Computer-Aided Design of Redundant Sensor Networks

Georges Heyen[1] , Marie-Noëlle Dumont[1], Boris Kalitventzeff[2]
1 : Laboratoire d'Analyse et Synthèse des Systèmes Chimiques, Université de Liège,
Sart Tilman B6A, B-4000 Liège (Belgium), email G.Heyen@ulg.ac.be
2 : BELSIM s.a., rue Georges Berotte 29A, B 4470 Saint-Georges-sur-Meuse, Belgium

Abstract

A systematic method to design sensor networks able to identify key process parameters with a required precision at a minimal cost is presented. The procedure is based on a linearised model, derived automatically from a rigorous non-linear data reconciliation model. A genetic algorithm is used to select the sensor types and locations.

1. Problem position

The application of data reconciliation to plant monitoring in now considered as standard practice. Redundant measurements allow reducing the uncertainty due to random errors. Unmeasured parameters can be estimated safely from reconciled state variables.

However little has been published on the design of measurement systems allowing to achieve a prescribed accuracy in the estimates of some key process parameters, and to secure enough redundancy to make the monitoring process resilient with respect to sensor failures. Madron (1972) solves the linear mass balance case using a graph-oriented method. Bagajewicz (1997) analyses the problem for mass balance networks, where all constraint equations are linear.

In this study, we propose a general mathematical formulation of the sensor selection and location problem, in order to reduce the cost of the measurement system while providing estimates of all specified key process parameters within a prescribed accuracy. The goal is to extend the capability of previously published algorithms, and to address a broader problem, not being restricted to flow measurements and linear constraints.

In the optimisation problem formulation, the major contribution to the objective function is the annualised operating cost of the measurement system. The set of constraint equations is obtained by linearising the process model at the nominal operating conditions, assuming steady state. The process model is complemented with link equations, that relate the state variables to any accepted measurements, or to key process parameters whose values should be estimated from the set of measurements. In our case, the set of state variables for process streams comprises all stream temperatures, pressures and partial molar flow rates. In order to handle total flow rate measurements, the link equation describing the mass flow rate as the sum of all partial molar flow rates weighted by the component’s molar mass has to be defined. Similarly, link equations relating the molar or mass fractions to the partial molar flow rates have also to be added for any stream where an analytical sensor can be located.

Link equations have also to be added to express key process parameters, such as heat transfer coefficients, reaction extents or compressor efficiencies.

In the proposed approach, we will assume that all variables are measured; those that are actually unmeasured will be handled as measured variables with a large standard deviation. Thus data reconciliation, taking linearised constraints into account, requires the solution of the following optimisation problem:

$$\min_{X} (X-X')^{T} W (X-X') \qquad (1)$$

s.t. $A X + D = 0$

where X is the array of process variables (size m), X’ is the set of measured values and $W=\mathrm{diag}(1/\sigma_i^2)$ is the weight matrix (the diagonal terms of the inverse of the measurement covariance matrix). The linear approximation of the constraints is easily obtained from the solution of the non-linear model, since A is the Jacobian matrix of the non-linear model evaluated at the solution. The constrained problem is transformed into an unconstrained one using Lagrange formulation :

$$\min_{X,\lambda} L = (X-X')^{T} W (X-X') + 2 \lambda^{T} (A X + D) \qquad (2)$$

Solving for stationarity conditions:

$$\begin{bmatrix} X \\ \lambda \end{bmatrix} = \begin{bmatrix} W & A^{T} \\ A & 0 \end{bmatrix}^{-1} \begin{bmatrix} WX' \\ -D \end{bmatrix} = \mathbf{M}^{-1} \begin{bmatrix} WX' \\ -D \end{bmatrix} \qquad (3)$$

