Flowsheet Optimization and Optimal Sensitivity Analysis Using Exact Derivatives

D. Wolbert, X. Joulia, B. Koehret, L. Biegler

EDRC 06-160-93

FLOWSHEET OPTIMIZATION AND OPTIMAL SENSITIVITY ANALYSIS USING EXACT DERIVATIVES

D. Wolbert, X. Joulia, B. Koehret
ENSIGC, 18 chemin de la loge
F-31078 Toulouse cedex, FRANCE.

L.T. Biegler
Chemical Engineering Department
Carnegie Mellon University,
Pittsburgh, PA 15213 U.S.A.

Over the past decade, flowsheet optimization has become an important tool for process design. However, for flowsheet optimization the most time consuming step is devoted to the evaluation of the first derivatives, especially from the modular process simulator. To overcome this problem, the PROSIM simulator has been modified in order to generate exact gradient information automatically. Based on the modular input-output sensitivities and a chain-ruling procedure, this strategy allows significant time savings during the optimization. Several optimization methods are implemented (a Reduced Gradient and two Successive Quadratic Programming strategies) and their results are compared in combination with the analytical derivatives. Furthermore, it is often useful to assess the sensitivity of the optimum flowsheet to design parameters (e.g., feed flowrates, constants in kinetic and transport equations) that may be subject to variation and uncertainty. With the calculation of exact derivatives, post-optimality analyses, which were considered too expensive before, can now be implemented with very inexpensive computational costs. The optimal flowsheet sensitivity analysis thus allows us to estimate easily the changes in the optimal process under parametric variations and other process uncertainties.

1. Introduction

Within the past decade, process flowsheet optimization has become a widely used process engineering tool. Spurred by the recent improvements in nonlinear programming algorithms as well as by an infeasible path approach to recycle convergence and optimization, process optimization is now a standard feature within many commercial process simulators.

However, the application of efficient nonlinear programming strategies requires the calculation of accurate derivatives from the process model. Currently, virtually all modular process simulators compute these by finite difference and therefore incur inaccurate derivatives due to truncation errors and convergence noise. These can have a detrimental effect on the performance of the nonlinear programming strategy. A worst case example of this deterioration is given in the next section. Moreover, calculation of derivatives by finite difference represents the most expensive part of a flowsheet optimization; thus there is a clear incentive to improve the efficiency and accuracy of this task.

To address this problem, numerous studies (Volin and Ostrovsky, **1981;** Chen and Stadtherr, 1985; Biegler, 1985; Chan and Prince, 1986) have outlined efficient derivative calculation strategies for modular simulators. Moreover, the use of analytical derivatives in equation-based environments such as SPEEDUP has also been exploited (Panteiides and Sargent, 1985). However, in sequential modular environments, calculation of exact derivatives has proved extremely difficult. In this study we describe how such a calculation has **been adapted** to **flowsheet** optimization within the PROSIM process simulator. An illustration of this implementation is presented in section 3.

With the availability of exact derivatives the efficiency and reliability of an optimization strategy are greatly accelerated and improved. Moreover, evaluating the sensitivity of an optimal flowsheet to changes in external parameters now becomes an easy and inexpensive task. Fiacco (1982) and Ganesh and Biegler (1987) developed efficient algorithms for sensitivity analysis for optimal solutions. Here we develop a reduced space strategy for optimal flowsheet that improves on this previous work and leads to a sensitivity approach that is very inexpensive (and often free) once an optimal flowsheet is obtained. This approach is developed and described in section 4.

The impact of exact derivative calculations within a process simulator for optimization and optimal sensitivity analysis is demonstrated in section 5. Here two process optimization problems are solved and compared with several nonlinear programming strategies, and the effect of exact derivatives is observed for reliability and fast performance. Both of these processes are then analyzed for changes in the optimum solution with respect to changes in external parameters. For both tasks, reductions of effort of up to 75% are observed. Finally, some concluding remarks and directions for future research are presented in section 6.

2. **Optimization Problem and Solution Strategies**

Process flowsheet optimization problems are usually non-linear in the objective function as well as in the constraints. Here we consider the standard nonlinear programming formulation given by:

$$\begin{aligned} &\min_{z} \; <D(z,p) && (1) \\ &\text{s.t.} \quad h(z,p) = 0 \\ &\qquad\; g(z.p) \;\wedge\; o \\ &\qquad\; z^-(p) \le z \le z^+(p) \end{aligned}$$

where O : the objective function
h : set of equality constraint functions
g : set of inequality constraint functions
z : optimization variables
z^-, z^+ : variable bounds
p : set of fixed external parameters

Efficient algorithms have been developed to solve problem (1) for fixed values of p, based on linearizations of the constraints and quadratic approximations to the Lagrangian function. We consider three such optimization algorithms for this study. The first algorithm is based on a reduced gradient method and was developed by Pibouleau et al. (1985). The second algorithm by Biegler and Cuthrell (1985) uses a successive quadratic programming (SQP) approach and the third one (Vasantharajan and Biegler, 1988) is also an SQP based algorithm but with a range and null space decomposition of the constraint space, in order to reduce the size of the Hessian matrix to be evaluated. The major steps of these algorithms are shown in Figure 1. All three codes use the Broyden-Fletcher-Goldfarb-Shanno (BFGS) update formula for the estimation of the Hessian matrix of the Lagrange function.

The flowsheet optimization formulation used to solve problems with recycle streams is based on an *Infeasible Path* approach, where the tear stream equations are included into the optimization problem as equality constraints. This further specializes problem (1) to the following formulation with $h^T = [\, s^T, c^T]$ and $z^T = [\, x^T, y^T]$:

$$
\begin{aligned}
&\underset{z}{\text{Min}}\ O(z.p) && (2)\\
&\text{s.t.}\quad s(z,p) - y - w(x, y, p) - 0 \\
&\qquad\ \ c(z, p) - 0 \\
&\qquad\ \ g(z.p) * 0 \\
&\qquad\ \ z.(p) \le z \le z{+}(p)
\end{aligned}
$$

where	s	: the tear or recycle equation
	c	: design equality constraints
	y	: tear variable, guessed tear stream
	w	: calculated tear stream
	x	: decision variables in process flowsheet

By handling multiple recycle and control loops directly within the optimization step, this approach has been shown to be quite efficient in modular simulators (Chen and Stadtherr, 1985; Biegler and Hughes, 1982, 1985; Lang and Biegler, 1987). However, the performance of these optimization algorithms is often governed by the accuracy of the derivatives supplied from the process model. Frequently, such derivatives need to be calculated by finite difference and this introduces truncation errors as well as convergence noise into the optimization algorithm.

To demonstrate the sensitivity of optimization methods to derivative errors and motivate the work described in the next section, we briefly consider a worst-case example proposed by Carter (1991). Consider a simple unconstrained quadratic problem, Min $z^T Az$, where we have:

$$A = \frac{1}{2}\begin{bmatrix} e 4- 1/e & e - 1/e \\ e — 1/e & e + 1/e \end{bmatrix}, \quad z_0 \; J\frac{2}{2}. \; \mathrm{m}$$

as the matrix and starting point. Note that matrix A is positive definite but the condition number of A equals $1/e^2$. Note that the gradient at zo is given by: V<X>(zo) - VZe/2 [1, 1]^T -e zo and if the approximated gradient contains some error and is given by t(zk) = (VO(zk)-V2e [1, 1]^T) we find that with a Newton method for unconstrained optimization, we have the following directions: djdeal = - A^{-1}VO(zo) • - zo and dactual «= - A'^{1} t(zo) = zo. Note that V<D(zo)^{T}dj<jeal - - zo^{T}Azo < 0 and VO(zo)^{T}d$_a$ctual - zo^{T}Azo > 0, and there is no descent direction for the Newton step generated with the inexact gradient. Further, if we define a **new** point z ' » z o + a dactual. where a is any positive stepsize, we have <J>(z') - O(zo) - (<x^2+2cc)zo^{T}Azo > 0. Consequently, the algorithm fails from a point far from the solution, **even** for small values of e! It is interesting to note that even if we used a "steepest descent" step, d$_{sc}$j = -t(zo) = e zo, the same behavior would be observed for this problem.

