Computers and Chemical Engineering 71 (2014) 154–157

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

On the equivalence between the modifier-adaptation and trust-region frameworks

Gene A. Bunin

ARTICLE INFO

Article history:
Received 23 December 2013
Received in revised form 28 July 2014
Accepted 31 July 2014
Available online 8 August 2014

Keywords:
Modifier adaptation
Trust-region methods
Real-time optimization

ABSTRACT

In this short note, the recently popular modifier-adaptation framework for real-time optimization is discussed in tandem with the well-developed trust-region framework of numerical optimization, and it is shown that the basic version of the former is a simplification of the latter when the problem is unconstrained. This relation is then exploited to propose a globally convergent modifier-adaptation algorithm using already developed trust-region theory. Cases when the two may not be equivalent and extensions to constrained problems are also discussed.

1. The real-time optimization problem

In the process systems engineering community, the basic idea of most real-time optimization (RTO) schemes consists in finding a set of optimal operating conditions – often steady-state setpoints in a multilayer hierarchical scheme – that minimize (resp., maximize) the steady-state cost (resp., profit) of some given plant subject to constraints (Brdys and Tatjewski, 2005). While models of the process being optimized are often available, it is generally the case that they are either inaccurate and/or incomplete, which motivates the data-driven "real-time" element of RTO, thereby forcing the optimization algorithm to use the measurements obtained from the process as feedback to modify the provided setpoints so as to ultimately reject the model uncertainty and converge to the optimal conditions of the plant.

A fairly general mathematical formulation of this problem that suffices for many practical cases is as follows:

$$\begin{aligned} &\underset{\mathbf{u}}{\text{minimize}} \quad \phi_p(\mathbf{u}) \\ &\text{subject to } g_{p,j}(\mathbf{u}) \le 0, \quad j = 1, \ldots, n_g, \end{aligned} \tag{1}$$

where $\mathbf{u} \in \mathbb{R}^{n_u}$ denote the decision variables, or the "inputs", of the problem, while the functions $\phi, g : \mathbb{R}^{n_u} \to \mathbb{R}$ denote the cost and constraints, respectively. The subscript p (for "plant") is used to indicate that the function corresponds to an experimental relationship that is not perfectly known and may only be approximated by a model, which we will mark with the subscript $\hat{p}$ (e.g., $\phi_{\hat{p}}$ being the model approximation of ϕ_p). In the simplest terms, the goal of an RTO algorithm is to solve Problem (1) by iterative experimentation, generating a sequence of steady-state $\mathbf{u}$ values that converges to the plant optimum.

For the majority of this document, we will not focus on Problem (1) but on the unconstrained case

$$\underset{\mathbf{u}}{\text{minimize}} \quad \phi_p(\mathbf{u}), \tag{2}$$

as this is sufficient to convey the main message. We will, however, return to Problem (1) in the end in passing, providing references to works where it is discussed properly and in much greater detail.

2. Review of the modifier-adaptation framework

An approach to solving (1) that has recently gained popularity in the research community is that of *modifier adaptation*, which originally dates back to the work of Roberts (1978) and owes its numerous refinements and fundamental ideas to the *ISOPE* ("iterative setpoint optimization and parameter estimation") framework (Brdys and Tatjewski, 2005). Recent works by Gao and Engell (2005), Chachuat et al. (2009), and Marchetti et al. (2009) have given the approach its modern form by accounting for plant-model mismatch in both the cost and constraints. A number of works in the past few years have also considered various particular aspects of the framework, such as mathematical reformulations to ease or better accommodate particular problem types (François and Bonvin, 2013; Serralunga et al., 2013; Costello et al., 2013), important implementation aspects (Marchetti et al., 2010; Rodger, 2010; Bunin et al., 2012), and major theoretical issues like feasibility

E-mail address: gene.a.bunin@ccapprox.info

http://dx.doi.org/10.1016/j.compchemeng.2014.07.028
0098-1354/

(Bunin et al., 2011; Navia et al., 2012) and global convergence (Faulwasser and Bonvin, 2014).