Matrix M can easily be built, knowing the variance of measured variables appearing in sub matrix W, and the model Jacobian matrix A (which is constant). This matrix will be modified when assigning sensors to variables. Any diagonal element of matrix W will remain zero as long as a sensor is not assigned to the corresponding process variable; it will be computed from the sensor precision and the variable value when a sensor is assigned, as shown later in section 2.3. In fact we are not interested in the solution of system (3), since measured values X’ are not known. The variance of reconciled values X is related to the variance of measurements X’ as shown in Heyen et al. (1996):

$$\mathrm{var}(X_i) = \sum_{j=1}^{m} \frac{\left(\left[M^{-1}\right]_{ij}\right)^2}{\mathrm{var}(X'_j)} \qquad (4)$$

The elements of $\mathbf{M}^{-1}$ are obtained by calculating a LU factorisation of matrix M. In case matrix M is singular, we can conclude that the measurement set has to be rejected, since it does not allow observing all the variables. Row i of M^{-1} is obtained by back substitution using the LU factors, using a right hand side vector whose components are δ_{ij} (Kronecker factor: $\delta_{ij}=1$ when i=j, $\delta_{ij}=0$ otherwise).

In the summation of equation (4), we only take into account variables X'_j that have been assigned a sensor, the variance of unmeasured variables being set to infinity.

2. Algorithm description

Solution of the sensor network problem is carried out in 7 steps:

1. Process model formulation and definition of link equations.
2. Model solution for the nominal operating conditions and model linearisation.
3. Specification of the sensor database and related costs.
4. Specification of the precision requirements for observed variables.
5. Verifications of problem feasibility.
6. Optimisation of the sensor network.
7. Report generation.

Each of the steps is described in details before presenting a test case.

2.1 Process model formulation and definition of link equations.

This is easily done in the Vali 3 data reconciliation software, which is used as the basis for this work (Belsim 2001). The model is formulated by drawing a flowsheet using icons representing the common unit operations, and linking them with material and energy streams. Physical and thermodynamic properties are selected from a range of physical property models. Any acceptable measurement of a quantity that is not a state variable (T, P, partial molar flow rate) requires the definition of an extra variable and the associated link equation, what is done automatically for standard measurement types (e.g. mass or volume flow rate, density, dew point, molar or mass fractions, etc). Similarly, extra variables and link equations must be defined for any process parameter to be assessed from the plant measurements. A proper choice of extra variables is important, since we may note that many state variables cannot be measured in practice (e.g. no device exists to directly measure a partial molar flow rate or an enthalpy flow).

In order to allow the model solution, enough variables need to be set by assigning them values corresponding to the nominal operating conditions. The set of specified variables must match at least the degrees of freedom of the model, but overspecifications are allowed, since a least square solution will be obtained by the data reconciliation algorithm.

2.2 Model solution for the nominal operating conditions and model linearisation.

The data reconciliation problem will be solved, either using a large-scale SQP solver, or the Lagrange multiplier approach. When the solution is found, the value of all state variables and extra variables is available, and the sensitivity analysis is carried out (Heyen et al, 1996). A dump file is generated, containing all variable values, and the non-zero coefficients of the Jacobian matrix of the model and link equations. All variables are identified by a unique name indicating its type (e.g. S32.T is the temperature of stream S32, E102.K is the overall heat transfer coefficient of heat exchanger E102, S32.MFH2O is the molar fraction of component H2O in stream S32).

2.3 Specification of the sensor database and related costs.

A data file must be prepared that defines for each sensor type the following parameters:

- the sensor name
- the annualised cost of operating such a sensor
- parameters a_i and b_i of the equation allowing to estimate the sensor accuracy σ_i from the measured value x_i', according to the relation: $\sigma_i = a_i + b_i x_i'$
- a character string pattern to match the name of any process variable that can be measured by the given sensor (e.g. a chromatograph will match any mole fraction, thus will have the pattern MF*, while an oxygen analyser will be characterized by the pattern MFO2)

2.4 Specification of the precision requirements for observed variables.

A data file must be prepared that defines the following information for the selected key performance indicators or for any process variable to be assessed:

- the composite variable name (stream or unit name + parameter name)
- the required standard deviation σ_i^t, either as an absolute value, or as a percentage of the measured value.

