_i^{Distillate} \geq 0.99 \\ &\sum_i Molar\ Flow_i^{Distillate} \geq 0.99 \sum_i Feed_i \end{aligned}\right\} i = n\text{-}butane, isobutane \\ &\left[Q, x, Flows\right] = ProcessSimulator(Reflux\ and\ reboil\ \ ratios) \end{aligned} \qquad \text{(M1)}$$

With the pressure fixed there are only two degrees of freedom. In those conditions the distillate (or bottoms flowrate) is almost fixed, and it would be also possible to specify purity. However, in order to illustrate the procedure, we choose two independent variables that converge for almost any specifications: reflux ratio and reboil ratio, and let the optimization algorithm deal with the purity constraints. In general, it is better to use “difficult to converge” specifications to bound the optimal solution and perform the optimization using independent variables that produce “easy to converge” flowsheets.

Using state of the art optimizers (CONOPT[31], SNOPT[32], KNITRO[33]) available in TOMLAB-MATLAB[34] and connected to Hysys.Plant through Windows COM capabilities, with default parameters and derivatives calculated by perturbing the independent variables, all the optimizers fail to obtain a solution. Increasing the perturbation parameter to 10^{-3} SNOPT found an optimal solution, the behavior of CONOPT and KNITRO was a bit erratic –sometimes stopped prematurely in an infeasible solution and others provided a solution-. It was necessary to increase the perturbation parameter to $5 \cdot 10^{-3}$ to get convergence in almost all runs (sometimes the optimizer finished with an infeasible solution). In order to get a solution we must tighten the convergence tolerances in the process simulator –HYSYS- (mass and energy balances to 10^{-6}), in this case we can use a perturbation parameter around $5 \cdot 10^{-4}$ to obtain

consistent solutions (objective function = 1556.69 kW; Reboil ratio = 1.3845; Reflux Ratio = 2.5284).

Consider now the case when we reformulate the problem by minimizing an auxiliary variable α that must be greater or equal than the condenser load (model M2). Clearly, the result should be the same. The first constraint in model M2 must be always active. However, in this case we need to modify not only the perturbation parameter in derivative calculation, but also to relax some tolerances, constraint satisfaction and KKT termination criteria. The reason is that solvers have problems in deciding if the first constraint is active or not, which is very dangerous because we can obtain false optimal solutions.

$$
\begin{aligned}
&\min \ \alpha \\
&s.t. \ \ \alpha \geq Q \\
&\left.\begin{aligned}
&\sum_i x_i^{Distillate} \geq 0.99 \\
&\sum_i Molar\ Flow_i^{Distillate} \geq 0.99 \sum_i Feed_i
\end{aligned}\right\} i = n\text{-}butane,\ isobutane \\
&\left[Q, x, Flows\right] = ProcessSimulator(Reflux\ and\ reboil\ \ ratios)
\end{aligned}
\tag{M2}
$$

To substitute the model of the column by a kriging metamodel –in this case four metamodels (Distillate Flowrate 'D', Reboiler Heat Flow 'Q, and two compositions in distillate) the first thing is to assess the amount of noise introduced by the flowsheet. One possibility is simulating the flowsheet from different starting points, and calculating the standard error. See Figure 6 as an example. It is also necessary to perform sensitivity analysis in order to estimate the influence of the independent variables. For example, the standard error (noise) in the reboiler heat flow is around 0.5-1.0 kW –depending on the values of the independent variables- and the values of reflux and reboil ratios should be separated around 0.01 units in order to get at least 1.5 kW of separation in the heat flow value. These results, although they represent only crude values, provide valuable information: 1. If we want a good interpolating kriging we need that sampled points are separated at least between 3σ -and better at least 6σ - units (for both reboil and reflux ratios); in other case, we must use a non–interpolating version of the kriging. 2. We cannot expect accuracies higher than around 0.01 units in the independent variables. Therefore, as termination criterion we can establish that if the error in two consecutive iterations is in this limit we can stop. If the error estimated by the kriging is large we must confirm the result by contracting the bounds of the kriging to values around this point. Of course, if the statistics of kriging confirm that we have a model whose accuracy if below the fixed limits, we can also stop without further checking. A post optimality sensitivity analysis will confirm all these a-priori estimations

Although for this example it is possible to get very tight bounds for the independent variables, in order to illustrate the procedure, let us start with a wide interval between 0.3 and 3 for both, reflux and reboil ratios. Table 2 shows some of the parameters in the iterations. Initially we used 33 points. The CPU time spent evaluating the 33 sampled points was 2.33 seconds. Then the kriging model for each of the adjusted surfaces was adjusted (Q, D, molar fractions) takes 3.7 seconds. A cross-validation confirms that the kriging is adequate. We consider that the kriging model is adequate if all the sampled points in the cross validation are in the interval $[-3\sigma, 3\sigma]$. Figure 7 shows, as an example, the results of the reboiler heat flow.

The model to solve now can be conceptually represented as follows:

$$\min\ f = Q^M(RR, RB)$$

$$s.t\ x^M_{1,Distillate}(RR, RB) \pm \varepsilon_{x_1^M} + x^M_{2,Distillate}(RR, RB) \pm \varepsilon_{x_2^M} \geq 0.99 \qquad \text{(M3)}$$

$$\left(D^M \pm \varepsilon_{x_1^M}\right)\left(x^M_{1,Distillate} + x^M_{2,Distillate} \pm (\varepsilon_{x_1^M} + \varepsilon_{x_2^M})\right) \geq 0.99 \sum_i Feed_i$$

where the superscript 'M' makes reference to the magnitudes calculated through a metamodel. The errors in constrained problems like M3 is that they can produce infeasible solutions, although in the actual model the solution is feasible. In this example we follow the criterion of removing the errors. If some of the sub-problems result to be infeasible, we will consider that the problem is feasible if the constraint(s) that produce infeasibility is under the error predicted by the kriging metamodel.

The solution to this initial problem is x = [1.519, 2.643] (reflux and reboil ratios respectively) and objective function 1630.8 kW. But the solution is infeasible. The residuals in the solution for the two constraints are [0.0014, 0] respectively. The standard deviations calculated by the kriging (see also Table 2) are $\sigma_Q = 40.3 kW$; $\sigma_D = 10.0$; $\sigma_{x_1} = 0.0139$; $\sigma_{x_2} = 0.0061$; Therefore, following the criteria of assuming that the solution can be considered feasible if the kriging error is larger than the infeasibility we can mark the problem as 'possibly feasible' and continue with the algorithm. Note that alternatively it would be possible to solve the problem like in equation 14, but maybe except in highly constrained problems in which is difficult to find a feasible point, the previous approach usually works better.

Instead of contracting the feasible region, it is possible to improve the kriging model by adding the optimal point without re-optimizing the kriging parameters. This procedure has almost no computational cost. Then, with the new point added, and starting from the previous obtained solution the problem is solved again. If the point found in the first problem was near optimum this second problem is expected to converge fast. Besides, if the new point is near the previous one, the error predicted by the kriging metamodel should decrease, because the interpolated point is expected to be near a sampled point. Note that in this example adding this new point produces a feasible solution. The procedure continues until there is not improvement in two iterations or a pre-specified number of iterations.

However, in the previous refining stage one important consideration must be taken into account. The new sampled points must be separated enough to guarantee that the noise is not a dominant effect and to avoid 'ill conditioning' during the matrix inversion. The first is mitigated by the fact that sampling in the flowsheet starting from closed points tend to decrease the noise. The second is more related with the kriging implementation, in our case the independent variables are scaled to values between 0-1 which help to maintain stability. After this stage the results are: f = 1556 kW x=[1.3842, 2.5281] and the standard deviations calculated by the model are $\sigma_Q = 1.2 kW$; $\sigma_D = 0.16$; $\sigma_{x_1} = 1.7 \cdot 10^{-4}$; $\sigma_{x_2} = 2.9 \cdot 10^{-5}$,(see further details in Table 2):

Although previous errors could be within the tolerance, at this stage we cannot ensure that we have obtained the optimal result. As mentioned in previous sections we must ensure that around the solution the model reproduces the functions and their derivatives. The next step is therefore, contracting the bounds of the independent variables, the new domain is centered in the last best point. At this point we sample again, inside the new domain, including as one of the sampled points the last obtained solution. In the example, in this second major iteration the

optimal solution coincides with that in the first iteration and errors are inside tolerance, and search is stopped.

The contraction factor is usually not critical (in this example we reduce the region in an 80%). If the reduction is too large and the optimal solution is inside the new region, the iteration will finish in a face (edge or vertex) of the hypercube. In the next iteration there will be only a movement of the domain without contraction.

The total CPU time used in the optimization (including sampling, kriging optimization and model optimization) was around 13 s.

