Brazilian Journal of Chemical Engineering

ISSN 0104-6632
Printed in Brazil
www.abeq.org.br/bjche

Vol. 31, No. 04, pp. 993 - 1002, October - December, 2014
dx.doi.org/10.1590/0104-6632.20140314s00002632

FITTING EQUATION OF STATE PARAMETERS IN PARALLEL COMPUTERS

M. Castier[1*], R. F. Checoni[1] and A. Zuber[1,2]

[1]Chemical Engineering Program, Texas A&M University at Qatar, PO Box 23874, Doha, Qatar.
Phone: +974-44230534
E-mail: marcelo.castier@qatar.tamu.edu
[2]Departamento de Engenharia Química, Universidade Estadual de Maringá, Av. Colombo 5790, 87020-900, Maringá - PR, Brazil.

(Submitted: April 1, 2013 ; Revised: November 15, 2013 ; Accepted: February 23, 2014)

Abstract - This work compares two strategies to fit parameters of equations of state in parallel computers, emphasizing solutions that require few changes to existing sequential programs. One strategy uses the conventional Nelder-Mead algorithm coupled with parallel objective function evaluation (SSPO). The other strategy uses a parallel Nelder-Mead algorithm coupled with sequential objective function evaluation (PSSO). The PSSO strategy, which executes parallel one-dimensional searches during each iteration, is simpler to implement and converged to parameter sets with objective functions smaller than those obtained by the SSPO strategy. The SSPO strategy produced speedups consistent with the number of processes used and is more suitable when many processors are available. Both strategies are potentially useful and choosing between them is a matter of convenience, depending on the problem at hand. With parallel computers increasingly available, the easy implementation and convenience of these two strategies should appeal to developers and users of thermodynamic models.
Keywords: Equations of state; Parallel; Message passing interface; Parameter fitting.

INTRODUCTION

Not long ago, parallel computers were associated with fairly large machines, most of them running Linux, dedicated to special computations that need extreme performance, such as quantum mechanical calculations, computational fluid dynamics, molecular simulations, and chemical process optimization, among other applications relevant to chemical industries. Early examples of such work include the development of parallel solvers for large scale linear and nonlinear systems in chemical process simulations (Paloschi, 1997, 1998; Scott, 2001, 2001a), dynamic simulations of chemical processes (Abdel-Jabbar *et al*., 1999; Leineweber *et al*., 2003), and an assessment of parallel computing paradigms (Brochard, 1998). Examples of papers on applied optimization include: the solution of process engineering problems that lead to non-convex mixed integer nonlinear optimization problems (Smith and Pantelides, 1997, 1999; Schilling and Pantelides, 1999); the use of a parallel implementation of the discrete element method (Bertrand *et al*., 2004), applying the message passing interface (MPI) to shorten the computer time needed to simulate the packing of spherical particles as a model of the consolidation of paper coating structures; the development of a framework (Siirola *et al*., 2003) to integrate several optimization algorithms into a cooperative method that outperforms its individual

*To whom correspondence should be addressed components in terms of locating the global minimum of complicated functions.

An application specific to thermodynamics has been the development of a procedure to balance the loads in multiprocessor computers when fitting the parameters of the Wilson excess Gibbs energy model from vapor-liquid equilibrium data (Gau and Stadtherr, 2002). This procedure uses interval analysis, which guarantees that the global minimum of the objective function is found within a preset search range. Explicit references to the use of parallel computing to fit equation of state (EOS) parameters could not be found in the literature. Although fitting EOS parameters is a seemingly routine problem, it is at least as complicated as fitting parameters of excess Gibbs energy models (Olaya *et al*., 2008, Staudt *et al*., 2013, and Soares and Gerber, 2013) and poses several challenges, which are often overlooked. Those relevant to this work are discussed in the next paragraphs.

A first challenge has direct relation with the massive number of experimental data points for several physical properties, such as vapor pressures, densities, enthalpies of vaporization (and phase compositions, in the case of mixtures) in EOS parameter fitting. This is especially true when developing group contribution methods, in which a few group parameters are intended to represent the physical properties of many substances and/or mixtures, fitted from hundreds or thousands of experimental data points. This demands large computational effort and time-consuming calculations, often carried out using a single processor in a personal computer (PC). However, modern desktop, laptop, and tablet computers and even mobile phones have multiple processors. Recent Windows versions of software such as Mathematica and Matlab exploit such configurations to speed up calculations in PCs. Nonetheless, many legacy sequential codes exist in languages such as Fortran and C. Adapting them for maximum performance in parallel computers may require extensive reprogramming, but substantial performance gains are possible in certain applications with few changes to existing codes. This paper shows that EOS parameter fitting is one of such applications and compares two strategies for utilizing the Nelder-Mead simplex algorithm in parallel computers, usable both in single multiprocessor PCs and in cluster supercomputers. In one of the strategies, parallelization takes place at the level of the optimization method, leaving the codes for evaluating the objective function and the thermodynamic properties unchanged. In the second strategy, parallelization takes place at the level of the objective function evaluation, leaving the rest of the code unchanged. Both strategies share the feature that there is no need to modify the code that evaluates the thermodynamic properties, which can be rather long depending on the EOS used.