The basic philosophy of modifier adaptation lies in applying local corrections to an inherently incorrect model at each RTO iteration k, and solving this corrected version to obtain the following iterate at $k+1$. For the unconstrained case, this would lead to the following update:

$$\mathbf{u}_{k+1} \in \arg\underset{\mathbf{u}}{\text{minimize}} \quad \phi_{\hat{p}}(\mathbf{u}) + \boldsymbol{\lambda}_k^T \mathbf{u}, \tag{3}$$

with the *modifiers* $\boldsymbol{\lambda}_k$, defined as

$$\boldsymbol{\lambda}_k := \nabla\phi_p(\mathbf{u}_k) - \nabla\phi_{\hat{p}}(\mathbf{u}_k), \tag{4}$$

serving to ensure that the plant and corrected model have matching first derivatives at the current iterate $\mathbf{u}_k$.

Placing this into algorithmic form yields the following basic implementation.

Algorithm 1 *(Basic modifier-adaptation algorithm).*

1. *Initialization*: The initial point, $\mathbf{u}_0$, is provided. Set $k := 0$.
2. *Modifier computation*: Compute the modifiers $\boldsymbol{\lambda}_k$ according to (4).
3. *New input calculation*: Obtain $\mathbf{u}_{k+1}$ by solving Problem (3) and apply this set of inputs to the plant.
4. *Iterate*: Set $k := k+1$ and return to Step 2.

The key oft-stated motivation for applying this algorithm is the following upon-convergence guarantee.

Theorem 1. *(First-order critical point upon convergence). Assume that the minimization of (3) always yields a first-order critical point of the modified objective function $\phi_{\hat{p}}(\mathbf{u}) + \boldsymbol{\lambda}_k^T\mathbf{u}$ and that Algorithm 1 has converged to a fixed point $\mathbf{u}_\infty$. It follows that $\mathbf{u}_\infty$ is a first-order critical point of ϕ_p.*

Proof. The result follows immediately from the fact that a first-order critical point for an unconstrained problem is defined entirely by the function's derivatives at that point. As these must match for the modified model and the plant at any iterate, including $\mathbf{u}_\infty$, it follows that finding a first-order critical point for the modified function implies finding one for the plant. □

3. The basic trust-region algorithm

A theoretically rigorous approach for iteratively minimizing a nonlinear function in the mathematical optimization context is that of trust-region methods. In this section, we will consider what attempting to solve Problem (2) in this framework would entail.

Let us start by stating the basic trust-region algorithm for solving (2). This is essentially the algorithm provided in the well-known monograph on trust-region methods (Conn et al., 2000, Ch. 6) but with a few additional simplifications and some notational changes. Namely, we use the 2-norm instead of the general p-norm and explicitly distinguish between the reference iterates, $\mathbf{u}_k^*$, and the iterates applied to the plant, $\mathbf{u}_k$.

Algorithm 2 *(Basic trust-region algorithm).*

1. *Initialization*: The initial point, $\mathbf{u}_0$, and initial trust-region radius, $\Delta_0 > 0$, are provided, together with the constants η_1, η_2, γ_1, and γ_2 satisfying $0 < \eta_1 \le \eta_2 < 1$ and $0 < \gamma_1 \le \gamma_2 < 1$. Set $k := 0$, $\mathbf{u}_0^* := \mathbf{u}_0$, and apply $\mathbf{u}_0$ to the plant to obtain $\phi_p(\mathbf{u}_0^*) = \phi_p(\mathbf{u}_0)$.
2. *Model construction*: Construct the model m_k, which is an approximation of ϕ_p over the trust region $\mathcal{B}(\mathbf{u}_k^*, \Delta_k)$, i.e., over a Euclidean ball of radius Δ_k centered at $\mathbf{u}_k^*$.
3. *New input candidate calculation*: Compute a candidate point $\mathbf{u}_{k+1} \in \mathcal{B}(\mathbf{u}_k^*, \Delta_k)$ that "sufficiently reduces the model" m_k.
4. *Acceptance of the candidate point*: Apply $\mathbf{u}_{k+1}$ to the plant to obtain $\phi_p(\mathbf{u}_{k+1})$. Define:

$$\rho_k := \frac{\phi_p(\mathbf{u}_k^*) - \phi_p(\mathbf{u}_{k+1})}{m_k(\mathbf{u}_k^*) - m_k(\mathbf{u}_{k+1})}. \tag{5}$$

If $\rho_k \ge \eta_1$, then set $\mathbf{u}_{k+1}^* := \mathbf{u}_{k+1}$. Otherwise, set $\mathbf{u}_{k+1}^* := \mathbf{u}_k^*$.