Example 3. Sequence of two distillation columns

In this example we separate a mixture of three hydrocarbons in their pure components using a direct distillation sequence of two columns. Table 3 shows the data for the example.

The objective in this case consists of minimizing the sum of maximum vapor flows in both columns, and we want recovery the 95% (in molar basis) of each of the hydrocarbons in the initial mixture. To simplify the problem we assume that the pressure is known and constant. With this information it would be possible to get tight bounds of the independent variables, but as in previous examples, in order to illustrate the algorithm we started with a larger interval.

In this example there are two columns. The first one is similar to the column in the second example. For the second one we have two approaches: 1. completely separate it from the first column (the independent variables for this model will be the two degrees of freedom of the column –i.e. Reflux and reboil ratios- and the independent variables in the feed –i.e. molar flow rates of each component-). 2. The feed to the second column depends on the first column (bottoms stream of the first column). Therefore, it is possible to remove the feed of this second column from the kriging and consider as independent variables for modeling the second column the reboil and reflux ratios of the first column together with its own reflux and reboil ratios.

The second approach is more compact, but we lose information about the feed to the second column. In the first one, the kriging model for the second column has higher dimensionality (the number of components in the feed plus the specifications in the column). However, this situation is mitigated because the kriging metamodel for the second column is not developed for the entire domain of feed individual component flowrate values, but only for the domain of possible values obtained when fitting the kriging in the first column. It is worth noting that in the first major iteration errors larger than in the compact approach are expected. But as the feasible region is contracted, the possible values for the feed of the second columns are rapidly constrained to a small region and both approaches have similar behaviors. These two models can be conceptually represented as follows:

$$\min: \quad f = V^{c_1} + V^{c_2}$$

$$s.t. \quad V^{c_1} = V^{c_1}(RR^{c_1},\ RB^{c_1})$$

$$V^{c_2} = V^{c_2}(RR^{c_1},\ RB^{c_1}, RR^{c_2},\ RB^{c_2})$$

$$x^{C_5} \geq 0.95; \quad x^{C_6} \geq 0.95; \quad x^{C_7} \geq 0.95;$$

$$x^{C_5} = x^{C_5}(RR^{c_1},\ RB^{c_1})$$

$$x^{C_6} = x^{C_6}(RR^{c_1},\ RB^{c_1}, RR^{c_2},\ RB^{c_2})$$

$$x^{C_7} = x^{C_7}(RR^{c_1},\ RB^{c_1}, RR^{c_2},\ RB^{c_2})$$

$$0.3 \leq RR^{c_1} \leq 3; \quad 0.3 \leq RB^{c_1} \leq 5;$$

$$0.3 \leq RR^{c_2} \leq 3; \quad 0.3 \leq RB^{c_2} \leq 5;$$

16(a)

$$\min: \quad f = V^{c_1} + V^{c_2}$$

$$s.t. \quad V^{c_1} = V^{c_1}(RR^{c_1},\ RB^{c_1})$$

$$V^{c_2} = V^{c_2}(F^{C5}, F^{C6}, F^{C7}, RR^{c_2},\ RB^{c_2})$$

$$x^{C_5} \geq 0.95; \quad x^{C_6} \geq 0.95; \quad x^{C_7} \geq 0.95;$$

$$x^{C_5} = x^{C_5}(RR^{c_1},\ RB^{c_1})$$

$$B^{C5} = B^{C5}(RR^{c_1},\ RB^{c_1})$$

$$B^{C6} = B^{C6}(RR^{c_1},\ RB^{c_1})$$

$$B^{C7} = B^{C7}(RR^{c_1},\ RB^{c_1})$$

$$B^{C5} = F^{C5}$$

$$B^{C6} = F^{C6}$$

$$B^{C6} = F^{C7}$$

$$x^{C_6} = x^{C_6}(F^{C5}, F^{C6}, F^{C7}, RR^{c_2}, RB^{c_2})$$

$$x^{C_7} = x^{C_7}(F^{C5}, F^{C6}, F^{C7}, RR^{c_2},\ RB^{c_2})$$

$$0.3 \leq RR^{c_1} \leq 3; \quad 0.3 \leq RB^{c_1} \leq 5;$$

16(b)

where c_1, c_2 make reference to columns 1 and 2, F^{Ci} is the feed flowrate to the second column. B^{Ci} is the bottoms stream errors of the first column. RR and RB are its reflux and reboil ratios respectively.

All the NLP sub-problems were solved using SNOPT. Figure 8 shows the scheme of the columns with the optimal results. Tables 4 and 5 show the main statistics of the algorithm for both models, case 1 equation 16(a), and case 2 equation 16(b). It is interesting to note that the initial problem in the second approach is infeasible within the error predicted by the kriging. As soon as this point is added to the kriging and updated (without modifying adjustable parameters), the model become feasible.

Example 4. Phthalic anhydride from O-Xylene

This example uses the synthesis of Phthalic Anhydride from O-Xylene to illustrate how a unit operation, (a plug flow reactor), can be substituted by a kriging metamodel in a flow-sheet optimization.

The most common method for production of phthalic anhydride is by catalytic oxidation of O-Xylene, in vapour phase, with a V_2O_5 catalyst. Phthalic anhydride is used in the manufacture of plasticizers (additives to polymers to give them more flexibility) and polyesters, among other applications.

Initially an air stream is compressed up to 3 atm, and mixed with a pressurized O-Xylene stream. The mixture is vaporized, heated and then introduced in a reactor. The reactions are highly exothermic, and the reactor must be refrigerated, in this case using the eutectic salt

HiTec™ (40% $NaNO_2$, 7% $NaNO_3$, 53% KNO_3), common in refrigeration of continuous processes at high temperatures. The reactor temperature must be between 300 and 400 ºC. The product is a mixture of phthalic anhydride, maleic anhydride, o-xylene, CO_2, water, O_2, and N_2. The product mixture is cooled, depressurizes until 1.5 atm, and separated using a flash and a distillation column. The phthalic anhydride is obtained in the bottom stream, with a molar fraction higher than 0.999. Figure 9 shows the flow-sheet.

The reaction is modeled as a non-isothermal plug flow reactor using the following reactions:

$$(R1)\quad 3O_2 + \underset{\text{o-xylene}}{C_8H_{10}} \longrightarrow 3H_2O + \underset{\text{phthalic anhydride}}{C_6H_4(CO_2)_2O}$$

$$(R2)\quad 7.5O_2 + \underset{\text{o-xylene}}{C_8H_{10}} \longrightarrow \underset{\textit{maleic anhydride}}{C_4H_2O_3} + 4CO_2 + 4H_2O$$

$$(R3)\quad 10O_2 + \underset{\text{o-xylene}}{C_8H_{10}} \longrightarrow 8CO_2 + 5H_2O$$

High pressure favors the main reaction, therefore it is fixed to the maximum recommended for the process. Independent variables are temperatures of the reactor inlet and outlet streams, reactor volume, temperature and pressure in the stream entering the flash. Relevant data are presented in Table 6.

The objective of the optimization is to maximize the annual profit. The investment cost includes cost for compressor, pump, heater, cooler, reactor, flash separator and distillation column (vessel, trays, reboiler and condenser). These costs were calculated using the correlations presented in Turton et al.[35] and annualized as described by Simth[36]. Services include heating and cooling utilities and electricity as well as raw materials and product (phthalic anhydride) prices.

The major difficulty when optimizing the process is related with the reactor. The differential equations involved introduce noise and the time to solve them is not negligible. Therefore, the reactor is substituted by a kriging metamodel, while the rest of unit operations continue to be calculated by the simulator. Therefore we have a hybrid system.

Although the reactor is substituted by a kriging model it is not convenient to substitute all the individual flows (phthalic anhydride, maleic anhydride, o-xylene, CO_2, water, O_2, and N_2) by a kriging model. If we do that, the errors introduced by the model violate the mass and energy balances. Therefore we develop a metamodel for the phthalic anhydride, the maleic anhydride and CO_2. The other components can be calculated at the exit by a simple mass balance. Once the mass balances have been calculated, the heat removed from the reactor is calculated by the simulator by an energy balance.

In this example the limiting stage is the sampling. Therefore, in order to avoid excessive number of major iterations we use 71 points (relatively large if compared with previous examples) to get a good metamodel and avoid further contraction steps. The algorithm converges in 2 major iterations. Table 7 and Figure 9 show the statistics of the algorithm and the main results.

Conclusions and Final Remarks

In this work, a new algorithm for constrained optimization containing implicit noisy black box functions was presented. The noisy implicit functions are substituted by a kriging metamodel

that allows the fast interpolation of new values and simultaneously calculate the standard error of the predicted point. The predicted errors in the model together with the knowledge of the error introduced by the system provide us with a reliable stopping criterion.