Of course, performance of an optimization algorithm is also determined by constraints and variable bounds, but this worst case example shows that gradient error can greatly affect the performance of any derivative-based optimization method. As a result of this behavior, we now consider the calculation of accurate derivatives from a process simulator.

3. Calculation of Exact Derivatives within a Modular Simulator

In order to implement efficient optimization methods, accurate first order derivatives of the optimization problem are required. Chen et al. (1985), Biegler (1985), Kisala et al. (1987) and many other authors dealing with process optimization agree that the computational effort for the evaluation of the derivatives is, by far, the most expensive part of the solution procedure. In order to obtain second derivatives, several authors (Boston et al., 1978; Jirapongphan et al., 1980; Pierucci et al., 1982; Perregard et al., 1992) considered simplified models as an alternative. But such an approximation can lead to a different optimum (Biegler et al., 1985). Since only the rigorous model can guarantee a rigorous optimal solution, Wolbert et al. (1991) developed gradient evaluation techniques for modular process simulators and showed that a complete analytical derivative formulation leads to significant time savings and accuracy, which amply justify the additional implementation effort. For flowsheet optimization on modular simulators, the generation and use of exact sensitivities has often been suggested (Biegier, 1985; Chen and Stadtherr, 1985; Leis et al., 1986; Chan and Prince, 1986), but until now was never fully implemented in a modular process simulator. Thus, in this section we briefly describe the implementation of Wolbert et al. (1991).

As shown in Figure 2, a module in a modular process simulator determines the output (streams, output variables) of a physical unit from the inputs (stream and operating parameters) by solving an algebraic set of equations representing a behavioral model of that unit (e.g., mass and energy balances, phase equilibrium) :

mk (4k .ilk) = 0 (4)

where mk is the model of unit k
^k are the inputs of unit k
η_k are the outputs of unit k

The input-output sensitivities (diik/d^k) of that module can be obtained through analytical differentiation at the solution :

$$\frac{d(m_k)}{\partial(\eta_k)}\,\frac{\partial(\eta_k)}{\partial(\xi_k)}+\frac{\partial(m_k)}{\partial(\xi_k)}=0 \qquad (5)$$

where each term can be evaluated analytically. Moreover, thermodynamic models within each module require special attention. Westerberg et al. (1979) mentioned that roughly 80% of the computation time during a simulation is devoted to the thermodynamic properties evaluation. Koehret (1987) observed results between 50 and 98% depending on the test problem and the thermodynamic model used. Since the derivatives of these models cannot be neglected for rigorous optimization, the efficiency of the methodology presented here is highly dependent on how well the analytical derivatives for thermodynamic models are implemented. As seen from Figure 2 and detailed in Wolbert et al. (1991), derivative and chainruling expressions for function relationships within each module have been implemented on a version of PROSIM and, as an option, are evaluated as part of the process simulation.

Once the module sensitivities are available, the input-output sensitivity can be represented locally through the following equation:

$$\frac{\partial(\eta_k)}{\partial(\xi_k)} = -\left[\frac{\partial(m_k)}{\partial(\eta_k)}\right]^{-1} \frac{\partial(m_k)}{\partial(\xi_k)} \qquad (6)$$

For the sake of clarity, we now present the process sensitivity generation through an example. The process is an adiabatic flash with partial liquid recycle (Fig. 3). The goal is to maximize an arbitrary non-linear function of stream S3 by acting on the flash pressure and the split ratio of stream S4. An inequality constraint is added to ensure that the flash temperature is greater than 270 K. The selected tear stream is S6, which adds 7 equality constraints (the pressure being specified by the pump). The data for this example are given in Table 1. Thermodynamic properties are obtained using the Soave-Redlich-Kwong equation of state.

By arranging optimization variables and process parameters in order of appearance in the calculation sequence, and by formulating all modules as in equations (5) and (6), the flowsheet matrix of the linearized process can be generated using the module sensitivities. The matrix for this example is shown in Figure 4a. Applying a Gaussian elimination of the process variables propagated onto the optimization variables only, the gradient matrix of the objective function and the constraints (black outline in

Figure 4b) is generated. This matrix is then supplied to the optimization algorithms described in Figure 1.

4. Parametric Sensitivity of Optimal Flowsheets

Once problem (2) has been solved, it is often desired to find the sensitivity of the optimal flowsheet to changes in the fixed design parameters, p. Typically, these parameters represent a single design case that is subject to variation or uncertainty. For example, the design parameters may represent transport coefficients, kinetic rate constant and feed flowrates.

To analyze the sensitivity of the optimum solution, Fiacco (1976) smoothness properties as well as a procedure based on expansion of the Karush-Kuhn-Tucker conditions. This was specialized to an efficient finite difference strategy by Ganesh and Biegler (1987) and applied to flowsheet optimization. Here we present a more efficient procedure based on the availability of first derivatives. In particular, we also take advantage of an observation by Kolstad and Lasdon (1990) .regarding the role of active bound constraints.

The Lagrangian for problem (1) can be formulated as follows :

$$L(z, p, X, n, TI) = F(z, p) + X^T h(z,p) + \wedge^T g(z,p) + rw^T (z-z_+(p)) + Ti-T(z-(p)-z)$$

where X, *\L, r\+t r_* are the Karush-Kuhn-Tucker multipliers. At the optimal solution (x*. X*. *\i*,* TI+*. TU*), the first order optimality conditions are satisfied, i.e. all the first derivatives with respect to x, X, ụ, *T*\+ and T|_ are equal to zero. Those conditions must also hold after perturbation of the parameters p. in the case of infinitesimal perturbations, this leads to :

$$\begin{aligned}
&d(V_z LCz^*,p,X,^*+i^*,n^*)) = V\&L^*\ dz + V_z h^*\ dX + V_z g\wedge dji \pm B\ dn + V_{zp}^2 L^*\ dp = 0 \\
&d\left(\nabla_\lambda L(z^*,p,\lambda^*\ \mu^*,\eta^*)\right) = Vjh^*\ dz + Vjh^*\ dp = 0 \\
&d\left(\nabla_\mu L(z^*,p,\lambda^*\ \mu^*,\eta^*)\right) = \nabla_z^T gXdz + V_p^r gXdp = 0 \qquad (7) \\
&d\left(\nabla_\eta L(z^*,p,\lambda^*\ \mu^*,\eta^*)\right) = B^T\ (dz+Vj4dp)=0
\end{aligned}$$

where B is a n x n_{actlve} bounds matrix with B(ij) - 1 if Zj is the j th active bound and g/v is the set of active inequality constraints. These equations can be rearranged to form the following linear system :

$$\begin{bmatrix} V\&L^* & V_z h^* & V_z g_A^* & B \\ V?h^* & 0 & 0 & 0 \\ \nabla_z^T g_A^* & 0 & 0 & 0 \\ B & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \nabla_p^T z^* \\ \nabla_p^T \lambda^* \\ vjn^* \\ \pm\nabla_p^T \eta_\pm^* \end{bmatrix} = - \begin{bmatrix} \nabla_{zp}^2 L^* \\ Vjh^* \\ \nabla_p^T g_A^* \\ -vj4 \end{bmatrix} \qquad (8)$$

For infinitesimal perturbations of the parameters and when strict complementary slackness holds, i.e., gj - 0 => HJ > 0 and $z\backslash$ - $z\backslash_\pm$ • 0 => T|$j_\pm$ > 0, then the following is true:

- the set of active inequality constraints is constant and the active constraints can be seen as additional equality constraints: H* - [h*; g_A*] and A* = [X* ; n*] are vectors with m elements (Fiacco, 1976)

- the variables can be partitioned into free and fixed variables : z - [Zf, zb] and the sensitivities for zb are immediately given by V_pZb $= \nabla_p z_\pm$ (Kolstad and Lasdon, 1990).