2.5 Verification of problem feasibility.

Before attempting to optimise the sensor network, the programme first checks the existence of a solution. It solves the linearised data reconciliation problem assuming all possible sensors have been implemented. In case several sensors are available for a given variable, the most precise one is adopted. This provides also an upper limit C_{max} for the cost of the sensor network.

A feasible solution is found when two conditions are met:

- the problem matrix **M** is not singular;
- the standard deviation σ_i of all selected reconciled variables is lower than the specified value σ_i^t.

When the second condition is not met, several options can be examined:

- extend the choice of sensors available in the sensor definition file by adding more precise instruments;
- extend the choice of sensors by allowing measurement of other variable types;
- modify the process definition by adding extra variables and link equations, allowing more variables besides state variables to be measured.

2.6 Optimisation of the sensor network.

Knowing that a feasible solution exists, we can start a search for a lower cost configuration. The optimisation problem as posed involves a large number of binary variables (in the order of number of streams * number of sensor types). The objective function is multimodal for most problems. However identifying sets of suboptimal solutions is of interest, since other criteria besides cost might influence the selection process. Since the problem is highly combinatorial and not differentiable, we attempted to solve it using a genetic algorithm (Goldberg, 1986). The implementation we adopted is based on the freeware code developed by Carroll (1998). The selection scheme used involves tournament selection with a shuffling technique for choosing random pairs for mating. The evolution algorithm includes jump mutation, creep mutation, and the option for single-point or uniform crossover.

The sensor selection is represented by a long string (gene) of binary decision variables (chromosomes); in the problem analysis phase, all possible sensor allocations are identified by finding matches between variable names (see section 2.2) and sensor definition strings (see section 2.3). A decision variable is added each time a match is found. Multiple sensors with different performance and cost can be assigned to the same process variable.

The initial gene population is generated randomly. Since we know from the number of variables and the number of constraint equations what is the number of degrees of freedom of the problem, we can bias the initial sensor population by fixing a rather high probability of selection (typically 80%) for each sensor. We found however that this parameter is not critical. The initial population count does not appear to be critical either. Problems with a few hundred binary variables were solved by following the evolution of populations of 10 to 40 genes, 20 being our most frequent choice.

Each time a population has been generated, the fitness of its members has to be evaluated. For each gene representing a sensor assignment, we can estimate the cost C of the network, by summing the individual costs of all selected sensors. We also have to build the corresponding matrix M (equation 3) and factorise it. In an initial version of the code, we used SUBROUTINE MA29AD from Harwell library (1984), aimed to factorise a general symmetric matrix. However later experience with larger test problems indicated that significant computer time could be saved by taking advantage of the sparsity of M matrix. Indeed using Belsim's sparse matrix code reduced the computer time by a factor of 25 for a problem involving 312 variables.

The standard deviation σ_i of all process variables is then estimated using equation 4.

This allows calculating a penalty function P that takes into account the uncertainty affecting all observed variables:

$$P = \sum_{i=1}^{m} P_i \qquad \text{where} \qquad P_i = \frac{\sigma_i}{\sigma_i^t} \quad \text{when } \sigma_i \le \sigma_i^t \tag{5}$$

$$P_i = 0.01 \min\left(10, \frac{\sigma_i}{\sigma_i^t}\right)^2 \quad \text{when } \sigma_i > \sigma_i^t \tag{6}$$

The fitness function F of the population is then evaluated as follows:

- if matrix M is singular, return $F = -C_{max}$
- otherwise return $F = -(C + P)$

Penalty function (5) increases (slightly) the merit of sensor network that perform better than specified. Penalty function (6) penalises genes that do not meet the specified accuracy, but it does not reject them totally, since some of their chromosomes might code interesting sensor sub networks.

The population is submitted to evolution according to the mating, crossover and mutation strategy. Care is taken that the current best gene is always kept in the population, and is duplicated in case it should be submitted to mutation. After a specified number of generations, the value of the best member of the population is monitored. When no improvement is detected for a number of generations, the current best gene is accepted as a solution. There is no guarantee that this solution is an optimal one, but it is feasible and (much) better than the initial one.