In noisy constrained systems it is possible obtain infeasibility due to inaccuracies in the constraints and objective function. The standard deviation predicted by the kriging can be used a as criterion for infeasibility if it is larger than the error (for example if the infeasibility is larger than 3σ, with a 99.7% confidence), or it can consider the point as 'possible feasible' if the infeasibilities are below that error. The refining stage –without recalibrating the kriging- and the bounds contraction in order to improve the kriging interpolation will confirm if the point is feasible or not. In each iteration, the optimization is performed substituting the black box function by its kriging metamodel using a NLP solver (SNOPT). Each problem is solved to optimality, the kriging is updated (i.e. adding the last best point) and the procedure is repeated.

It is worth mentioning that in all the examples presented, and in most of the applications it is possible to add new points without recalibrating the kriging parameters. Since kriging is an interpolating method, the value in these new points is exact, and the quality of the interpolation improves. However, the cost is that we have to invert a large matrix, which increases the CPU time in the optimization although it tends to decrease the total number of major iterations.

The major capability of the kriging metamodel is that it captures the main trends in the model; even in the case in which the interpolation is not accurate. Successive contractions can locate the optimum in a reduced number of major iterations. However, the quality of the kriging metamodel depends on the number of variables and on how much the bounds of the independent variables can be tightened. A pre-processing stage whose objective is only to find a near optimal region would improve the performance of the optimization avoiding unnecessary contraction steps. If the algorithm is used to substitute deterministic models (without noise), the convergence of the model is guarantee to a local stationary point, because it is possible to assure that both the original model and their gradient information are matched by the metamodel within a given tolerance.

The algorithm presented has proved to be robust and reliable when some of the most difficult to converge and usually noisy unit operations (distillation columns and reactors) are substituted by a kriging model. Finally, the statistical parameters calculated by the kriging approach, ensures a solution in the limits of the error introduced by the noise and (or) bound the errors assumed.

Acknowledgements

The authors gratefully acknowledge the financial support from the "Ministerio de Educacion y Ciencia" in Spain under project CTQ 2005 – 05456.

References

1. Papalambros PY, Wilde DJ. *Principles of Optimal Design: Modeling and Computation*. 2nd Edition ed: Cambridge University Press; 2000.

2. Jones DR, Schonlau M, Welch WJ. Efficient global optimization of expensive black-box functions. *Journal Of Global Optimization.* Dec 1998;13(4):455-492.

3. Krige DG. *A Statistical Approach to Some Mine Valuations and Allied Problems at the Witwatersrand.*, University of Witwatersrand; 1951.

4. Cressie N. Spatial Prediction And Ordinary Kriging. *Mathematical Geology.* May 1988;20(4):405-421.

5. Goovaerts P. *Geostatistics for Natural Resources Evaluation*. New York: Oxford University Press.; 1997.

6. Kushner HJ. A new method of locating the maximum point of an arbitrary multipeak curve in presence of noise. *Journal of Basic Engineering.* 1964;86:97-106.

7. Betro B. Bayesian methods in global optimization. *Journal Of Global Optimization.* 1991;1:1-14.

8. Boender CGE, Romeijn HE. Stochastic methods. In: Horst R, Pardalos PM, eds. *Handbook of Global Optimization*. Dordrecht/London/Boston: Kluwer Academic Publishers; 1995:829-869.

9. Mockus J. Application Of Bayesian-Approach To Numerical-Methods Of Global And Stochastic Optimization. *Journal Of Global Optimization.* Jun 1994;4(4):347-365.

10. Zilinskas A. A review of statistical models for global optimization. *Journal Of Global Optimization.* 1992;2:154-153.

11. Sacks J, Schiller SB, Welch WJ. Design for computer experiments. *Technometrics.* 1989;31:41-47.

12. Sacks J, Welch WJ, Mitchell WJ. Design and Analysis of Computer Experiments. *Statistical Science.* 1989;4(4):409-435.

13. Jones DR. A taxonomy of global optimization methods based on response surfaces. *Journal Of Global Optimization.* 2001;21(4):345-383.

14. Simpson TW, Mistree F. Kriging models for global approximation in simulation-based multidisciplinary design optimization. *Aiaa Journal.* Dec 2001;39(12):2233-2241.

15. Sasena MJ. *Flexibility and Efficiency Enhancements for Constrained Global Design Optimization with Kriging Approximations.* [Ph. D Thesis]. Michigan, University of Michigan; 2002.

16. *The Language of Technical Computing.* [computer program]. Version: The Mathworks Inc; 2006.

17. Biegler LT, Grossmann IE. Retrospective on optimization. *Computers & Chemical Engineering.* Jul 15 2004;28(8):1169-1192.

18. Lang YD, Biegler LT. A unified algorithm for flowsheet optimization. *Computers & Chemical Engineering.* 1987;11(2):143-158.

19. Michelewicz Z. A survey of constraint handling techniques in evolutionary computation methods. In: McDonnell J, Reynolds R, Fogel D, eds. *Evolutionary Programming IV*. Cambridge, MA.: MIT Press; 1995.

20. Michalewicz Z, Dasgupta D, Le Riche RG, Schoenauer M. Evolutionary algorithms for constrained engineering problems. *J Comput. Ind. Eng.* 1996;30(4):851-870.

21. *Hyprotech Ltd.* [computer program]. Version; 1995-2002.

22. Diwekar U. *Introduction to Applied Optimization.* Dordrecht: Kluwer Academic Publishers.; 2003.

23. McKay MD, Conover WJ, Beckman RJ. A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics.* 1979;21(2):239-245.

24. Iman RL, Helton JC, Camobell JE. An approach to sensitivity analysis of computer models, Part 1. Introduction, input variable selection and preliminary variable assestment. *Journal of Quality Technology.* 1981;13(3):174-183.

25. Kalagnanan JR, Diwekar U. An efficient sampling technique for off-line quality control. *Technometrics.* 1997;21(2):239.

26. Press WH, Teukolsky SA, Vetterling WT, Flannery BP. *Numerical Recipies in Fortran*. Cambridge: Cambridge University Press; 1992.

27. Biegler LT, Grossmann IE, Westerberg AW. A Note on Approximation Techniques used for Process Optimization. *Computers & Chemical Engineering.* 1985;9(2):201-206.

28. Alexandrov NM, Lewis RM, Gumbert CR, Green LL, Newman PA. Optimization with variable-fidelity models applied to wing design. Paper presented at: 38th Aerospaces Sciences Meeting & Exibit. AIAA Paper 2000-0841, 2000.

29. Davis E, Ierapetritou M. A kriging method for the solution of nonlinear programs with black-box functions. *AIChE Journal.* 2007;53(8):2001-2012.

30. Takahama T, Sakai S. Constrained Optimization by alpha Conatrained Genetic Algorithm. *Systems and Computers in Japan.* 2004;35(5):11-22.

31. Drud AS. *CONOPT: A System for Large Scale Nonlinear Optimization. Reference Manual.* Bagsvaerd, Denmark: ARKI Consulting and Development A/S; 1996.

32. Gill WPE, Murray W, Saunders. SNOPT: an SQP algorithm for large-scale constrained optimization. *SIAM J. Optimization.* 2002:979-1006.

33. Waltz RA, Plantenga TD. *KNITRO User's Manual*: Ziena Optimization, Inc.; 2007.

34. Holmström K. The Tomlab Optimization Environment in Matlab. *Adv. Model Optim.* 1999;1:47-69.

35. Turton R, Bailei RC, Whiting WB, Shaeiwitz JA. *Analysis Synthesis and Design of Chemical Processes*. New York: McGraw-Hill.; 1998.

36. Smith R. *Chemical Process Design and Integration*. Chichester: John wiley & Sons, Ltd.; 2005.

Table 1. Summary of results from example 1.