System (8) can now be simplified to:

$$\begin{bmatrix} \nabla_{z_f z_f}^2 L^* & V_{Zf}H \\ \nabla_{2f}H^T & 0 \end{bmatrix} \begin{bmatrix} VjzJ \\ VjA^* \end{bmatrix} = - \begin{bmatrix} V2_{f,}pL^* + \nabla_{z_f z_b}^2 L^* \, \nabla_p^T z_b^* \\ V_p H^T + \nabla_{z_b} H^T \, \nabla_p^T z_b^* \end{bmatrix} \qquad (9)$$

Now in order to eliminate the multiplier sensitivities as well as reduce the size of the Hessian that needs to be supplied in (9), we further perform a range and null space decomposition of the equality constraints H* by constructing a nonsingular nt x nf matrix [Z Y], with Z a rif x (nf-m) matrix with columns spanning the null space of $V_{zf}H^{*T}$, i.e., $V_{2f}H^{*T}Z=0$. Here Z and Y are obtained by forming ^{x}a QR factorization of V_{2f}H*T, although for larger systems advantage can also be taken to exploit sparsity of H* (see Vasantharajan and Biegler, 1988). By representing $V_p zf^{*T}$ in terms of range and null space components, dy^T and dz^T, respectively, and multiplying the first row of (9) by Z^T we have:

$$\begin{bmatrix} Z^T\nabla^2_{z_f z_f}L^* Z & Z^T\nabla^2_{z_f z_f}L^* Y \\ V_{Zf}Hl\ Y & 0 \end{bmatrix} \begin{bmatrix} 4 \\ d_Y^T \end{bmatrix} = -\begin{bmatrix} Z^T\left(\nabla^2_{z_f,p}L^* + \nabla^2_{z_f z_b}L^*\ \mathrm{VfzJ}\right) \\ \nabla_p^T H_*^T + \nabla_{z_b}H_*^T\ \nabla_p^T z_b^* \end{bmatrix} \qquad (10)$$

This system is easily solved, first for dy^T (see Figure 4c) and then dz^T with $V_pZf^*T = Z\ d_Z{}^T + Y\ d_Y{}^T$.

To construct (10) with accurate gradient information available from section 3, we also need to consider an efficient strategy to obtain the necessary second derivative information. Note that all of the information to obtain dy^T from the second row in (10) is obtained directly from the gradients at the optimum point and is supplied by ProSim without additional effort. However, Ganesh and Biegler (1988) showed that, while an approximated Hessian is sufficient for optimization purposes, it is not adequate for sensitivity analysis and further numerical perturbations must be performed.

On the other hand, from eqn. (10) it is clear that the *entire* V^2L matrix is not needed and this leads to considerable savings in evaluating second order information. To obtain the reduced Hessian Z^TV^2LZ, system (10) requires only (nf-m) perturbations. Here, instead of perturbing each variable independently, we perturb all the variables along the directions of the orthonormal columns of Z :

$$\left(e_i^T Z^T \nabla^2_{z_f,?}L^*\right)^T = \frac{\left(\nabla_? L(z_f^* + \alpha Z\ q) - V?L^*(z?)\right)}{\alpha} \qquad (11)$$

where the question mark subscript can be replaced by Zf, z^ or p; a is a coefficient automatically tuned to keep the perturbation size within a given range and ej is a zero (nf-m) vector except for a 1 in its i^{tn} element. In addition, the $Z^TV^2LYdY^T$ and $Z^TV^2LV_pzb^T$ terms are directly obtained from (11) by first premultiplying by the appropriate vector and transposition. Note also that dy^T is easily obtained from (10) through:

$$d ? - [v_{Zf}HT\ Y]"^{1} (VJHT + V_{Zb}Hl\ VjfcJ) \qquad (12)$$

In addition, as a byproduct, the direct calculation of Z^TV^2LZ in (11) allows a check for satisfaction of the second order Kuhn-Tucker conditions. This condition is seldom considered by current nonlinear programming algorithms. Here if the reduced Hessian matrix is not positive semidefinite, the solution is not locally optimal. Fortunately, by restarting the problem along a direction of negative curvature (e.g., shifting z along the eigenvector corresponding to one of the negative eigenvalues) will lead to convergence to a better (and probably locally optimal) solution. An example of checking second order conditions as well as the restart procedure is detailed in Wolbert (1992).

System (10) leads to first order parametric sensitivities, i.e., changes in the minimizer and objective function for infinitesimal perturbations of p. In the case of finite parameter variations, on the other hand, system (10) may be insufficient because of the influence of higher order terms as well as changes in the active constraint sets or bounds. These problems can be tackled by solving the following quadratic programming problem, with Z spanning the null space of $(V_zh^*)^T$, and Y an orthonormal basis for V_zh \ First, if we ignore higher order terms, but consider finite parameter perturbations with the possibility of active set changes, we can solve a QP for each parameter parameter variation. For a parameter pj , we have:

$$\text{Min } [(zT\ V2_{Pi}L^*\ A_{Pi} + zT\ v£,_{x}L^*\ Y\ djfdj + \frac{1}{2} d_Z(Z^T\nabla_{ZZ}L^*\ Z)d_Z^T]$$
$$\text{s.L} \quad Vpjg^*Api + (Vjg^*\ Z)dj + (V?g^*\ Y)d\$\ £\ 0 \qquad (13)$$
$$\nabla_{p_i}z\text{-}\Delta p_i \le (Z\ d_Z^T + Y\ d_Y^T) \le \nabla_{p_i}z\text{+}\Delta p_i$$

with $d\$=-[v_zhTY]\sim^{1}Vp.hTAp_i$ and Azi= Z dz + Y dy for parameter pj. Otherwise, including the effect of all higher order terms is difficult; frequently the only alternative is to re-solve problem (1). To address higher order variations in p, however, requires only a slight modification of (13). Here we re-evaluate the objective and constraint functions at (z*. p + Ap) and solve:

$$\text{Min } [(VJ\overline{L^*}\ Z + dylYTVxxL^*\ z))d£ +_{2} L\&\ dz\{zTv_{xx}L^*\ z)d£]$$

$$\text{s.t} \quad \bar{g}^{*"} + (vjg^*\ Z)\ dJ + (Vjg^*\ Y)\ d\$\ £\ 0 \qquad (14)$$

$$(\overline{z_-} - z^*) \leq (5\ Z\ d\overset{T}{z} + Y\ d \quad \leq (\overline{z_+} - z^*)$$

where Az « $Z\ dz + Y\ dY$ and $d\$ = -\ (v_z hT\ Y)"^{1}\ \overline{h^*}$,

$$\overline{L^*} = L(z^*, p + \Delta p, \lambda^*, \mu^*, \eta^*)$$
$$\bar{h} = h(z^*, p + \Delta p)$$
$$\bar{g}^* = g(z^*, p + \Delta p)$$
$$\overline{z_\pm} = z_\pm(p + Ap)$$

and Z and Y have the same definition as for (13). Note from (14) that solution of this quadratic program will yield the *exact* optimal solution for parameters p+Ap for a problem that is separable in p and z, is linearly constrained and has a quadratic objective in z, and is *arbitrarily nonlinear in* p.