A second challenge is related to multiple minima in objective functions, each given by a different parameter set. Ways of dealing with this include running local optimization algorithms several times from different initial estimates or using global optimization methods (Gau *et al*., 2000; Dominguez *et al*., 2002). These multiple minima often have similar objective functions, although with rather different EOS parameter values. Sometimes, model developers wish to correlate experimental data as well as possible, meaning the global minimum of the objective function is the desirable target. However, it is often preferable to find a local minimum whose parameters are similar to those found previously for related substances or mixtures. This is useful when trying to develop generalized correlations for EOS parameters or assessing trends in parameter values and model performance. Therefore, interestingly, finding the global minimum of the objective function may not be the most meaningful or desirable target in some situations.

A third challenge is the mathematical complexity of modern EOSs, which makes it cumbersome to obtain analytical derivatives of thermodynamic properties with respect to their adjustable parameters. Therefore, direct optimization methods, which do not use derivative information, are convenient, but have the drawback of slow numerical convergence. The Nelder-Mead simplex algorithm (Nelder and Mead, 1965), in original or modified form, is one such method and is widely applied in EOS parameter fitting. There are several examples of its use in recent work (Lai *et al*., 2009; Santos *et al*., 2010; Justo-García *et al*., 2010; Llano-Restrepo and Muñoz-Muñoz, 2011; Cho *et al*., 2012; Schmid and Gmehling, 2012) and in many older references. Also, there is abundant evidence in the literature of its common application to fit the parameters of excess Gibbs energy models, in recent (Jana *et al*., 2012; Ingram *et al*., 2012; Mesquita *et al*., 2012; Almeida *et al*., 2012) and older work. In summary, despite having been developed in the 1960s, the Nelder-Mead simplex optimization method remains a very popular choice for fitting the parameters of thermodynamic models.

The next section describes the two parallel strategies for parameter fitting that address these challenges. This discussion is followed by a section on results and discussion. The paper ends with the conclusions drawn from this research.

PARALLELIZATION STRATEGIES

We compare two strategies that do not require rewriting legacy sequential implementations of EOSs or writing new parallel codes for them: sequential simplex with parallel objective function (SSPO) and parallel simplex with sequential objective function (PSSO). Our implementation uses Fortran and MPI. At the user level, MPI provides functions (in C) and subroutines (in Fortran) to control parallel computations and pass information from one process to another. In Windows, we used the freely available MS MPI (HPC Pack 2008 R2 MS-MPI Redistributable Package with Service Pack 3), from Microsoft. MPI was designed for distributed-memory systems instead of shared-memory systems. However, it performed well in single multicore PCs, as the examples of this paper show. In addition, using MPI has the advantage of allowing the easy transfer of code to large cluster computers. In fact, once developed and tested in Windows with MS MPI, the program was ported to a Linux-based computer cluster with the Intel MPI library, with no need to change its code. Alternative approaches, not adopted, would include using programming environments such as Cactus (Goodale *et al*., 2003) or TAO (Munson *et al*., 2012).

Sequential Simplex with Parallel Objective Function (SSPO)

In the SSPO strategy, a master process runs the conventional Nelder-Mead simplex algorithm. Each slave process reads an exclusive portion of the experimental data points, not read by any of the other slave processes. Experimental data sets are generally organized by systems, which may contain different physical properties and number of components and, consequently, different computational needs for their modeling. To balance the load, when reading the data set, each data point is directed to the next available slave process. After the data input, all slave processes have nearly identical numbers of data points (at most, differing by one data point). Also, this approach to allocating data points to processes is unlikely to overload some of the slave processes with systems that are more difficult or have more chemical components than others. With these precautions, all slave processes are bound to have similar computational loads. When the simplex algorithm in the master process needs the value of the objective function, each slave process works on its portion of the data points, passing its contribution to the objective function. The master node adds these contributions to determine the value of the objective function. Little inter-process communication is needed to pass the current estimates of the parameters values from the master to the slave processes and, in the reverse direction, to pass the contribution of each slave process to the objective function. This is accomplished by including calls to MPI procedures in the objective function and in the sequential Nelder-Mead simplex algorithm, whose logical steps, however, remain unchanged. We used a publicly available (Hutt, 2012) Nelder-Mead simplex procedure, which we adapted to include inter-process communication commands. The SSPO strategy is particularly suitable for situations with a large number of data points running on computers with many processors.