Iteration 1:

Number of sampled points in f = 107 — CPU time kriging f = 30.6 s
Number of sampled points in z = 55 — CPU time kriging z = 6.6 s

Optimal values (after refining stage)

x1	2.06	x6	2.30	f	75.52
x2	3.46	x7	0.52	z	-50.58
x3	7.21	x8	7.96	CPU time(s)	3.65
x4	5.12	x9	7.65		
x5	1.40	x10	8..37		

Iteration 2:

Number of sampled points in f = 108 — CPU time kriging f = 24.5 s
Number of sampled points in z = 56 — CPU time kriging z = 4.98 s

Optimal values (after refining stage)

x1	2.13	x6	1.29	f	43.88
x2	3.46	x7	1.44	z	-64.0
x3	6.88	x8	9.28	CPU time(s)	2.08
x4	5.58	x9	7.24		
x5	0.98	x10	7.05		

Iteration 3:

Number of sampled points in f = 108 — CPU time kriging f = 28.9 s
Number of sampled points in z = 57 — CPU time kriging z = 4.9 s

Optimal values (after refining stage)

x1	2.14	x6	1.17	f	31.39
x2	2.90	x7	1.01	z	-34.44
x3	7.62	x8	9.44	CPU time(s)	1.78
x4	4.39	x9	7.70		
x5	0.95	x10	7.56		

Iteration 4:

Number of sampled points in f = 108 — CPU time kriging f = 26.48 s
Number of sampled points in z = 56 — CPU time kriging z = 5.98 s

Optimal values (after refining stage)

x1	2.21	x6	1.81	f	25.80
x2	2.53	x7	1.25	z	-18.50
x3	8.26	x8	9.69	CPU time(s)	1.02
x4	4.89	x9	8.21		
x5	1.18	x10	8.20		

Iteration 5:

Number of sampled points in f = 108 — CPU time kriging f = 18.64 s
Number of sampled points in z = 56 — CPU time kriging z = 5.73 s

Optimal values (after refining stage)

x1	2.21	x6	1.49	f	24.52
x2	2.49	x7	1.27	z	-10.28
x3	8.56	x8	9.72	CPU time(s)	1.19
x4	5.44	x9	8.33		
x5	1.04	x10	8.50		

Table 1. Summary of results from example 1.(cont)

Iteration 6:

Number of sampled points in f = 108 — CPU time kriging f = 16.89 s
Number of sampled points in z = 56 — CPU time kriging z = 5.53 s

Optimal values (after refining stage)

x1	2.15	x6	1.52	f	24.28
x2	2.44	x7	1.26	z	-0.0004
x3	8.84	x8	9.77	CPU time(s)	1.311
x4	5.32	x9	8.14		
x5	1.04	x10	8.22		

Iteration 7:

Number of sampled points in f = 108 — CPU time kriging f = 16.56 s
Number of sampled points in z = 56 — CPU time kriging z = 5.72 s

Optimal values (after refining stage)

x1	2.15	x6	1.50	f	24.19
x2	2.43	x7	1.27	z	0.00
x3	8.81	x8	9.78	CPU time(s)	1.22
x4	5.25	x9	8.22		
x5	1.03	x10	8.39		

Iteration 8:

Number of sampled points in f = 108 — CPU time kriging f = 16.51 s
Number of sampled points in z = 56 — CPU time kriging z = 4.11 s

Optimal values (after refining stage)

x1	2.15	x6	1.46	f	24.19
x2	2.45	x7	1.26	z	0.00
x3	8.83	x8	9.77	CPU time(s)	1.23
x4	5.31	x9	8.21		
x5	1.02	x10	8.40		

Total CPU time (s) = 236.11 (94.3% kriging generation)

Table 2. Summary of algorithm for example 2.

Variables: Reflux ratio (RR); Reboil Ratio (RB)

Initial interval: $0.3 \le RR \le 3; \quad 0.3 \le RB \le 3$

Kriging metamodels: Q = Reboiler Heat Flor (kW)
D = Distillate Flow rate (kmol/h)
xD_1 = molar fraction of n-butane in distillate
xD_2 = molar fraction of iso-butane in distillate

NLP Solver : SNOPT

Iteration 1
Number of sampled points = 33
Sampling time = 1.927 s

Metamodel	Q	D	xD_1	xD_2
μ	1103	67.3	0.560	0.357
σ	254	18.1	0.052	0.037
θ	[1.072, 4.346]	[0.974, 3.749]	[2.585, 2.090]	[5.048, 9.728]
P	[1.318, 1.415]	[1.070, 1.367]	[1.398, 1.467]	[1.675, 1.998]
CPU time (s)	0.735	0.766	0.812	1.406

Optimization and refinement

	RR	RB	Q kW (σ_Q)	D (kmol/h) (σ_D)	xD_1 (σ_{x_1})	xD_2 (σ_{x_2})	CPU time (s)
1	1.519	2.634	1630 (40)	100.1 (10)	0.499 (0.014)	0.489 (0.061)	0.344
2	1.431	2.567	1581 (44)	100.0 (3.7)	0.499 (0.007)	0.491 (0.002)	0.046
3	1.409	2.549	1569 (23)	100.0 (2.1)	0.4996 (0.004)	0.4904 (0.0014)	0.078
4	1.386	2.529	1557 (21)	100.0 (2.1)	0.4999 (0.004)	0.4901 (0.0014)	0.032
5	1.384	2.528	1556 (1.2)	100.0 (0.16)	0.4998 (0.0002)	0.4902 (0.0001)	0.047

Table 2.(cont) Summary of algorithm for example 2.

Iteration 2
Number of sampled points 34
Sampling time = 1.823 s.

Metamodel	Q	D	xD_1	xD_2
μ	1789	103.1	0.4904	0.4446
σ	136	5.0	0.0243	0.0166
θ	[1.463, 3.054]	[2.779, 7.080]	[2.948, 7.331]	[7.729, 41.780]
P	[1.247, 1.331]	[1.362, 1.373]	[1.396, 1.377]	[1.579, 1.888]
CPU time (s)	0.688	0.828	0.734	1.828

Optimization and refinement

	RR	RB	Q kW (σ_Q)	D (kmol/h) (σ_D)	xD1 (σ_{x_1})	xD2 (σ_{x_2})	CPU time (s)
1	1.400	2.542	1564 (27)	100.0 (1.3)	0.4998 (0.0061)	0.4902 (0.0034)	0.047
2	1.388	2.532	1558 (13)	100.0 (0.6)	0.4998 (0.0026)	0.4902 (0.0012)	0.063
3	1.385	2.529	1557 (6)	100.0 (0.2)	0.4998 (0.0011)	0.4902 (0.0005)	0.125
4	1.384	2.528	1557 (1)	100.0 (<0.1)	0.4998 (0.0001)	0.4902 ($<10^{-4}$)	0.45

Total CPU time (s) = 12.779 (29.4% sampling; 61.0% kriging generation)

Q in kW, D in kmol h^{-1}, xD molar fraction.

Table 3 Data of example 3.

Components : n-pentane; n-hexane; n-butane
Thermodynamics: Peng Robinsong equation of state.
Assumed constant pressure in the columns (200 kPa)

Feed		Column 1		Column 2	
P (kPa)	200 kPa	Trays	25	Trays	32
T (ºC)	83.89	Feed Tray	14	Feed Tray	16
Molar Flow (kmol/h)	200	Specifications	Reflux ratio Reboil Ratio	Specifications	Reflux ratio Reboil Ratio
Composition (molar fract)	[0.3, 0.4, 0.3]				

External constraints: Recovery of each component with a purity higher than 95% (molar)

Table 4. Summary of the main steps in algorithm in example 3. Case 1.

Variables: Reflux ratio (RR); Reboil Ratio (RB) in each column

Initial interval: $0.3 \leq RR_i \leq 3; \quad 0.3 \leq RB_i \leq 5; \quad i = 1, 2.$

Kriging metamodels: V^{c1} = Maximum vapour flow rate in column 1 (kmol/h)
x^{C5} = C5 molar fraction in distillate of column 1
V^{c2} = Maximum vapour flow rate in column 2 (kmol/h)
x^{C6} = C6 molar fraction in distillate of column 2
x^{C7} = C7 molar fraction in bottoms of column 2

NLP Solver : SNOPT

Iteration 1:
Number of sampled points in V^{c1}, x^{C5}= 35
Number of sampled points in Vc2, x^{C6}, x^{C7} = 73

Independent variables Reflux ratio (RR) and Reboil ratio (RB) in each column

CPU time kriging V^{c1} = 0.92 s
CPU time kriging x^{C5} = 1.27
CPU time kriging V^{c2} = 5.89 s
CPU time kriging x^{C6} = 16.59
CPU time kriging x^{C7} = 15.32

Sampling time = 13.59 s

Optimal values (after refining stage)

RR1	1.948	RR2	1.511	F (kmol/h)	374.8
RB1	1.172	RB2	2.992	CPU time (s)	7.9

Iteration 2:
Number of sampled points in V^{c1}, x^{C5}= 36
Number of sampled points in Vc2, x^{C6}, x^{C7} = 74

CPU time kriging V^{c1} = 0.63 s
CPU time kriging x^{C5} = 0.81 s
CPU time kriging V^{c2} = 5.82 s
CPU time kriging x^{C6} = 6.59 s
CPU time kriging x^{C7} = 7.50 s

Sampling time = 12.9 s

Optimal values (after refining stage)

RR1	1.945	RR2	1.501	F (kmol/h)	374.8
RB1	1.180	RB2	2.999	CPU time (s)	1.28

Total CPU time (s) = 97.0 (27.3% sampling; 63.23% kriging generation)

Table 5. Summary of the main steps in algorithm in example 3. Case 2.