Finally, the results of the quadratic programming problems (10) must be analyzed with **great care. The sensitivities are often one-sided values and,** when a combined parametric sensitivity is estimated, the consistency and satisfaction of the constraints has to be checked. This can usually be done by analyzing the Kuhn-Tucker coefficients and their sensitivity. From these sensitivities or variations of the optimization parameters, the sensitivities and variations of any other (objective or constraint) functions of the process, *v.* can be deduced simply by using the derivatives :

$$V_p i>^* = V_p i> + V_z v\ VpZ^* \quad \text{or} \quad to^*m \quad V_p » Ap + \nabla_z v\ \Delta z^* \qquad (15)$$

or evaluating the function directly at the predicted point.

$$Av\ ^* = v(z^* + Az,\ p + Ap) - v(z\backslash\ p) \qquad (16)$$

5. Example Problems

Adiabatic flash with liquid recycle

We used this example to present the analytical derivative generation, the flowsheet was shown in Figure 3 and the main characteristics given in Table 1. The thermodynamic model is the Soave.Redlich and Kwong equation of state. The three optimization algorithms have been used to solve this problem with and without analytical derivatives. The number of model linearizations (NML), the number of flowsheet evaluations (NFE) and the relative time (normalized to the Reduced Gradient case with numerical perturbations) are given for each case in Table 2.

The reduced gradient method appears to be the slowest algorithm, regardless of the approach used to generate the derivatives. This is mainly due to the fact that an exact line search is done by golden section search. While robust and reliable, this method can be quite expensive in terms of function evaluations. The Reduced SQP algorithm is slightly faster than the Full SQP algorithm when numerical perturbations are used, but with analytical derivatives their computation times are equivalent. The Full SQP method needs more iterations than reduced SQP, but the latter needs to reduce its stepsize more often to satisfy the Armijo inequality for the line search.

The use of analytical derivatives leads to important time savings for all of the optimization methods. These savings are however different for each method (48% for the Reduced Gradient method, 73% for the Full SQP and 66% for the Reduced SQP), since they are only realized over the fraction of time that gradients are required.

Considering now the optimum sensitivity analysis, we first evaluate the computation times (equivalent number of function evaluations, ENFE) of a full Hessian approach (Fiacco, 1976), a reduced approach proposed by Ganesh and Biegler (1987) and the reduced approach with analytical derivatives (Wolbert et al., 1992). We consider here, for a finite parameter variation, that all the feed's partial flow rates and the upper bound of the split ratio are simultaneously increased by a given percentage. The resulting ENFE are shown in Figure 5. The reduced approach results in an average of 75% savings when compared to a full Hessian approach. The use of analytical derivatives leads to an additional 75% savings for the reduced approach; this sums up to an average of 94% time savings between a full approach with numerical perturbations and a reduced approach with analytical derivatives.

The reduced Hessian for this problem is a positive scalar equal to 0.06097. Thus second order conditions are satisfied and further results of the post-optimality analysis can be seen in Figures 6 and 7. Figure 6 shows the evolution of the actual, optimal objective function value and its approximations obtained by three techniques: an extrapolation from the optimum sensitivities (10), from the finite variations (14, 15) and by using the results for the optimization variables of the finite variation study to run a new flowsheet evaluation (14, 16). Figure 8 gives a better idea about the accuracy of the different approximations through the relative error of each estimation. These results show the efficiency but also the limitations of such methods. Quite good estimates can be obtained for a low cost. However, since these methods are based on a one step linearization technique, their accuracy is highly dependent on the extrapolation size, the nonlinearities taken into account during re-solution or estimation, and also on the nonlinearities in the model.

Ammonia synthesis plant

The flowsheet of this chemical process is shown in Figure 8 and its characteristics are given in Table 3. An economic objective function is maximized subject to 8 equality (tear) constraints and 3 inequalities; the tear stream equations have to be satisfied, the preheater (C5 - C6) has to compensate the energy consumption inside the reactor, a purity of at least 99 % ammonia must be in stream C16, less than 3.4 mol/s of ammonia must be lost in stream C12 and the pressure drop of the high pressure flash is at least 0.4 MPa. Sixteen optimization variables have been considered (see Table 3) and the Soave-Redlich-Kwong equation of state has again been used for this example. The three optimization algorithms have also been compared to solve this problem with and without analytical derivatives. The number of model linearizations (NML), the number of flowsheet evaluations (NFE) and the relative time are given for each case in Table 4 (normalized to the Reduced Gradient method with numerical perturbations).

The results are similar to those obtained for the first example, and so are the reasons for these results. In particular,

- the Reduced Gradient method is always slower.
- the Reduced SQP is faster than Full SQP for numerical perturbations, and about the same when analytical derivatives are used.
- important time savings are observed due to analytical derivatives
  42 % Reduced Gradient
  74 % Full SQP
  74 % Reduced SQP

For the optimum sensitivity analysis, we note that the reduced Hessian is a fourth order matrix with eigenvalues of [2.8657E-4, 8.3561 E-10, 1.8335E-4, 7.7208E-5], which satisfy second order optimality. Note also that the extremely low second eigenvalue indicates the presence of nonunique optima. Again we consider a simultaneous variation of the feed flowrates as well as the upper bound for the reactor conversion. The changes in the true optimum and its approximations are shown in Figure 19, while the relative error of these approximations are presented in Figure 10. The approximations are in good agreement with re-evaluated optimal values of the objective function. Moreover, the accuracy, for a given parameter perturbation size, is even better than for the first example.

6. Conclusions

The methodology developed for the generation of analytical derivatives in a simultaneous modular simulator leads to significant time savings for process optimization and post-optimality analysis. Computation time has been reduced by up to 75% for SQP based algorithms. Moreover, the Reduced Gradient code was less efficient than SQP and did not take as much advantage of the analytical derivative feature. This is mainly due to an expensive exact line search step. While the Reduced SQP performed better than the Full SQP code when numerical perturbations are used, with analytical derivatives both codes led to similar computation times. In fact, the Reduced SQP method needed fewer iterations to converge but here full steps in the line search are not accepted as often as with the Full SQP method. Therefore it sometimes requires one or two additional flowsheet evaluations per iteration.

For the post optimality analyses, optimum sensitivity analysis and finite parametric variation analysis, appear to perform accurately and efficiently. Combining model projection techniques and fast derivative generation, these approaches can be done at reasonable cost and provide very useful information about the optimal solution. The analysis of the results should however be done with caution; for large parameter variations ("large" being problem dependent), the linear or quadratic approximations of the constraints and the Lagrangian, respectively, may no longer be valid.

The advantages of analytical derivatives are not only based on their low cost as much better accuracy is also observed and module convergence "noise" during perturbations and perturbation size selection no longer affect gradient values. Two applications of our methodology have been presented here, but many other applications can take advantage of this feature. Simulation problems, process sensitivity studies and control loop generation using static gain matrices are some methods that have already been used with analytical derivatives. Other methods (MINLP, flexibility analysis, process parameter identification, ...) are also being considered. In addition, this methodology has already been extended to distillation columns and plug flow reactor models with similar results.