Parallel Simplex with Sequential Objective Function (PSSO)

The PSSO strategy uses a modified version of the Nelder-Mead simplex algorithm for parallel computations proposed recently (Lee and Wiswall, 2007). Complete details of the procedure are available in the original reference, which includes optimization examples of very simple objective functions with 100 or more unknowns. Here, the situation is much different: typical EOS parameter fitting problems have fewer unknowns and their objective functions depend on properties evaluated using complicated thermodynamic models.

The central idea is that each parallel process takes care of a vertex of the multidimensional simplex. A distinctive feature of applying this modified Nelder-Mead algorithm is that all parallelization commands are in the optimization procedure. Therefore, existing codes that calculate objective functions sequentially remain unchanged. Then, it is easy to link existing implementations of objectives functions and thermodynamic models with the parallel optimization procedure, with no need to rewrite code. There is little inter-process communication. The master process distributes the values of the objective function in all vertices of the simplex, sorted in increasing order, and the corresponding simplex coordinates to all other processes. Each process then does the reflexion, expansion, or contraction of the vertex assigned to it, calculating the objective function as needed, according to criteria that are immediate extensions of the usual Nelder-Mead algorithm. After exploring its simplex vertex, each process returns the objective function and the corresponding simplex coordinates to the master process. The master process then replaces its previous simplex points with the new ones, sorts them according to the value of their objective functions, and starts a new iteration. These steps are similar but not the same as in the conventional Nelder-Mead algorithm, causing important differences in performance, as discussed in the section "Results and Discussion".

To implement the Lee and Wiswall (2007) method, we adapted the Nelder-Mead simplex procedure of Hutt (2012) to include inter-process communication commands. The parallel Nelder-Mead method requires sorting the objective function in increasing order. While sorting is a problem suitable for parallel implementation, the lists to be sorted are small (the number of elements being equal to the number of parameters to fit plus one). Therefore, no attempt at parallelizing the sorting part was made and the hybrid quick sort algorithm, as implemented by Moss (2013), was used without any change.

The overall implementation is simple. All parallel processes read all the experimental data from the same data files when program execution begins. Then, the master process in the parallel Nelder-Mead simplex procedure interacts with its slave processes, distributing and collecting information, as outlined in the previous paragraphs. The PSSO strategy has the advantage of being easy to implement, but does not scale well if more processors become available. The number of slave processes it uses is less than or equal to the number of parameters to be fitted plus one (i.e., the number of vertices in the simplex). For example, if an eight processor computer is available to fit four parameters, the PSSO strategy will use up to six processors (one master and five slaves), while the other two processors will remain idle.

RESULTS AND DISCUSSION

The SSPO and PSSO strategies are applicable to parameter fitting problems with any objective function, any EOS, and experimental data points for any physical properties the EOS can predict. However, the PSSO strategy does not take advantage of slave processes in excess of the number of adjustable parameters plus 1 (i.e., the number of vertices in the simplex). The examples presented here refer to fitting binary interaction parameters between ions and water in the electrolattice EOS (Santos, 2010; Zuber *et al.*, 2013) for electrolyte solutions. The EOS details are of minor importance in what follows, but are available in the original references. The assumption is that a given interaction parameter, for example, between water and the sodium ion, will have the same value regardless of the anion(s) present. This problem is similar to fitting parameters in a group contribution model, in which the parameters of a group are the same in any molecule in which it is present.

When fitting EOS parameters, it is common to adopt parameter values previously fitted for similar systems (possibly with some perturbation) as initial guesses. To mimic this, perturbed values of parameters for the same interactions obtained by a different program for the same model were used. The experimental sets used consist of vapor pressure and mean ionic activity coefficient data. The objective function (f) used in all examples is:

$$f=\sum_{k=1}^{n_{sys}}\left(\sum_{j=1}^{n_{Pk}}\left(\frac{P_{jk}^{calc}-P_{jk}^{\exp}}{P_{jk}^{\exp}}\right)^2+\sum_{j=1}^{n_{\gamma k}}\left(\frac{\gamma_{jk}^{\pm calc}-\gamma_{jk}^{\pm\exp}}{\gamma_{jk}^{\pm\exp}}\right)^2\right) \quad (1)$$