Iteration 1:
Number of sampled points in V^{c1}, x^{C5}, F^{C5}, F^{C6}, F^{C7} = 35
Number of sampled points in Vc2, x^{C6}, x^{C7} = 73

CPU time kriging V^{c1} = 1.77 s
CPU time kriging x^{C5} = 1.33 s
CPU time kriging F^{C5} = 1.42 s
CPU time kriging F^{C6} = 1.78 s
CPU time kriging F^{C7} = 3.13 s
CPU time kriging V^{c2} = 9.12 s
CPU time kriging x^{C6} = 10.95
CPU time kriging x^{C7} = 13.41

Sampling time = 8.8 s

Optimal values (after refining stage)

RR1	1.934	RR2	1.518	F (kmol/h)	374.8
RB1	1.175	RB2	2.999	CPU time (s)	14.53

Iteration 2:
Number of sampled points in V^{c1}, x^{C5}, F^{C5}, F^{C6}, F^{C7} = 36
Number of sampled points in Vc2, x^{C6}, x^{C7} = 74

CPU time kriging V^{c1} = 1.52s
CPU time kriging x^{C5} = 1.53 s
CPU time kriging F^{C5} = 1.72 s
CPU time kriging F^{C6} = 1.36 s
CPU time kriging F^{C7} = 2.33 s
CPU time kriging V^{c2} = 10.00 s
CPU time kriging x^{C6} = 12.91 s
CPU time kriging x^{C7} = 14.39

Sampling time = 8.5 s

Optimal values (after refining stage)

RR1	1.939	RR2	1.509	F (kmol/h)	374.9
RB1	1.180	RB2	2.994	CPU time (s)	2.09

Total CPU time (s) = 16.62 (14.1 % sampling; 72.3% kriging generation)

Table 6. Data for example 4

Components: Air, o-xylene, phthalic anhydride, maleic anhydride, water, CO_2.

Thermodynamics: UNIQUAC

Feeds	Air	o-xylene	Distillation Column	
P (kPa)	101.3	30.82	Trays	10
T (ºC)	25	25	Feed Tray	5
Molar Flow (kmol/h)	1000	30.82	Specifications	Reflux ratio Phthalic anhydride purity 99.9% in bottoms strams

Cost Data			
Phthalic anhydride	1554 $/ton	Cooling water	0.354 $/GJ
o-xylene	1106 $/ton	Molten salt	12 $ /GJ
		HP steam	9.83 $/GJ
		Electricity	480 $/kW-year

Table 7. Summary of the main steps in algorithm in example 4.

Variables T1 = Temperature in the reactor inlet stream(ºC)

T2 = Temperature in the reactor inlet stream (ºC)
L = Reactor length (m);
T = exit cooler temperature (ºC)

Initial interval:

$$300 \le T1 \le 400; \quad 300 \le T2 \le 400; \quad 20 \le L \le 180; \quad 25 \le T3 \le 300;$$

Kriging metamodels: Pht = Molar flow at the exit of reactor of phthalic anhydride
Ma = Molar flow at the exit of reactor of maleic anhydride
CO_2 = Molar fraction at the exit of reactor of CO_2

NLP Solver : SNOPT

Iteration 1:
Number of sampled points = 71

CPU time kriging Pht (T1, T2, L) = 5.6 s
CPU time kriging Ma (T1, T2, L) = 6.5 s
CPU time kriging CO_2 (T1, T2, L) = 6.6 s

Sampling time = 73.1 s

Optimal values in first iteration before refining					
T1	323.4	T2	392.1	L	121.6
T3	72.8	Objective	3.92 10^6 $/year		
Optimal values in first iteration after refining					
T1	306.7	T2	397.1	L	144.0
T3	71.0	Objective	4.01 10^6 $/year		
CPU time (s)	13.1				

Iteration 2:
Number of sampled points = 72

CPU time kriging Pht (T1, T2, L) = 5.3 s
CPU time kriging Ma (T1, T2, L) = 6.6 s
CPU time kriging CO_2 (T1, T2, L) = 6.2 s

Sampling time = 70.4 s

Optimal after refining					
T1	306.6	T2	396.8	L	144.0
T3	71.0	Objective	4.02 10^6 $/year		
CPU time(s)	3.49				
Total CPU time (s) = 169.9 s (72.9 % sampling; 18.7 % kriging generation)					

Figure Captions

Figure 1. Level curves for 'peaks' function. (a) Actual model (b) kriging interpolation using the 33 points marked as dots in figure. In both figures level curves correspond to the same function values

Figure 2. 3 D plot of the 'peaks' function. (a) actual model. (b) kriging interpolation using the 33 points showed in Figure 1a. (c) standard error predicted by the kriging.

Figure 3. Example of non-interpolating kriging. 3σ error lines, predicted by kriging, (99.7% confidence) are included. U makes reference to uniform random error.

Figure 4. Flowchart of kriging based NLP optimization algorithm

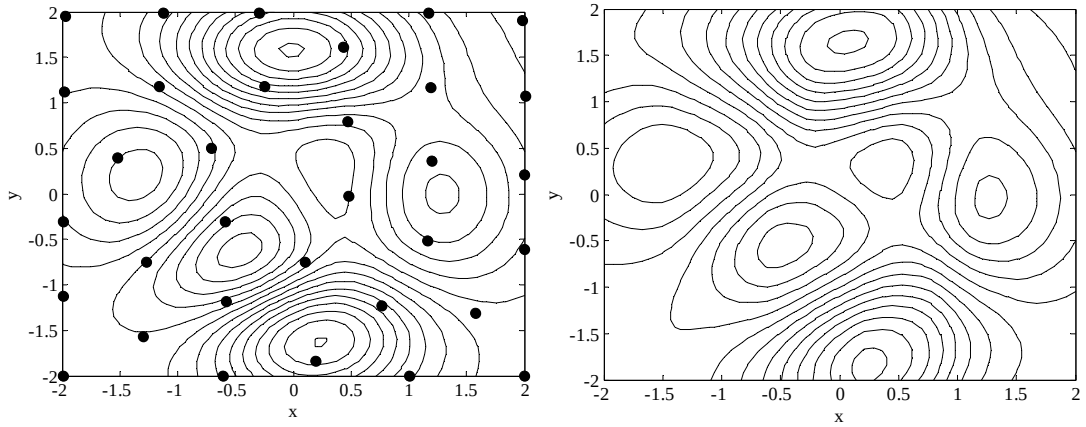
Figure 5. Graphical representation of the movements in the sampling hypercube –squares-. a) If the optimal point in iteration k is inside the sampling region in iteration k+1 the sampling region is contracted and centred in the best point from previous iteration. b) If the optimal point is in the limit of the sampling region, this region is simply moved. c) Same as case a but now the optimal point in iteration k is at a bound (dotted line). Note that the limits of the hypercube are used only to decide if there is or not contraction, but actual sampling is constrained to values inside the domain of variables. d) Same as b with optimal point simultaneously in the limit of sampling hypercube and domain of a variable.

Figure 6. Value of the reboiler heat flow rate (kW) in distillation column of example 1 obtained from 100 runs starting from random initial points. For these 100 points the mean value is 1556.4 kW with a standard deviation equal 0.49.

Figure 7. Cross validation for the heat flow rate in reboiler of distillation column in example 1. (a) number of standard errors that a point is above or bellow the predicted value. All the values are clearly in the interval [-3,3]. (b) Comparison of actual an interpolated values. The outliers are points in the borders of the sampling region, when they are removed in cross validation the rest of the points are performing an extrapolation instead of an interpolation which justifies their deviation.

Figure 8. Scheme of the columns sequence in example 3. Optimal solution and some relevant data are included.

Figure 9. Hysys scheme of the Petlyuk flowsheet to obtain phthalic anhydride from o-xylene, example 4. . Optimal solution and some relevant data are included.