For future work, a key aspect will be the automatic generation of exact derivatives from *existing codes* for process models. To accomplish this, a number of such derivative generating codes are currently available or under development (see Griewank and Corliss, 1991, for a survey). In general, these tools parallel the model's calculation sequence with a derivative calculation that applies the chainrule to the sequence and handles the bookkeeping to keep track of intermediate values. Such strategies are preprocessors or coprocessors that use the model's FORTRAN source code directly and generate derivative FORTRAN code that is run along with the model. Examples of these include:

- JAKE-F, which has seen many applications but is limited to a subset of FORTRAN (Hillstrom, 1982)
- DAPRE, which has been developed for use with the NAG library (Pryce and Davis, 1987)
- ADOL-C, which is implemented very efficiently using operator overloading features of C++ (Griewank et al, 1990)
- ADIFOR, the most recent development (Bischof et al, 1992), which uses a source transformation approach within the ParaScope environment (Callahan et al, 1988). This environment is used for dependency analysis among the variables as well as parsing the FORTRAN code .

In Bischof et al. (1992), for example, a FORTRAN adiabatic flash block was processed and Jacobian matrices were calculated by an ADIFOR-generated derivative code, for all outputs with respect to "all inputs. No changes were required in the original model and the total time for evaluating both the flash block and its Jacobian was only twice that of evaluating the function. A number of similar examples were solved in other disciplines with up to a 70-fold increases in performance. Future development and application of ADIFOR as well as other tools will deal with calculation of higher derivatives, undoubtedly for use in optimization algorithms as well as nonlinear analysis. These developments, coupled with the benefits of exact derivatives demonstrated in this study will yield tremendous performance improvements for process optimization.

References

Biegler L.T. (1985). "Improved Infeasible Path Optimization for Sequential Modular Simulators-I : The interface." , *Comp. Chem. Eng.* , 9_ , 3 , 247 - 256.

Biegler L.T. ; J.E. Cuthrell (1985). "Improved infeasible path optimization for sequential modular simulators-II: the optimization algorithm." *Comp. & Chem. Eng.*, £ , 3 , 257 - 267.

Biegler L.T., I.E. Grossmann, A.W. Westerberg (1985). "A note on approximation techniques used for process optimization." *Comp. & Chem. Eng.* , a , 2 , 201 - 206

Bischof, C, A. Carle, G. Corliss, A. Griewank and P. Hovland, "ADIFOR Generating Derivative Codes for FORTRAN Programs," Preprint MCS-P263-

0991, Argonne National Laboratory (1992)

Boston J.F., H.I. Britt (1978). "A radical formulation and solution of single stage flash problems." *Comp. & Chem. Eng.,* 2. , 109 -122.

Chan W.K., R.G.H. Prince (1986) "Application of the chain rule of differentiation to sequential modular flowsheet optimization." *Comp. & Chem. Eng.*, 1Q. , 3 , 223 - 240.

Callahan, D., K. Cooper, R. T. Hood, K. Kennedy and L M. Torczon, "ParaScope: A parallel programming environment," *Intl. J. of Supercomputer Applications," 2,* p. 4 (1988)

Carter, R. G., "A Worst Case Example Using Linesearch Methods for Numerical Optimization with Inexact Gradient Evaluations," Argonne National Laboratory, Preprint MCS-P283-1291 (1991)

Chen H.-S., M.A. Stadtherr (1985). "A Simultaneous Modular Approach to Process Flowsheeting and Optimization : Part I : Theory and implementation. Part II : Performances on Simulation Problems. Part III : Performances on Optimization Problems." *A.I.Ch.E. J.* ,3_L , 11 , 1843 - 1881.

Dennis J.R., J.J. More (1977). Quasi-newton methods , motivation and theory," *S.I.A.M. Review*, 12., 1 ,46-89.

Ganesh N., L.T. Biegler (1987). "A Reduced Hessian Strategy for Sensitivity Analysis of Optimal Flowsheets. *A.I.Ch.E. J.*, 22., 2 , 282 - 296.

Griewank, A., and G. F. Corliss (eds.), **Automatic Differentiation of Algorithms: Theory, Implementation and Application,** SIAM, Philadelphia (1991)

Griewank, A., D. Juedes, J. Srinivasan and C. Tyner, "ADOL-C, a package for the automatic differentiation of algorithms written in C/C++," *ACM Trans. Math. Soft,* to appear (1990)

Hillstrom, K. E, "JAKEF-a portable symbolic differentiator of functions given by algorithms," Technical Report ANL-82-48, Argonne National Laboratory (1982)

Jirapongphan S. ; J.F Boston. ; H.I. Britt ; L.B. Evans (1980). "A non linear simultaneous modular algorithm for process flowsheet optimization." *80 th A.I.Ch.E. Meeting, Chicago*

Kisala T. P. ; P.A. Trevino-Lozano ; J.F. Boston ; H.I. Britt ; LB. Evans (1987). "Sequential modular and simultaneous modular strategies for process flowsheet optimization,"Comp. *& Chem. Eng.*, **1 1** , 6 , 567 - 579.

Koehret B. (1987). Conception d'une simulateur de proc6d6s. *Thèse de Doctoral d'Etat, I.N.P.T.*

Kolstad, C. D. and L. S. Lasdon, "Derivative Evaluation and Computational Experience with Large Bilevel Mathematical Programs," *JOTA,* £5., 3, p. 485 (1990)

Lang Y.D. ; L.T. Biegler (1987). "A unified algorithm for flowsheet optimization." *Comp. & Chem. Eng.* **, 1 1** , 2 ,**143 - 158.**

Leis J.R., S.A. Gallagher, M.A. Kramer (1987). "Parametric Sensitivity Analysis of Complex Process Flowsheets Using Sequential Modular Simulation." *Comp. Chem. Eng.* **11** , 4, 409 - 421.

Perregard ; Sorensen (1992). "Simulation and optimization of chemical processes : numerical and computational aspects." *Proceedings of the ESCAPE 1 conference,* Helsingor, Danemark, *Supplement to Comp. & Chem. Eng. ,* 1& S247 - S254

Pibouleau L. ; **P.** Floquet ; S. Domenech **(1985). "Optimisation de** procédés chimiques par une méthode de gradient r£duit : Partie 1 : Presentation de Palgorithme." *R.A.I.R.O. Recherche operationnelle / Operation research ,* 19 , 3 , 247 - 274

Pierucci S.J. ; E.M. Ranzi ; G.E. Biardi (1982). "Solution of recycle problems in a sequential modular approach." *AJ.Ch.E. J.,* **2fi.**, 5 , pp 820 - 827

Pryce, J. D., and P. H. Davis, "A new implementation of automatic differentiation for use with numerical software," Tech. Report AM-87-11, School of Mathematics, University of Bristol, UK (1987)

Vasantharajan S., L.T. Biegler (1988). "Large-Scale Decomposition for Successive Quadratic Programming." *Comp. Chem.* Eng. ,12., 11 , 1087-1101