in which n_{sys} is the number of systems, n_{Pk} and $n_{\gamma k}$ are the number of vapor pressure and mean ionic activity coefficient data points available for system k, respectively. The symbol P_{jk} represents the system pressure in the j-th vapor pressure data point of the k-th system. Its value is calculated by solving the isofugacity equation of water, assuming the ions are absent from the vapor phase. The symbol $\gamma_{jk}^{\pm}$ represents the mean ionic activity coefficient in the j-th mean ionic activity coefficient data point of the k-th system. This property is calculated from the fugacity coefficients of the cations and anions in the liquid phase using standard procedures of solution thermodynamics (Myers *et al.*, 2002). The superscripts *exp* and *calc* refer to experimental and calculated values, respectively. The calculations were assumed to converge when:

$$\frac{1}{n_p}\sum_{k=1}^{n_p+1}\left(f_k-\bar{f}\right)^2<10^{-7} \quad (2)$$

in which n_p is the number of fitted parameters, f_k is the value of the objective function in vertex k of the n_p+1 simplex vertices, and $\bar{f}$ is their average value.

The computation times reported refer to wall time, not CPU time, spent to execute the optimization, disregarding the time spent reading the input data and initializing the algorithm. Example 1 was executed on a Sony Vaio laptop, model VPCSB19GG, with an Intel Core i7-2620M processor (2 cores, 4 threads) at 2.70 GHz, running 64-bit Windows 7

Professional (computer S) and on a Dell desktop computer, model OPTIPLEX 990, with Intel Core i7-2600 processor (4 cores, 8 threads) at 3.40 GHz (computer D), running 64-bit Windows 7 Enterprise. These computations were executed when no other user-triggered calculations and operations were carried out. We used the Intel Visual Fortran compiler XE 12.0.1.127 (IA-32) and the MS MPI library. Example 2 was executed on computer D. Example 3 was executed on the Linux-based Suqoor supercomputer installed at Texas A&M University at Qatar, which is a SGI Altix XE1300 cluster system, with Intel Xeon X5472 CPUs, 3.00 GHz, 6MB L2 with Infiniband interconnection (20 Gbps links). The source code was ported unchanged to the Suqoor supercomputer and recompiled with the Intel MPI library and Fortran compiler.

Example 1: Fitting 2 Parameters from 50 Data Points

In this example, a single parameter (interaction between water and the sodium ion) is fitted using 25 experimental vapor pressure data points and 25 mean ionic activity coefficients data points of aqueous solutions of sodium chloride, with all the other model parameters kept fixed. The speedups of the SSPO strategy in each computer are shown in Table 1. With multiple processes, one of them is the master process that manages the computations and executes the operations of the sequential Nelder-Mead algorithm. The others are slave processes that execute the bulk of the computations, evaluating the thermodynamic properties and their contributions to the objective function. The processor in Computer S has only two cores and with three processes in it, one master and two slaves, the observed speedup was equal to 1.8. It is more interesting to observe what happens in Computer D, whose processor has four cores. Speedups follow the number of slave processes very closely up to three slave processes, when the total number of processes is equal to the number of processor cores. Because the slave processes do most of the calculations, this result suggests that the SSPO strategy has excellent parallelization characteristics, that a small fraction of the time is spent on inter-process communication, and that the computational loads are well balanced across the processes. Still analyzing the results of Computer D in Table 1, the use of more processes than the number of processor cores causes less substantial speedup increases. This suggests that the number of processor cores plays the major role in providing speedups and multithreading plays a secondary role.

Table 1: SSPO performance and speedup in Example 1 (wall time in the optimization routine).

		Computer S		Computer D	
Processes	Slave processes	Time (s)	Speedup	Time (s)	Speedup
1	0	1084	1.0	951	1.0
3	2	604	1.8	493	1.9
4	3	542	2.0	344	2.8
8	7	n.a.	n.a.	237	4.0

The PSSO strategy was executed with one and two processes. With a single process, it is equal to the conventional Nelder-Mead simplex algorithm and its execution took 1006 s. This value is similar to the 1084 s it took to run the SSPO approach in a single process (Table 1). The discrepancy between these two wall times (of about 8%) is attributed to differences in implementation details of the two numerical procedures. The trajectory to the solution and the parameters found at the end of each run are identical, as they should be. The PSSO strategy took 1250 s to run in two processes, more than the time to run it in a single process. However, the calculations converge to a different point of minimum, whose objective function is smaller than that of the other runs (SSPO with any number of processes and single process PSSO).

In an iteration of the SSPO strategy, only the vertex with the largest objective function is reflected and/or expanded and/or contracted, in what is essentially a one-dimensional search that strictly follows the Nelder-Mead simplex algorithm. In an iteration of the PSSO strategy, each simplex vertex assigned to a process undergoes such operations and, therefore, multiple one-dimensional searches occur in parallel. This more detailed mapping of the function behavior in a PSSO iteration seemingly increases the chance of finding minima with smaller objective functions.