(a)

(b)

$$z = 3(1-x)^2 e^{-x^2} - (y+1)^2 - 10\left(\frac{x}{5} - x^3 - y^5\right)e^{\left(-x^2-y^2\right)} - \frac{1}{3}e^{\left(-(x+1)^2-y^2\right)}$$

Figure 1

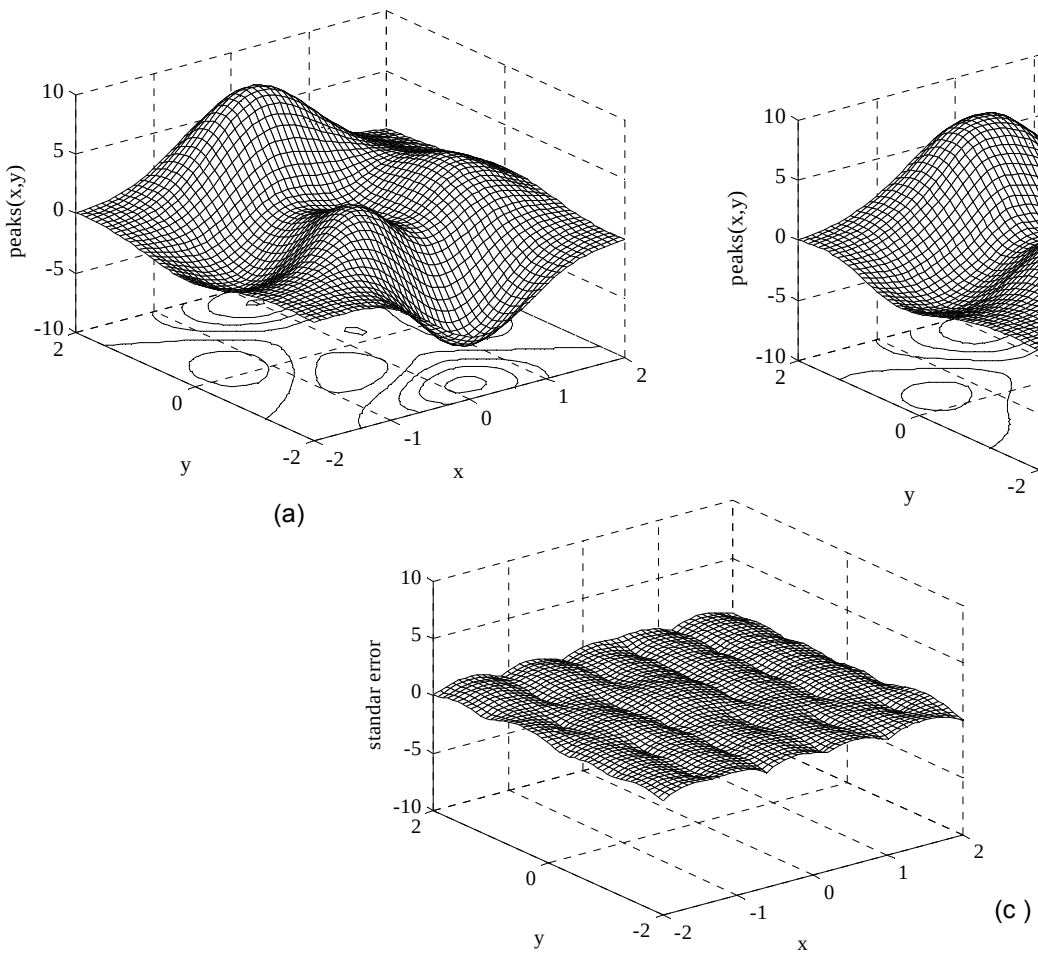

peaks(x,y)
10
5
0
-5
-10
2
0
-2
y
-2
-1
0
1
2
x
(a)
peaks(x,y)
10
5
0
-5
-10
2
0
-2
y
-2
-1
0
x
(b)
standar error
10
5
0
-5
-10
2
0
-2
y
-2
-1
0
1
2
x
(c)

Figure 2.