5. *Trust-region radius update*: Set Δ_{k+1} such that

$$\Delta_{k+1} \in \begin{cases} [\Delta_k, \infty) & \text{if } \rho_k \ge \eta_2, \\ [\gamma_2\Delta_k, \Delta_k] & \text{if } \rho_k \in [\eta_1, \eta_2), \\ [\gamma_1\Delta_k, \gamma_2\Delta_k] & \text{if } \rho_k < \eta_1. \end{cases} \tag{6}$$

6. *Iterate*: Set $k := k+1$ and return to Step 2.

Let us now state the assumptions sufficient to prove the global convergence of Algorithm 2 to a first-order critical point (Conn et al., 2000). The following are assumed about the nature of the plant:

Assumption 1. ϕ_p is $\mathcal{C}^2$ (twice continuously differentiable) on $\mathbb{R}^{n_u}$.

Assumption 2. ϕ_p is lower-bounded on $\mathbb{R}^{n_u}$.

Assumption 3. The Hessian of ϕ_p is upper-bounded on $\mathbb{R}^{n_u}$.

As mentioned in Conn et al. (2000), Assumption 3 is often too strong and could actually be restricted to the subspace of $\mathbb{R}^{n_u}$ where the iterates lie. However, as this subspace is not known *a priori*, $\mathbb{R}^{n_u}$ is used for notational convenience.

The following assumptions are made on the model:

Assumption 4. For all k, m_k is $\mathcal{C}^2$ over $\mathcal{B}(\mathbf{u}_k^*, \Delta_k)$.

Assumption 5. m_k matches ϕ_p locally to first order at every k, i.e.:

$$m_k(\mathbf{u}_k^*) = \phi_p(\mathbf{u}_k^*), \tag{7}$$

$$\nabla m_k(\mathbf{u}_k^*) = \nabla\phi_p(\mathbf{u}_k^*). \tag{8}$$

Finally, one requires the following assumption on the algorithm used to solve the trust-region subproblem with regard to its ability to achieve "sufficient reduction" in the model:

Assumption 6. There exists a constant $\kappa \in (0, 1)$ such that for all k:

$$m_k(\mathbf{u}_k^*) - m_k(\mathbf{u}_{k+1}) \ge \kappa\|\nabla m_k(\mathbf{u}_k^*)\| \min\left[\frac{\|\nabla m_k(\mathbf{u}_k^*)\|}{\beta_k}, \Delta_k\right], \tag{9}$$

with $\beta_k > 1$ a finite constant.

One may then state the following.

Theorem 2 (Global convergence to a first-order critical point). *If Assumptions 1–6 are satisfied, it then follows that the iterates generated by Algorithm 2 converge asymptotically to a first-order critical point, i.e.:*

$$\lim_{k\to\infty} \|\nabla\phi_p(\mathbf{u}_k^*)\| = 0. \tag{10}$$

Proof. The reader is referred to Theorem 6.4.6 in Conn et al. (2000). Note that we have, for simplicity, used a slightly stronger assumption and have assumed that m_k is $\mathcal{C}^2$ over $\mathcal{B}(\mathbf{u}_k^*, \Delta_k)$. The two assumptions made by Conn et al. (2000) – namely, that over $\mathcal{B}(\mathbf{u}_k^*, \Delta_k)$ the model m_k is twice differentiable and that its Hessian is bounded – are implied by the single $\mathcal{C}^2$ assumption here. □

4. Equivalence and a globally convergent modifier-adaptation scheme

Both the modifier-adaptation and trust-region algorithms seek to minimize ϕ_p by iteratively optimizing a local approximation of ϕ_p around each $\mathbf{u}_k^*$. The key differences between the two may be summarized as follows:

1. The model $m_k(\mathbf{u}) = \phi_{\hat{p}}(\mathbf{u}) + \boldsymbol{\lambda}_k^T\mathbf{u}$ used by modifier-adaptation enforces, by construction, (8) but not (7). The standard trust-region algorithm usually enforces both as this is required for the convergence proof.
2. The modifier-adaptation subproblem (3) considers the whole input space while the trust-region subproblem limits its search to the ball $\mathcal{B}(\mathbf{u}_k^*, \Delta_k)$.
3. The concept of a "reference point" is absent in the basic modifier-adaptation algorithm, as the computed $\mathbf{u}_{k+1}$ is always used as the reference with respect to which the model is corrected at the subsequent iteration. In the trust-region scheme, the model is always built with respect to the latest "successful" iterate for which a sufficient decrease in the plant cost function value has been achieved.