Example 2: Fitting 5 Parameters from 321 Data Points

In this example, 321 experimental data points (223 vapor pressure data points and 98 mean ionic activity coefficient data points) of 4 aqueous single-salt solutions are used to fit 5 parameters simultaneously, which represent binary interactions between water and different cations (Li^+, Na^+, K^+, Cs^+) and the Cl^- anion.

Table 2 summarizes the results of applying the SSPO strategy with different numbers of processes in computer D. As in the previous example, the

SSPO strategy converges to the same solution regardless of the number of processes used. Its speedups are in good agreement with those observed in Example 1. With 4 processes (1 master and 3 slaves), the speedup is 2.7. Since the 3 slave processes carry out most of the calculations, this speedup of 2.7 confirms the excellent parallelization characteristics of the SSPO strategy. The next entry in Table 2 is for the run with 5 processes, more than the number of available cores (4) but fewer than the number of available threads (8) in computer D. The speedup with 5 processes (4 of them slaves) is equal to 3.3, which is larger than that with 4 processes (3 of them slaves), which is equal to 2.7. With 8 processes (7 of them slaves), the speedup is equal to 4.3. The joint analysis of these results supports the comment made in the discussion of Example 1, suggesting the number of processor cores is the key factor in speeding up the calculations, with multithreading playing a less prominent role.

Table 2: SSPO performance and speedup in Example 2 using computer D (wall time in the optimization routine).

Processes	Slave processes	Time (s)	Speedup
1	0	49172	1.0
4	3	18349	2.7
5	4	14702	3.3
8	7	11510	4.3

In the same computer, the PSSO strategy with 5 processes converged in 23583 s, i.e., a speedup of 2.1 compared to the SSPO case with one processor. This speedup is smaller than that of the SSPO case but, as in Example 1, the PSSO strategy converges to a different parameter set, with smaller objective function.

Example 3: Fitting 8 Parameters from 781 Data Points

In this example, 8 parameters that represent binary interactions between water and different cations (Li^+, Na^+, K^+, Rb^+, Cs^+) and anions (Cl^-, Br^-, I^-) are fitted simultaneously using 781 experimental data points of 13 aqueous single-salt solutions. 493 of these points are vapor pressure data and 288 are mean ionic activity coefficient data.

This problem was solved on the Suqoor supercomputer. Table 3 summarizes the results using a single process and applying the SSPO strategy with 8 processes. Taking into account that the master process in the SSPO implementation performs only a few and simple calculations, the slave processes execute the bulk of the computations. The observed speedup of 6.9 is very close to 7, which is the number of slave processes in the example of Table 3. This confirms the observations of the previous examples in that the burden of inter-process communication is small compared to evaluating the thermodynamic properties needed to calculate the objective function, as also observed in Examples 1 and 2, executed on computers S and D.

Table 3: SSPO performance and speedup in Example 3 using the Suqoor supercomputer (wall time in the optimization routine).

Processes	Slave processes	Time (s)	Speedup
1	0	96734	1.0
8	7	14040	6.9

The problem was also solved using the PSSO strategy with 8 processes, spending 21411 s in the optimization part of the run, i.e., a speedup of about 4.5 from the single process SSPO run (Table 3). This speedup is less pronounced than that obtained by the SSPO strategy with the same number of processes, but the PSSO run converges to a different solution, which has a smaller final value of the objective function.

Table 4 presents the initial estimate and the optimal values of the fitted parameters, as well as the average relative deviations, calculated by the SSPO and PSSO strategies in Examples 1-3. In this table, O.F. stands for the objective function values for the PSSO strategy with one (PSSO (1)) or two processes (PSSO (2)), and SSPO strategy with one, three, four, five, and eight processes (SSPO (1, 3, 4, 5, 8)). In Example 1, the O.F. value for PSSO (2) is lower than those obtained for PSSO (1) and SSPO (1, 3, 4, 8). The parameters fitted by these strategies are different, as expected, since the O.F. values are not identical. To illustrate, Figure 1 compares the mean ionic activity coefficient of NaCl at 298.15 K to experimental data, using the PSSO (1) and PSSO (2) strategies. Despite the different parameters and O.F. values, the adherence of the model to the experimental data is adequate and similar in both cases. Similar patterns were observed in all examples.

Comparing the performance of the two strategies in this set of examples, the SSPO strategy produces larger speedups than the PSSO strategy for the same number of processes. The SSPO strategy implements the conventional Nelder-Mead simplex procedure widely used for fitting the parameters of thermodynamic models, with speedups that result from

Table 4: Initial estimate and optimal values of the parameters calculated for Examples 1 to 3 and the objective function in each case.