$$y = \sin\left((3x)^2\right) + U(-0.15, 0.15) \qquad 0 \le x \le 1$$

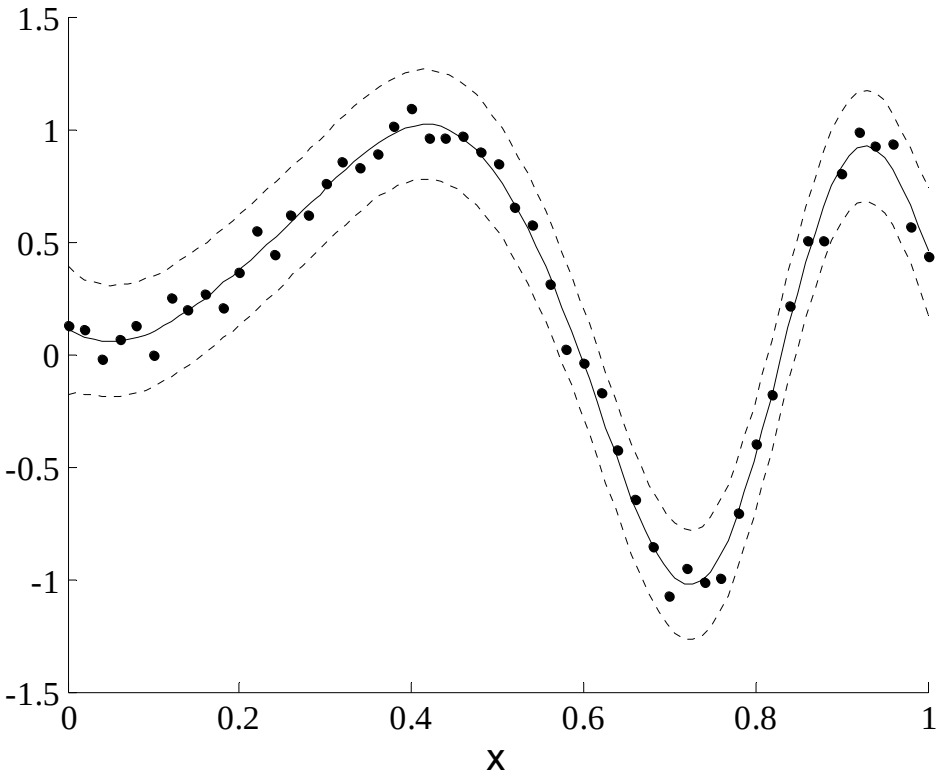


Figure 3

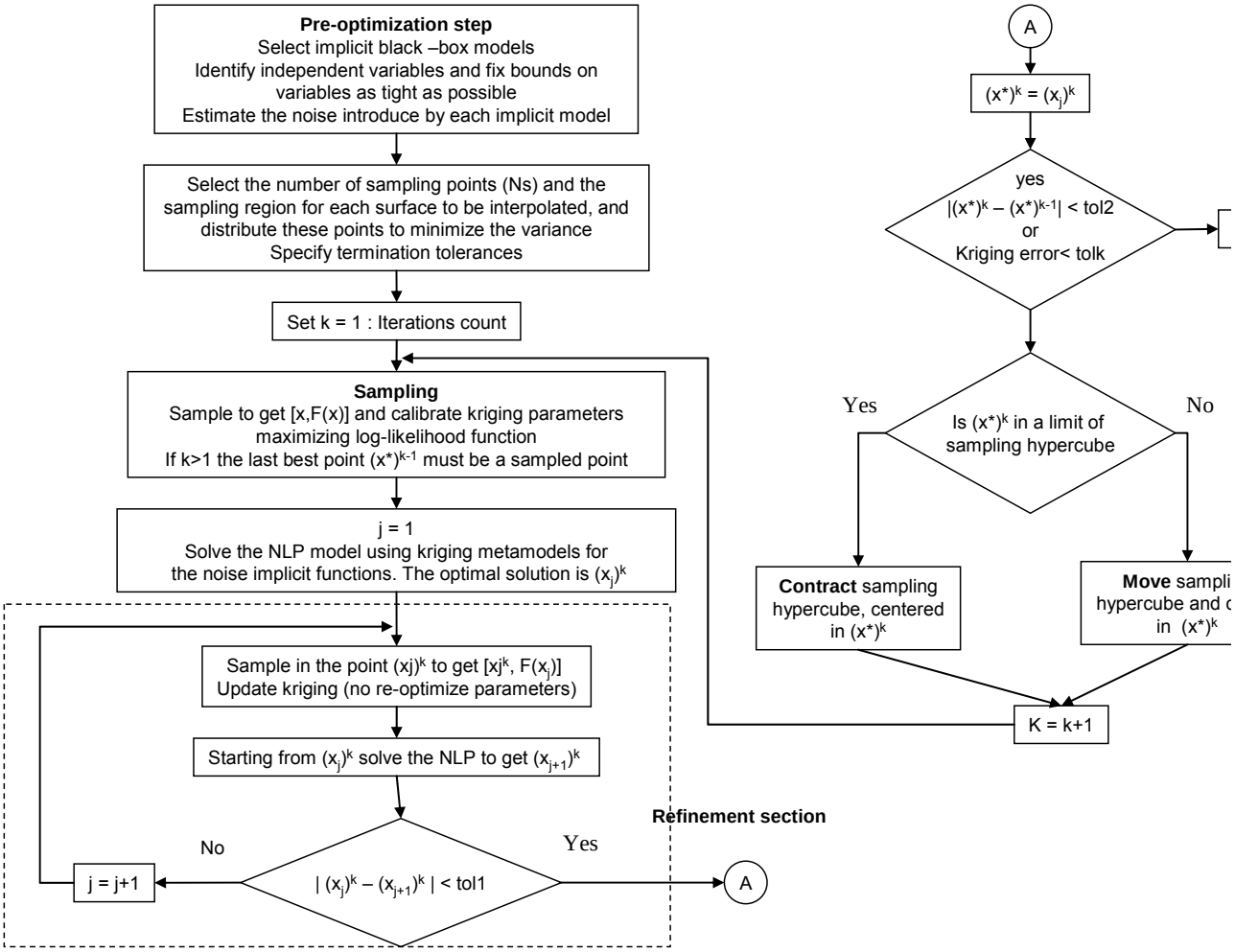

Pre-optimization step
Select implicit black –box models
Identify independent variables and fix bounds on variables as tight as possible
Estimate the noise introduce by each implicit model
Select the number of sampling points (Ns) and the sampling region for each surface to be interpolated, and distribute these points to minimize the variance
Specify termination tolerances
Set k = 1 : Iterations count
Sampling
Sample to get [x,F(x)] and calibrate kriging parameters maximizing log-likelihood function
If k>1 the last best point (x*)k-1 must be a sampled point
j = 1
Solve the NLP model using kriging metamodels for the noise implicit functions. The optimal solution is (xj)k
Sample in the point (xj)k to get [xjk, F(xj)]
Update kriging (no re-optimize parameters)
Starting from (xj)k solve the NLP to get (xj+1)k
Refinement section
No
Yes
j = j+1
| (xj)k – (xj+1)k | < tol1
A
A
(x*)k = (xj)k
yes
|(x*)k – (x*)k-1| < tol2
or
Kriging error< tolk
Yes
No
Is (x*)k in a limit of sampling hypercube
Contract sampling hypercube, centered in (x*)k
Move sampli
hypercube and c
in (x*)k
K = k+1

Figure 4

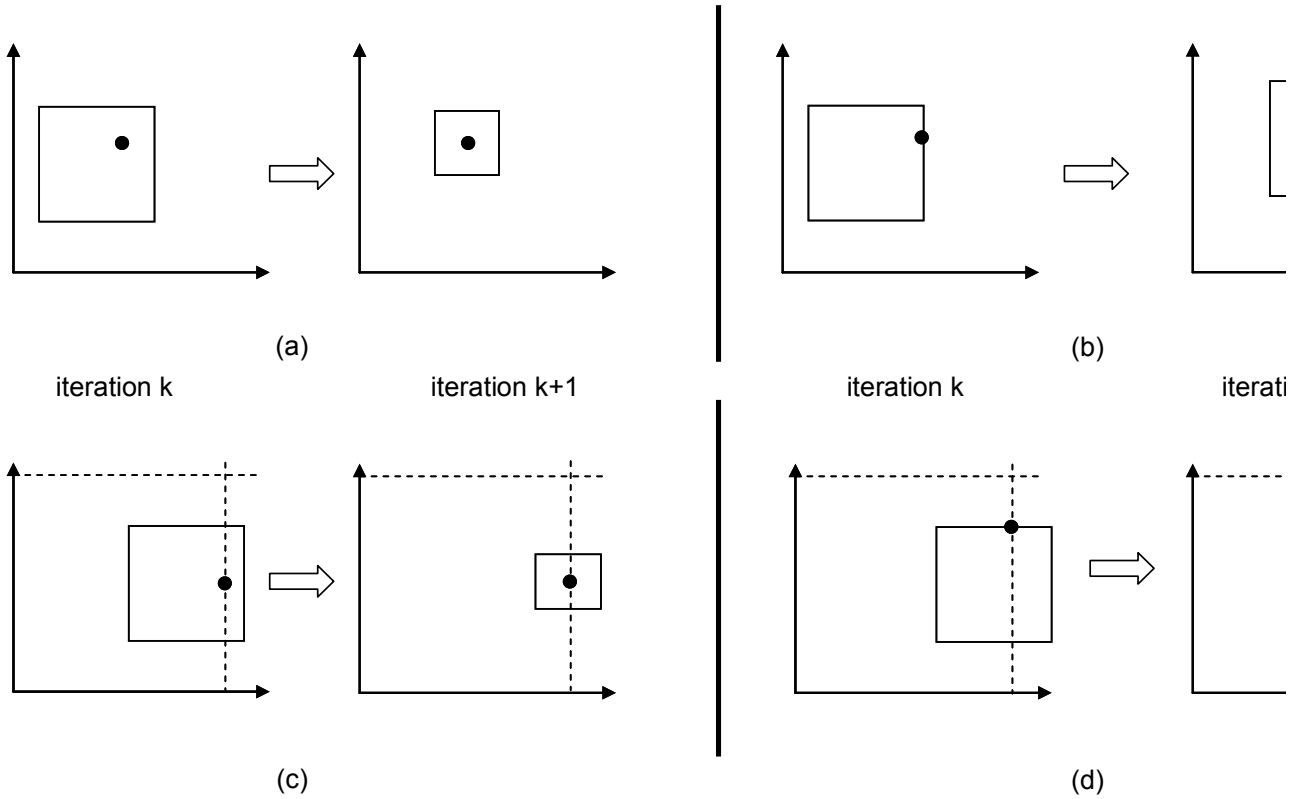


Figure 5

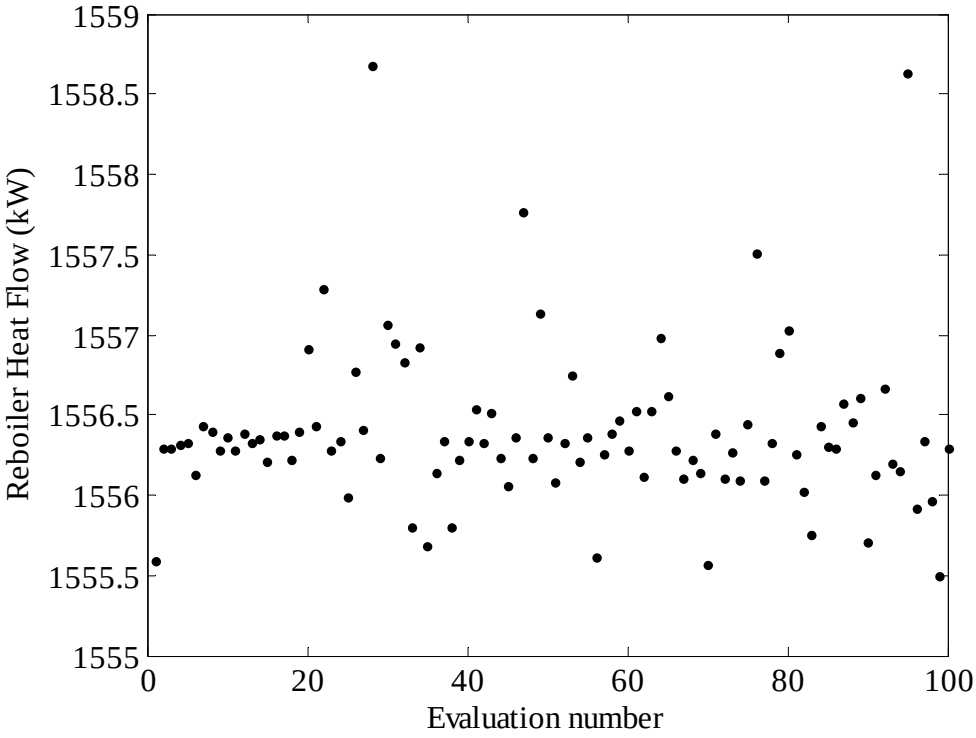
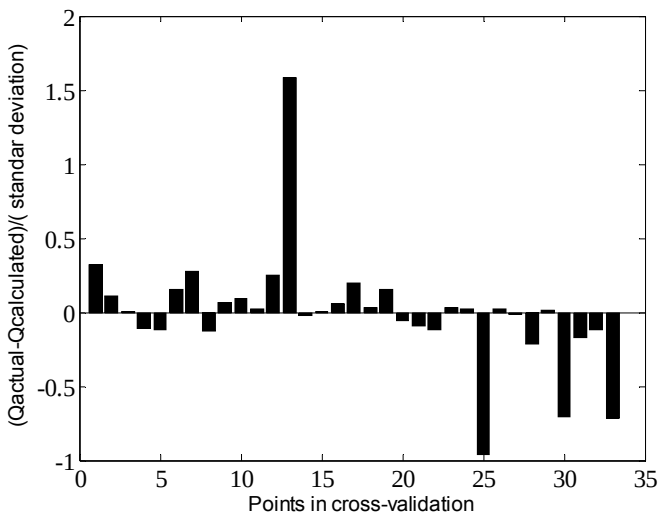
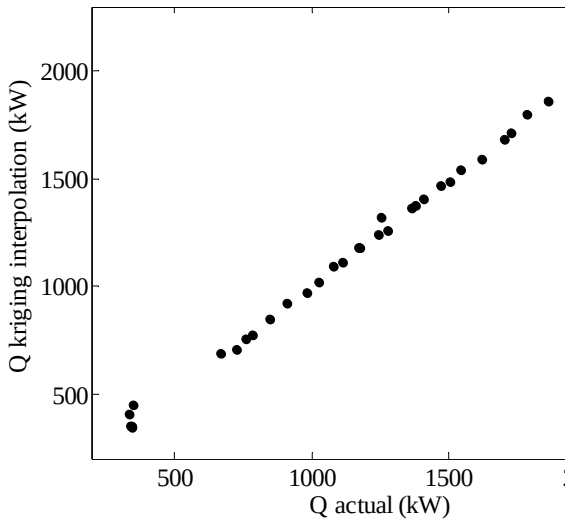