The first difference is actually of no practical consequence – as discussed later (see Corollary 1), one could always use a model that satisfies both (7) and (8) without changing the iterates generated by the modifier-adaptation algorithm. The second and third differences, however, are important and may aid in explaining why no globally convergent version of Algorithm 1 has been derived to date. Without the use of a reference point, it is difficult to ensure the stability of the algorithm, since any progress made may always be undone by a single bad iteration. Optimizing with respect to the best known point effectively prevents bad iterations from having any lasting effect on convergence, but is not sufficient to guarantee the existence of a good iteration. For this, one needs the guarantee that the model used by the algorithm become sufficiently good under certain conditions. Since the model is only good locally and to first order, the natural approach, and the one pursued in trust-region methods, is to shrink the search space until this approximation is good enough to generate a successful iterate. By considering the entire input space, the modifier-adaptation algorithm may generate iterates in portions of the input space that are not accurately modeled, and so it should not be surprising that the guarantee of successful iterates is absent in this algorithm.

Note that all of these differences are of the same nature, in that they are all things that are present in the trust-region framework but absent in modifier adaptation. In fact, if we were to enforce that $\mathbf{u}_{k+1}^* := \mathbf{u}_{k+1}$ always and let $\Delta_0 \to \infty$ and $\gamma_1 \to 1$ (i.e., remove the trust-region restriction) in Algorithm 2, we would essentially end up with Algorithm 1. Considering things from this perspective, let us now avoid these simplifications and propose the following modifier-adaptation scheme.

Algorithm 3 *(Trust-region supplemented modifier-adaptation algorithm).*

1. *Initialization*: Identical to Step 1 of Algorithm 2.
2. *Modifier computation*: Compute the modifiers $\boldsymbol{\lambda}_k := \nabla\phi_p(\mathbf{u}_k^*) - \nabla\phi_{\hat{p}}(\mathbf{u}_k^*)$.
3. *Model correction*: Construct the model $m_k(\mathbf{u}) := \phi_{\hat{p}}(\mathbf{u}) + \boldsymbol{\lambda}_k^T\mathbf{u}$.
4. *New input candidate calculation*: Compute a candidate point $\mathbf{u}_{k+1}$ by approximately solving the problem

$$\mathbf{u}_{k+1} \in \arg\underset{\mathbf{u}\in\mathcal{B}(\mathbf{u}_k^*,\Delta_k)}{\text{minimize}} \; m_k(\mathbf{u}). \tag{11}$$

Furthermore, compute the Cauchy point, $\mathbf{u}_{k+1}^{\mathrm{CP}}$, via the line search

$$[\mathbf{u}_{k+1}^{\mathrm{CP}}, t^{\mathrm{CP}}] \in \arg \begin{array}{ll} \underset{\mathbf{u}\,\in\,\mathcal{B}(\mathbf{u}_k^*,\Delta_k)}{\text{minimize}} & m_k(\mathbf{u}) \\ t \geq 0 & \\ \text{subject to} & \mathbf{u} = \mathbf{u}_k^* - t\nabla m_k(\mathbf{u}_k^*). \end{array} \tag{12}$$

If $m_k(\mathbf{u}_{k+1}) > m_k(\mathbf{u}_{k+1}^{\mathrm{CP}})$, set $\mathbf{u}_{k+1} := \mathbf{u}_{k+1}^{\mathrm{CP}}$.

5. *Acceptance of the candidate point*: Identical to Step 4 of Algorithm 2.
6. *Trust-region radius update*: Identical to Step 5 of Algorithm 2.
7. *Iterate*: Set $k := k+1$ and return to Step 2.

Prior to proving the global convergence of Algorithm 3, we modify Assumption 4 to make it more direct.

Assumption 7. For all k, $\phi_{\hat{p}}$ is $\mathcal{C}^2$ over $\mathcal{B}(\mathbf{u}_k^*, \Delta_k)$.

The following key result follows.