Example	Condition	Li^+	Na^+	K^+	Rb^+	Cs^+	Cl^-	Br^-	I^-	O. F.
1	Initial	–	–2300	–	–	–	–1300	–	–	0.2734
	PSSO (1) and SSPO (1, 3, 4, 8)	–	–2454	–	–	–	–1316	–	–	0.0147
	PSSO (2)	–	–2498	–	–	–	–621	–	–	0.0134
2	Initial	–2700	–2300	–150	–	1100	–1700	–	–	1.1961
	SSPO (1, 4, 5, 8)	–2689	–2122	–120	–	1163	–1949	–	–	0.3334
3	Initial	–2900	–2500	–40	100	1000	–1900	–1600	–1500	90.1650
	SSPO (1, 4, 5, 8)	–2778	–2373	–43	99	1132	–1657	–1564	–1463	2.3860

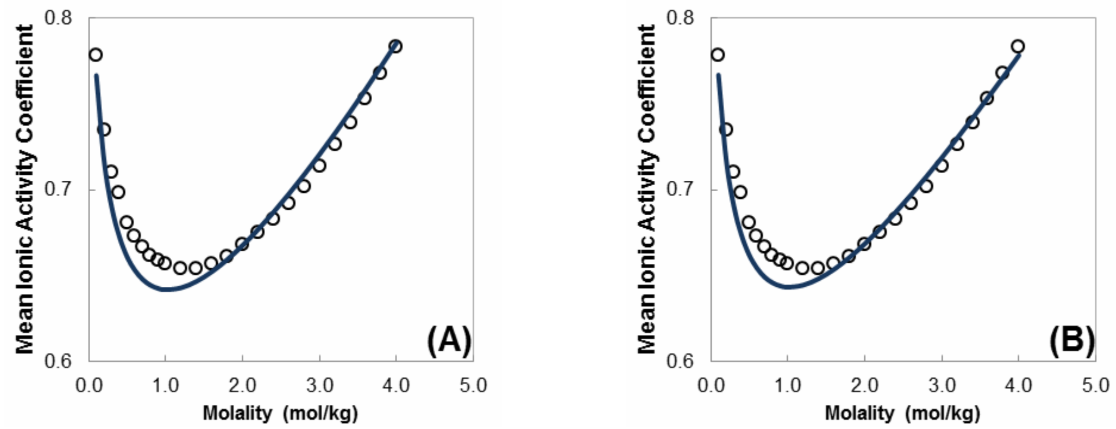

Figure 1: Mean ionic activity coefficient of NaCl using the following strategies: (A) – PSSO (1) and SSPO (1, 3, 4, 8) and (B) – PSSO (2); model (full line) and experimental data (circles) from Lobo and Quaresma (1989).

splitting the evaluation of the objective function among the running processes. Also, the number of processes the SSPO strategy can use effectively is not limited by the number of parameters to be fitted, as in the PSSO strategy. However, the PSSO strategy implements a modified version of the simplex algorithm. Its speedups are less impressive than those of the SSPO strategy but, on the other hand, it found smaller objective functions than the SSPO strategy, likely because of the multiple parallel one-dimensional searches that occur during each iteration. Choosing which of these two strategies to use in a given situation is a matter of convenience and need, and both are potentially useful to developers and users of thermodynamic models.

CONCLUSIONS

The two strategies for exploiting parallelization when fitting equation of state parameters tested in this work, the sequential simplex with parallel objective function (SSPO) and the parallel simplex with sequential objective function (PSSO), proved to be simple to set up, with the latter being simpler than the former. Both allow using sequential codes to evaluate physical properties, which is arguably the most intricate part of parameter fitting codes for modern equations of state. The two strategies were shown to be practical for using the multiple processors of modern PCs to perform parallel computations in parameter fitting problems, with few changes to legacy sequential programs.

The SSPO strategy implements the conventional Nelder-Mead simplex algorithm, widely used for fitting the parameters of thermodynamic models. As implemented, its maximum achievable speedup should be equal to the number of slave processes used. Speedups very close to this upper limit were observed whenever the total number of processes did not exceed the number of processor cores. This indicates the excellent parallelization features of the SSPO strategy and of the problem, as a consequence of the fact that the computational load is dominated by the evaluation of the objective function in a direct optimization method. Multithreading also contributes to speedups but the improvement it causes seems to be secondary compared to increasing the number of processor cores. Taken together, these results of the SSPO strategy show that it is possible to run the conventional Nelder-Mead simplex algorithm much faster than usual, if one takes advantage of the multiple cores currently available in many PCs.

The PSSO strategy uses a modified simplex algorithm that executes multiple parallel one-dimensional searches during each iteration. In the examples of this paper, speedups of the PSSO strategy are smaller than those of the SSPO strategy using the same number of processes. This is possibly because of its more detailed mapping of the objective function behavior and the PSSO strategy converged to parameter sets with objective functions smaller than those obtained by the SSPO strategy in the examples of this paper. While the PSSO strategy cannot be claimed to be globally convergent, it avoided local minima that trapped the conventional Nelder-Mead algorithm of the SSPO strategy.