Reboiler Heat Flow (kW)
Evaluation number
1559
1558.5
1558
1557.5
1557
1556.5
1556
1555.5
1555
0
20
40
60
80
100

Figure 6


(Qactual-Qcalculated)/(standar deviation)
Points in cross-validation
2
1.5
1
0.5
0
-0.5
-1
0
5
10
15
20
25
30
35

(a)


Q kriging interpolation (kW)
Q actual (kW)
2000
1500
1000
500
500
1000
1500

(b)

Figure 7

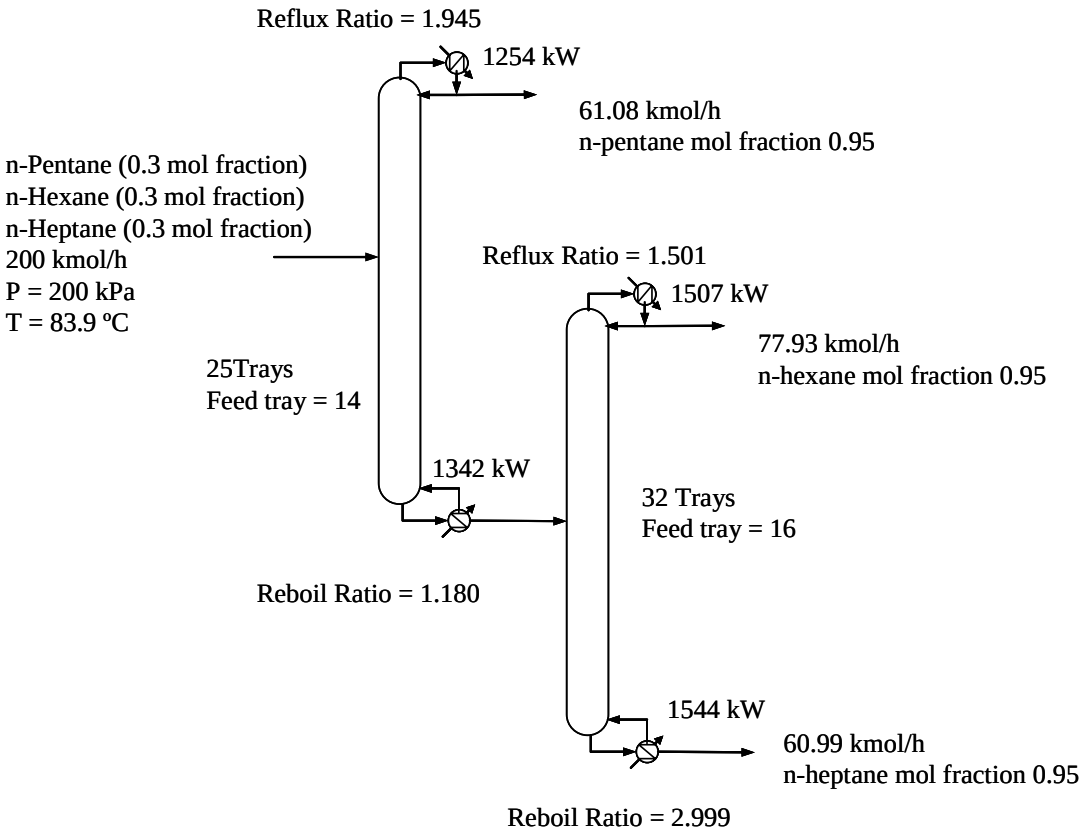


Figure 8

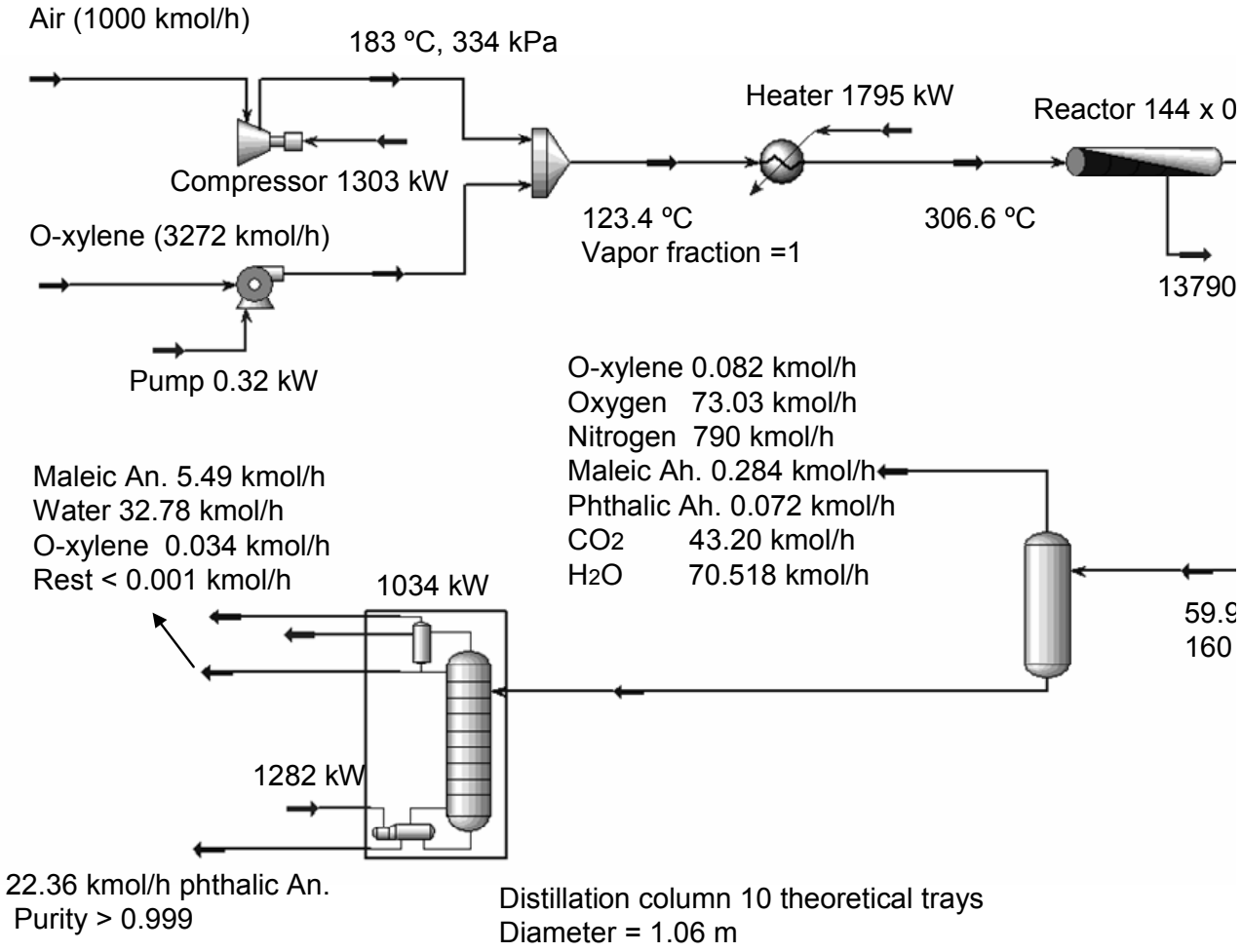

Air (1000 kmol/h)
183 ºC, 334 kPa
Heater 1795 kW
Reactor 144 x 0
Compressor 1303 kW
123.4 ºC
Vapor fraction =1
306.6 ºC
O-xylene (3272 kmol/h)
13790
Pump 0.32 kW
O-xylene 0.082 kmol/h
Oxygen 73.03 kmol/h
Nitrogen 790 kmol/h
Maleic Ah. 0.284 kmol/h
Phthalic Ah. 0.072 kmol/h
CO2 43.20 kmol/h
H2O 70.518 kmol/h
Maleic An. 5.49 kmol/h
Water 32.78 kmol/h
O-xylene 0.034 kmol/h
Rest < 0.001 kmol/h
1034 kW
59.9
160
1282 kW
22.36 kmol/h phthalic An.
Purity > 0.999
Distillation column 10 theoretical trays
Diameter = 1.06 m

Figure 9