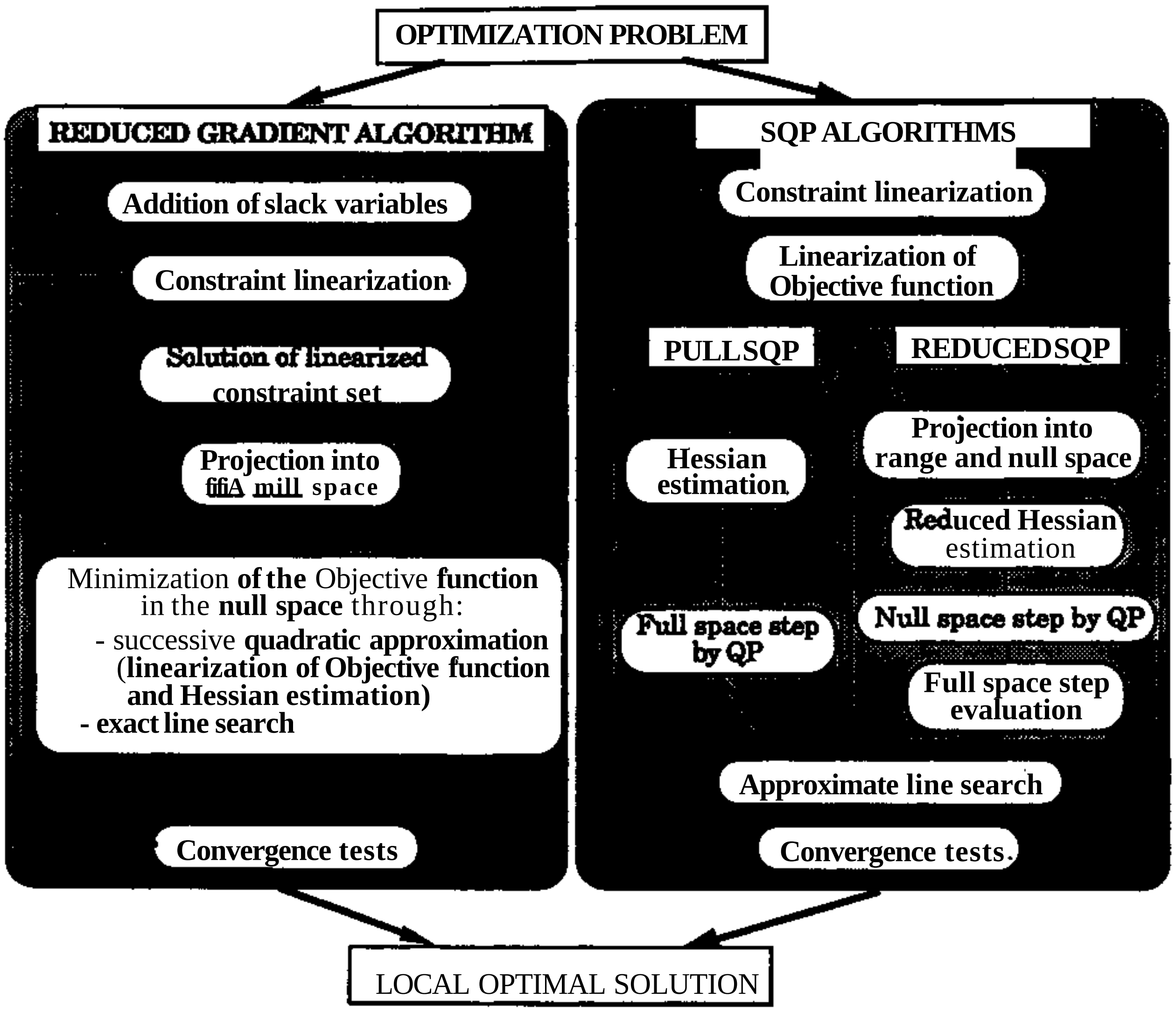
Volin, Y. M., and G. M. Ostrovskiy, "Sensitivity Calculation Methods for Complex Chemical Systems. 1: Algorithmization." *Comp. Chem. Eng., 5,* 1, p. 21-30 (1981)

Westerberg A.W., H.P. Hutchison, R.L. Motard, P. Winter (1979). Process flowsheeting, Cambridge University Press

Wolbert D., X. Joulia, B. Koehret, M. Ports (1991). "Analyse de sensibility pour l'optimisation des procédés chimiques." 3ième Congrès de Génie des Proc&tes, Compiègne, France. In : *R6cents Progrès en G6nie des ProcédSs, £*, 13, 415-420.

Wolbert D., X. Joulia, B. Koehret, L.T. Biegler (1992). "Optimal flowsheet sensitivity in a sensitivity oriented environment."ESCiAPE *2 proceedings,* Toulouse, France. In : *Supplement to Comp. & Chem. Eng.* l i , pp S117 - S122.

Wolbert, D., Analyse de Sensibilite Optimization dans un Simulateur de Procedes Modulaire Simultane, *Thdse de doctorat, I.N.P.T.* (1992)


OPTIMIZATION PROBLEM
REDUCED GRADIENT ALGORITHM
Addition of slack variables
Constraint linearization
Solution of linearized constraint set
Projection into fifiA mill space
Minimization of the Objective function in the null space through:
- successive quadratic approximation (linearization of Objective function and Hessian estimation)
- exact line search
Convergence tests
SQP ALGORITHMS
Constraint linearization
Linearization of Objective function
PULL SQP
REDUCED SQP
Hessian estimation
Projection into range and null space
Reduced Hessian estimation
Full space step by QP
Null space step by QP
Full space step evaluation
Approximate line search
Convergence tests
LOCAL OPTIMAL SOLUTION

Eigure 1: Optimization algorithms for NLP problems

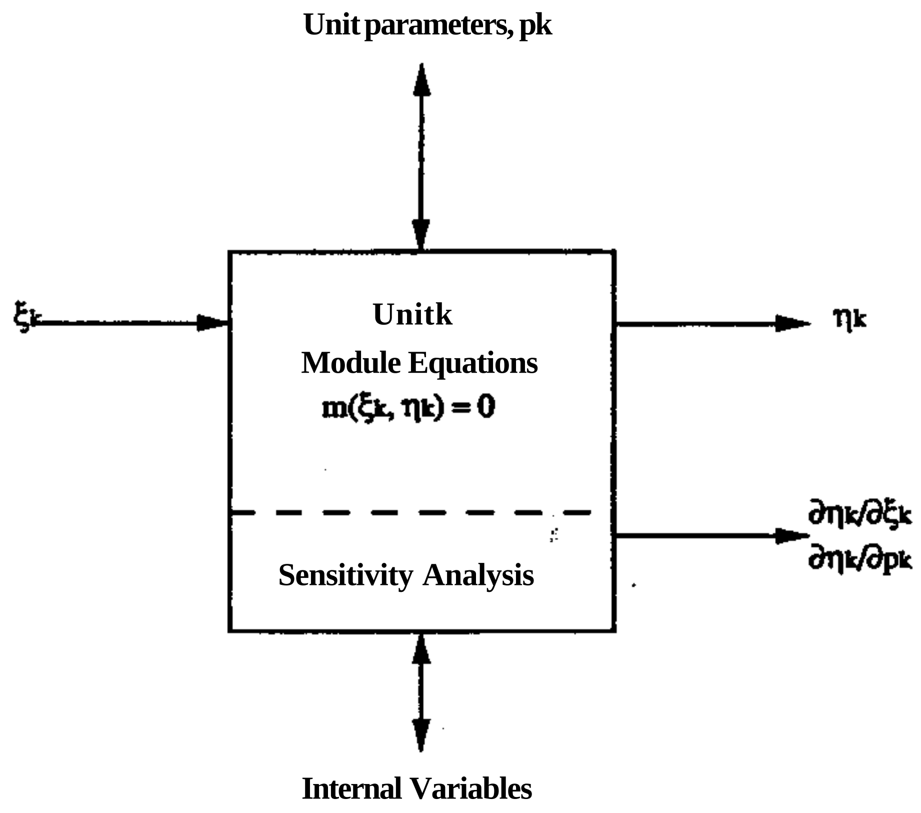
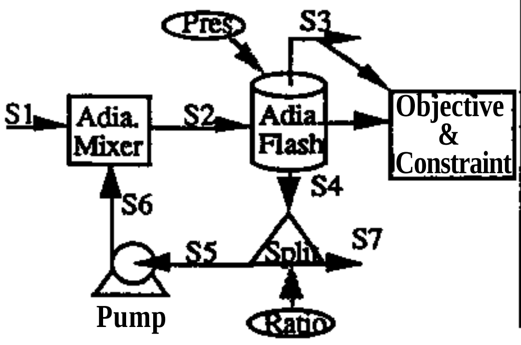