In summary, both strategies are potentially useful and utilizing one or the other is a matter of convenience, depending on the parameter fitting problem at hand. The use of MPI allowed porting the program from Windows-based computers to a large Linux-based cluster without code changes. This is helpful if a parameter fitting problem is too demanding for a single PC, even when it has multiple processors. The easy implementation and excellent results of the strategies discussed here should appeal to developers and users of thermodynamic models.

ACKNOWLEDGMENT

The authors thank Dr. Charlles R. A. Abreu (Federal University of Rio de Janeiro, Brazil) for his suggestions during the early stages of this work.

LIST OF SYMBOLS

f	objective function
f_k	objective function in vertex k of the $n_p + 1$ simplex vertices
$\overline{f}$	average value of the objective function in the $n_p + 1$ simplex vertices
n_p	number of fitted parameters
n_{Pk}	number of vapor pressure data points available for system k
n_{sys}	number of systems
$n_{\gamma k}$	number of mean ionic activity coefficient data points available for system k
P_{jk}	system pressure in the j-th vapor pressure data point of the k-th system

Greek Letter

$\gamma_{jk}^{\pm}$	mean ionic activity coefficient in the j th data point of the k th system.

Superscripts

calc	calculated values
exp	experimental values

REFERENCES

Abdel-Jabbar, N., Carnahan, B. and Kravaris, C., A Multirate parallel-modular algorithm for dynamic process simulation using distributed memory multicomputers. Comput. Chem. Eng., 23, 733 (1999).

Almeida, B. F., Waldrigui, T. M., Alves, T. C., Oliveira, L. H. and Aznar, M., Experimental and calculated liquid-liquid equilibrium data for water plus furfural plus solvents. Fluid Phase Equilib., 334, 97 (2012).

Bertrand, F., Gange T., Desaulniers, E., Vidal, D. and Hayes, R. E., Simulation of the consolidation of paper coating structures: Probabilistic versus deterministic models. Comput. Chem. Eng., 28, 2595 (2004).

Brochard, L., Hardware and software perspectives in engineering computing. Comput. Chem. Eng., 22 Suppl., S1085 (1998).

Cho, D. W., Shin, J., Bae, W., Kim, H., Lee, J. H. and Shin, M. S., High-pressure phase behavior of triethylene glycol dimethacrylate and tetra-ethylene glycol dimethacrylate in supercritical carbon dioxide. Fluid Phase Equilib., 319, 37 (2012).

Dominguez, A., Tojo, J., Castier, M., Automatic implementation of thermodynamic models for reliable parameter estimation using computer algebra. Comput. Chem. Eng., 26, 1473 (2002).

Gau, C. Y., Brennecke, J. F. and Stadtherr, M. A., Reliable nonlinear parameter estimation in VLE modeling. Fluid Phase Equilib., 168, 1 (2000).

Gau, C. Y. and Stadtherr, M. A., Dynamic load balancing for parallel interval-newton using message passing. Comput. Chem. Eng., 26, 811 (2002).

Goodale, T., Allen, G., Lanfermann, G., Massó, J., Radke, T., Seidel, E. and Shalf, J., The Cactus framework and toolkit: Design and applications. Vector and Parallel Processing – VECPAR'2002. 5th International Conference, Lecture Notes in

Computer Science, Springer, Berlin (2003).

Hutt, M. F., Program FROSEN. <http://www. mikehutt. com/frosen.f> (Access date: May 30, 2012).

Ingram, T., Gerlach, T., Mehling, T. and Smirnova, I., Extension of COSMO-RS for monoatomic electrolytes: Modeling of liquid-liquid equilibria in presence of salts. Fluid Phase Equilib., 314, 29 (2012).

Jana, C., Ray, P. and De, P., Ternary liquid-liquid equilibrium: Nitric acid-water-anisole/4-methyl anisole. Fluid Phase Equilib., 314, 82 (2012).

Justo-García, D. N., García-Sánchez, F., Díaz-Ramírez, N. L. and Díaz-Herrera, E., Modeling of three-phase vapor-liquid-liquid equilibria for a natural-gas system rich in nitrogen with the SRK and PC-SAFT EoS. Fluid Phase Equilib., 298, 92 (2010).

Lai, N. A., Wendland, M. and Fischer, J., Description of linear siloxanes with PC-SAFT equation. Fluid Phase Equilib., 283, 22 (2009).

Lee, D. and Wiswall, M., A Parallel implementation of the simplex function minimization routine. Comput. Econom., 30, 171 (2007).