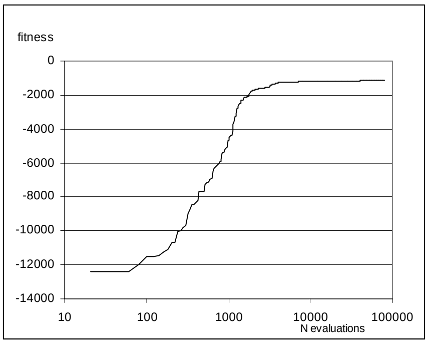
2.7 Report generation.

The program reports the best obtained configurations, as a list of sensors assigned to process variables to be measured. The predicted standard deviation for all process variables is also reported, as well as a comparison between the achieved and target accuracies for all key process parameters.

3. An example

As an example we tried to design a sensor network for an ammonia synthesis loop. The process involves a 5-component mixture (N_2, H_2, NH_3, CH_4, Ar), 13 units, 19 process streams, 10 utility streams (cooling water, refrigerant, boiler feed water and steam). The units are: 2-stage compressor with 2 intercoolers, recycle mixer, recycle compressor, reactor preheater, ammonia converter, waste heat boiler, water cooled condenser, ammonia cooled condenser, vapour-liquid separator, purge divider and flash drum for expanded ammonia condensate.

The model involves 181 variables and 131 constraint equations. Accuracy targets are specified for 45 variables. It includes extra measurable variables (molar fractions) and some unit parameters to be monitored (e.g. heat exchange transfer coefficient, compressor efficiency, extent of reaction, departure from equilibrium),



The sensor database allows choosing between sensors of different accuracy and cost, namely 2 temperature sensors, 3 flow meters, 2 composition analysers and 2 types of pressure gauges. The program detects that up to 166 sensors could be installed. Thus the solution space involves 2^{166} =9.3 10^{49} solutions (most of them being unfeasible).

We let the search algorithm generate 4000 populations, which required 80000 function evaluations and 776 CPU seconds (PC with 1.33 GHz AMD Athlon processor, program compiled with Compaq Fortran compiler, only local optimisation). The initial cost function was C_{max}=17340 cost units with all sensors selected. Optimisation brought it down to 1110 cost units. We may note that a solution within 60 cost units of the final one was attained after 5700 evaluations (in less then one minute).

4. Conclusions and future work

The available software could be improved by allowing more flexibility in the sensor definition (e.g. defining acceptable application ranges for each sensor type), or by implementing different objective functions besides the cost. Possible objectives could address the resiliency of the sensor network to equipment failures, or the capability to detect gross errors, in the line proposed by Bagajewicz (2001).

There is no guarantee that this solution found with the proposed method is an optimal one, but it is feasible and (much) better than the initial one. Thus we claim that this algorithm contributes to the rational design of sensor networks.

References

Bagajewicz M.J., 1997, chapter 6 in “Process Plant Instrumentation: Design and Upgrade”, Technomic Publishing Company.

Bagajewicz M.J., 2001, Design and Retrofit of Sensor Networks in Process Plants, AIChE J., 43(9), 2300-2306

Belsim., 2001, VALI 3 User’s Guide, Belsim s.a, B 4470 Saint-Georges-sur-Meuse, Belgium

Carroll D.L., 1998, FORTRAN Genetic Algorithm Driver version 1.7, download from <http://www.staff.uiuc.edu/~carroll/ga.html>

Goldberg, D.E.,1989,"Genetic Algorithms in Search, Optimization and Machine Learning," Addison-Wesley

Heyen G. , Maréchal E. , Kalitventzeff B., 1996, Sensitivity Calculations and Variance Analysis in Plant Measurement Reconciliation, Computers and Chemical Engineering, vol. 20S, 539-544

Madron F., 1992, section 6.3 in “Process Plant Performance Measurement and Data Processing for Optimization and Retrofits”, Ellis Horwood