Figure 2: Structure of modules for sensitivity analysis

$$\mathrm{Max}(s3_1^{*}*S3_2 - S3_1^2 - S3_3^3 + S3_4 - \sqrt{S3_5})$$



		Initial values	Solution	Feed
Tear Stream variables (S6)	Propane (kmol/fr	5.682	1.239	10
	1-Butene (")	10.13	2.881	15
	Butane (")	13.73	4.006	20
	trans-2-Butene (Ir	13.78	4.059	20
	ds-2-ButeneO	13.88	4.133	20
	PentaneO	7.301	2.355	10
	Temperatuie(K)	297.3	277.0	310.927
	Diessurefbar)	10.073	*in nrx*	10.073
Optimization variables	Pres. Flash (bar)	2.722	1.315	
	Splitratio	0.5	OR	
Criteria		-1.57113	2.14620	

-^ Active bound

Figure 3: Adiabatic flash with liquid recycle

Table 1: Data and results for the adiabatic flash problem

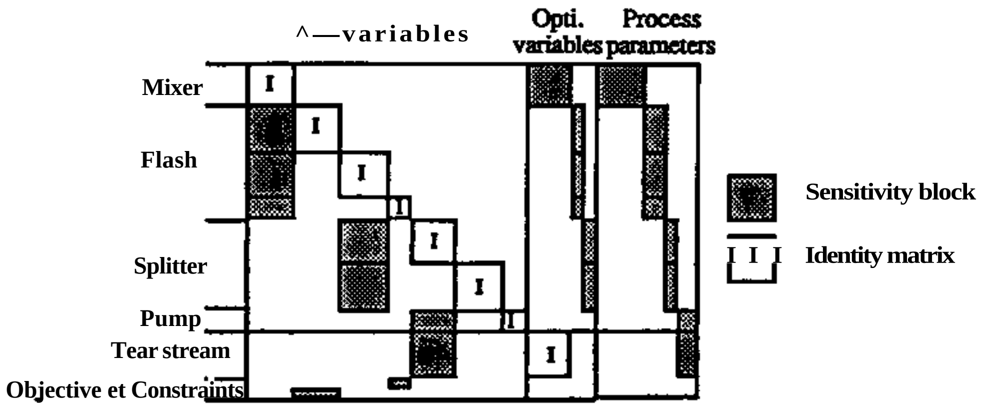


Figure 4a: Incidence matrix for the flash problem

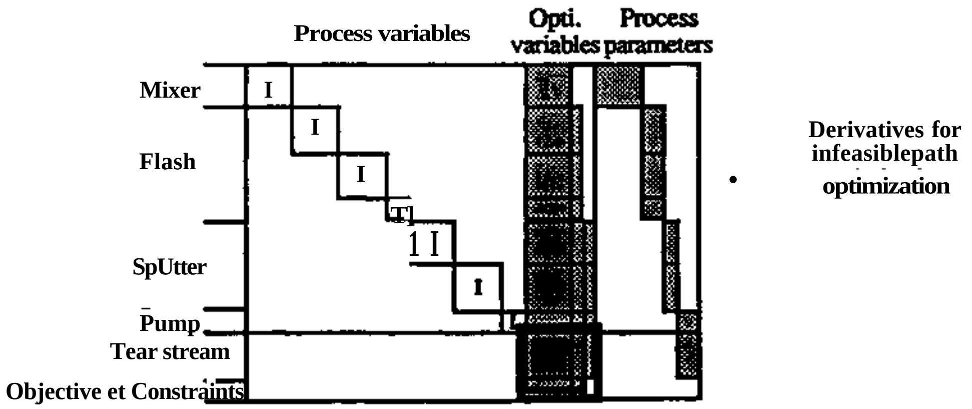


Figure 4b: Generation of the optimization derivatives

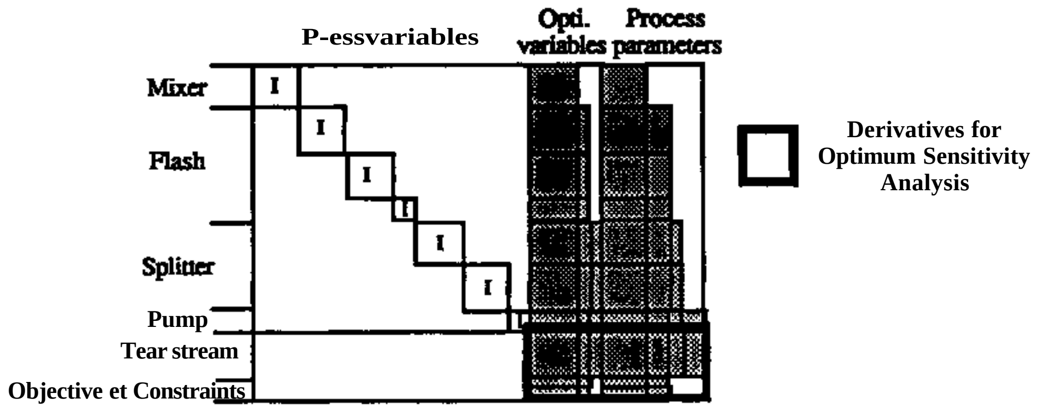
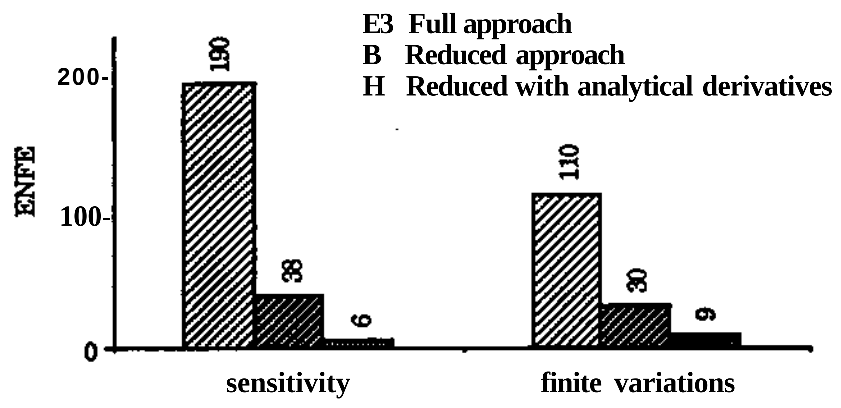


Figure 4c: Generation of the optimum sensitivity analysis derivatives



Eigure 5: Cost comparison of the different approaches for optimum analysis.

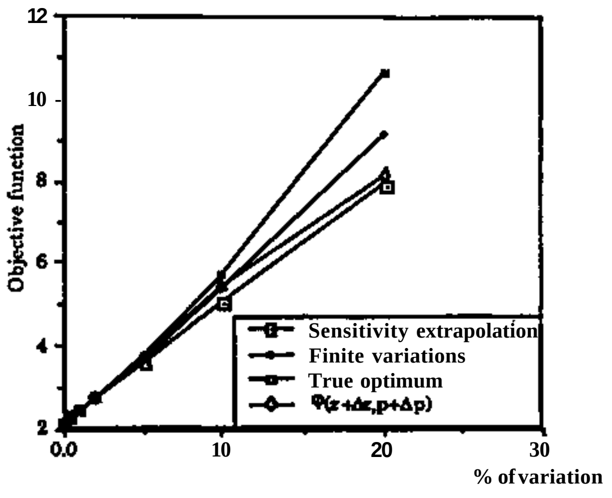


Figure 6 : True optimum and estimation as a function of perturbation size.