Leineweber, D. B., Schafer, A., Bock, H. G., Schloder, J. P., An efficient multiple shooting based reduced SQP strategy for large-scale dynamic process optimization - Part II: Software aspects and applications. Comput. Chem. Eng., 27, 167 (2003).

Llano-Restrepo, M. and Muñoz-Muñoz, Y. M., Combined chemical and phase equilibrium for the hydration of ethylene to ethanol calculated by means of the Peng-Robinson-Stryjek-Vera equation of state and the Wong-Sandler mixing rules. Fluid Phase Equilib., 307, 45 (2011).

Lobo, V. M. M., Quaresma, J. L., Handbook of Electrolyte Solutions, Part A and B. 1st Ed. Amsterdam, Elsevier (1989).

Mesquita, F. M. R., Bessa, A. M. M., Lima, D. D., Sant'Ana, H. B. and Santiago-Aguiar, R. S., Liquid-liquid equilibria of systems containing cottonseed biodiesel plus glycerol plus ethanol at 293.15, 313.15 and 333.15 K. Fluid Phase Equilib., 318, 51 (2012).

Moss, L. J., Subroutine SORTRX. <http://www.nco.ncep.noaa.gov/pmb/codes/nwprod/sorc/cfs_atmos_fcst.fd/sortrx.f> (Access date: April 1, 2013).

Munson, T., Sarich, J., Wild, S., Benson, S. and McInnes, L. C., TAO 2.0 Users Manual. Mathematics and Computer Science Division. Argonne National Laboratory (2012).

Myers, J. A., Sandler, S. I. and Wood, R. H., An equation of state for electrolyte solutions covering wide ranges of temperature, pressure, and composition. Ind. Eng. Chem. Res., 41, 3282 (2002).

Nelder, J. A. and Mead, R., A simplex method for function minimization. Computer Journal, 7, 308 (1965).

Olaya, M. M., Reyes-Labarta, J. A., Velasco, R., Ibarra, I., Marcilla, A., Modeling liquid-liquid equilibria for island type ternary systems. Fluid Phase Equilib., 265, 184 (2008).

Paloschi, J. R., Testing a new parallel preconditioner on linear systems arising from flowsheeting simulation. Comput. Chem. Eng., 21 Suppl., S433 (1997).

Paloschi, J. R., Steps towards steady-state process simulation on mimd machines: Implementation in the SPEEDUP simulator. Comput. Chem. Eng., 22, 1189 (1998).

Santos, J. P. L., Equilíbrio de fases de misturas polares e iônicas via equação de estado baseada em modelo de rede. D.Sc. Thesis, Federal University of Rio de Janeiro, Brazil (2010). (In Portuguese).

Santos, J. P. L., Tavares F. W. and Castier M., Vapor-liquid equilibrium calculations for refrigerant mixtures with the Mattedi-Tavares-Castier EOS. Fluid Phase Equilib., 296, 133 (2010).

Schilling, G. and Pantelides C. C., Optimal periodic scheduling of multipurpose plants. Comput. Chem. Eng., 23, 635 (1999).

Schmid, B. and Gmehling J., Revised parameters and typical results of the VTPR group contribution equation of state. Fluid Phase Equilib., 317, 110 (2012).

Scott, J. A., Two-stage ordering for unsymmetric parallel row-by-row frontal solvers. Comput. Chem. Eng., 25, 323 (2001).

Scott, J. A., The design of a portable parallel frontal solver for chemical process engineering problems. Comput. Chem. Eng., 25, 1699 (2001a).

Siirola, J. D., Hauan S. and Westerberg A. W., Toward agent-based process systems engineering: Proposed framework and application to non-convex optimization. Comput. Chem. Eng., 27, 1801 (2003).

Smith, E. M. B. and Pantelides C. C., Global optimisation of nonconvex MINLPs. Comput. Chem. Eng., 21 Suppl., S791 (1997).

Smith, E. M. B. and Pantelides C. C., A symbolic reformulation/spatial branch-and-bound algorithm for the global optimisation of nonconvex MINLPs. Comput. Chem. Eng., 23, 457 (1999).

Soares, R. P., Gerder, R. P., Functional-segment activity coefficient model. 1. Model formulation. Ind. Eng. Chem. Res., 52(32), 11159 (2013).

Staudt, P. B., Cardozo, N. S. M., Soares, R. P., Phase stability analysing using a modified affine arithmetic. Comput. Chem. Eng., 53, 190 (2013).

Zuber, A., Checoni, R. F., Mathew R., Santos J. P. L., Tavares F. W. and Castier M., Thermodynamic properties of 1:1 salt aqueous solutions with the electrolattice equation of state. Oil and Gas Sci. Technol., Revue d'IFP Energies Nouvelles, 68(2), 255 (2013).