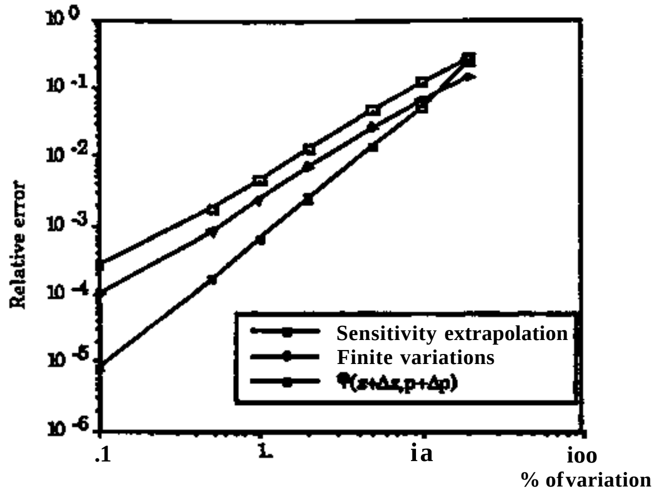
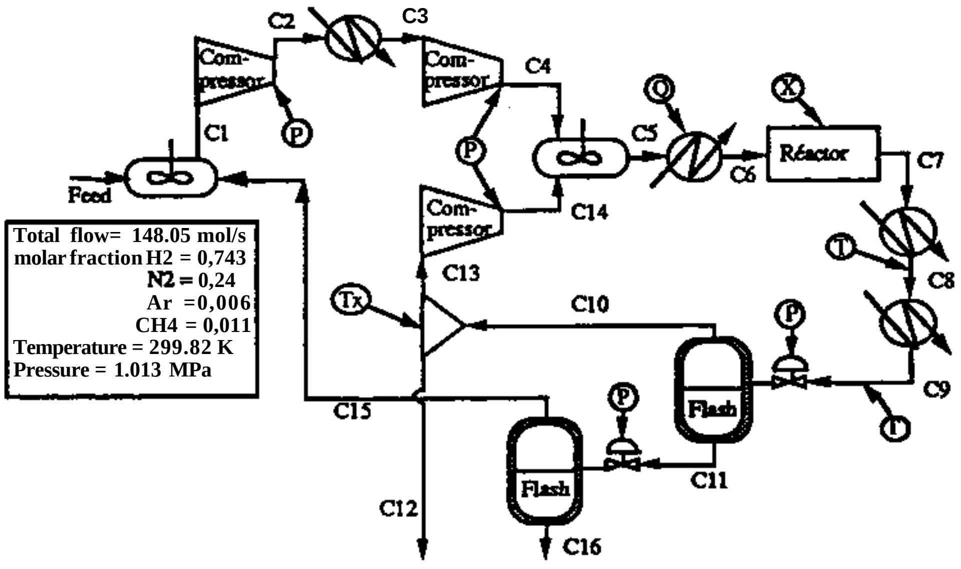


Figure 7: Relative error of the optimum estimates.



Total flow= 148.05 mol/s
molar fraction H2 = 0,743
N2 = 0,24
Ar =0,006
CH4 = 0,011
Temperature = 299.82 K
Pressure = 1.013 MPa

Figure 8: Flowsheet of the ammonia synthesis plant.

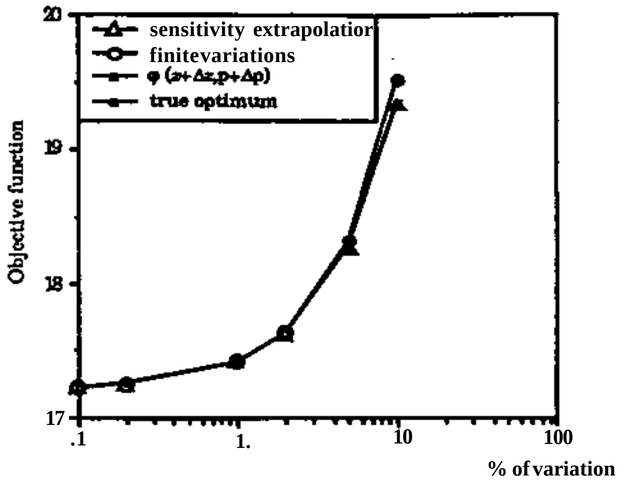


Figure 9: True optimum and its estimations as a function of variation size.

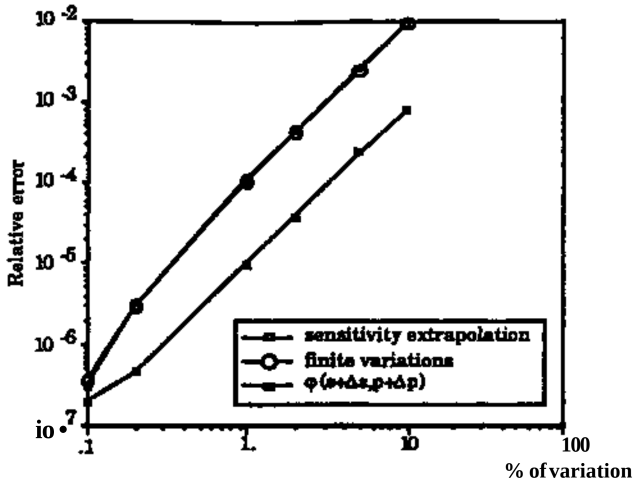


Figure 10: Relative error of the optimum estimates.

Derivative generation	Optimization methods	NML	NFE	Relative time
Numerical perturbations	Reduced Gradient	20	321	LOO
	FullSQP	23	228	0,72
	Reduced SQP	16	170	0.53
Analytical derivatives	Reduced Gradient	20	141	0.52
	FullSQP	*22*	25	0.19
	Reduced SQP	16	26	0.18

NML: Number of model linearizations
NFE: Number of flowsheet evaluations

Table 2: Optimization results **for the adiabatic flash problem.**

	initialisation	Lo. bounds	[Jp. bounds	Solution	
Pressure between com. (MPa)	4.413	4.137	6.895	4.869	
Output pressure comp. (MPa)	20.68	19.99	29.72	29.72	Up.b.
preheater duty (MW)	3.456	L46	5.86	3-538	
conversion ratio	0.4100	035	0.45	0.4500	Up.b.
Temp, output 1st exch. (K)	299.8	295.9	310.9	295.9	Lab.
Temp, output 2nd exch. (K)	242.0	233.1	333.1	2441	
Pressure 1st flash (MPa)	L241	1.014	6.895	2.402	
Pressure 2nd flash (MPa)	19.99	19.65	29.41	2931	
Splitter ratio	0.1000	0.05	0.12	0.5261E-O1	
<u>Tear stream</u>					
Partial flow H2 (mol/s)	168.2	0.0	1000	18&5	
Partial flow N2	44.70	0.0	1000	40.77	
Partial flow Ar	6352	0.0	1000	1424	
Partial flow CH4	14.47	0.0	1000	22J99	
Partial flow NH3	64.76	0.0	1000	7023	
Temperature (K)	242.0	100	1000	244.1	
Objective function	14.27			1721	

Table 3: Data and results for the ammonia synthesis problem

Derivative generation	Optimization methods	NML	NFE	Relative time
Numerical perturbations	Reduced Gradient	87	2092	1.00
	FullSQP	49	839	0.40
	Reduced SQP	44	760	0.36
Analytical derivatives	Reduced Gradient	103	927	0.58
	FullSQP	49	56	0.10
	Reduced SQP	*46*	51	0.09

NML: Number of model linearizations
NFE: Number of flowsheet evaluations

Table 4: Optimization results **for the ammonia** synthesis problem