Corollary 1 (Global convergence to a first-order critical point for modifier adaptation). *If Assumptions 1–3 and 7 are satisfied, it then follows that the iterates generated by Algorithm 3 converge asymptotically to a first-order critical point, i.e.:*

$$\lim_{k\to\infty} \|\nabla\phi_p(\mathbf{u}_k^*)\| = 0. \tag{13}$$

Proof. Algorithm 3 is special case of Algorithm 2, and so we just need to show that all of the assumptions needed for Theorem 2 are satisfied either implicitly or explicitly. As Assumptions 1–3 are made explicitly throughout, we focus on Assumptions 4–6. Since adding a linear correction term to a $\mathcal{C}^2$ function will not jeopardize the $\mathcal{C}^2$ property, making Assumption 7 implies that Assumption 4 holds. While Condition (8) of Assumption 5 is satisfied by construction, Condition (7) is not. However, note that we may just as easily use the model $m_k(\mathbf{u}) := \phi_{\hat{p}}(\mathbf{u}) + [\phi_p(\mathbf{u}_k^*) - \phi_{\hat{p}}(\mathbf{u}_k^*)] + \boldsymbol{\lambda}_k^T(\mathbf{u} - \mathbf{u}_k^*)$, which satisfies both (7) and (8) by construction but does not influence the sequence of iterates produced by Algorithm 3 since the addition of the constant term $\phi_p(\mathbf{u}_k^*) - \phi_{\hat{p}}(\mathbf{u}_k^*) - \boldsymbol{\lambda}_k^T\mathbf{u}_k^*$ does not influence the computation of ρ_k or $\mathbf{u}_{k+1}$ in any way. By sleight of hand, we may thus "pretend" to use the latter model and consider Assumption 5 satisfied, as the two models are equivalent with respect to the sequence of iterates generated. Finally, overriding the standard computation of $\mathbf{u}_{k+1}$ in Step 4 with the Cauchy point when needed ensures that Assumption 6 is met (Conn et al., 2000, §6.3). □

5. Nonequivalent cases and practical considerations

As the guarantee of global convergence is a very desirable property, and as the additions to ensure it for the basic modifier-adaptation scheme are simple and algorithmic in nature, it is tempting to ask if not every modifier-adaptation scheme could be cast in a globally convergent trust-region formulation. While further research is required to give a definitive answer, a preliminary inspection seems to suggest the answer to be positive.

Perhaps of greatest interest is the question of how the prior discussion extends to the constrained problem (1), since almost all problems in practice are constrained. The standard approach in trust-region methods is to cast such problems as unconstrained problems with a penalty for constraint violations included in the augmented cost function (Conn et al., 2000, Ch. 14), and the recent work by Biegler et al. (2014), without stating so explicitly, essentially shows how the constrained modifier-adaptation problem may be solved in the trust-region framework by exploiting this approach. While one could propose different implementation routes with regard to particular algorithmic aspects, there appears to be no reason as to why the extension to (1) would not come easily.

Another popular technique in modifier-adaptation schemes is to filter the modifiers (Marchetti et al., 2009; Chachuat et al., 2009; Serralunga et al., 2013) so as to not "overcorrect" the model, and to define them as

$$\boldsymbol{\lambda}_k := \alpha[\nabla\phi_p(\mathbf{u}_k^*) - \nabla\phi_{\hat{p}}(\mathbf{u}_k^*)] + (1-\alpha)\boldsymbol{\lambda}_{k-1}, \tag{14}$$

starting from some initial values $\boldsymbol{\lambda}_{-1}$, with $\alpha \in (0, 1]$ a filter gain. For $\alpha < 1$, the crucial Condition (8) of Assumption 5 is generally not satisfied, and one thus cannot apply the same global convergence analysis to such algorithms. However, this technique of "model filtering" is very similar in essence to the "models with memory" discussed in the trust-region literature (Conn et al., 2000, §9.5), and so it would not be surprising if the analysis of the latter were directly applicable to modifier-adaptation schemes that employed a filter.

Finally, it is important to emphasize that much of the discussion so far has focused on very idealized cases, without considering how the algorithms would behave in real application, where neither accurate function nor derivative values would be available and where numerous other implementation issues could enter to complicate analysis (Quelhas et al., 2013; Bunin et al., 2014). While recent research on trust-region methods has looked into cases with corrupted function values and derivatives (Larson, 2012), it is probably too early for such methods to be directly applicable to many practical real-time optimization problems. Nevertheless, there is no reason to suspect why theory developed for such problems not be equally applicable to both frameworks.

References

Biegler LT, Lang Y, Lin W. Multi-scale optimization for process systems engineering. Comput Chem Eng 2014;60:17–30.

Brdys M, Tatjewski P. Iterative algorithms for multilayer optimizing control. Imperial College Press; 2005.

Bunin GA, François G, Bonvin D. Exploiting local quasiconvexity for gradient estimation in modifier-adaptation schemes. In: 2012 American Control Conference; 2012. p. 2806–11.

Bunin GA, François G, Bonvin D. Implementation techniques for the SCFO experimental optimization framework; 2014, arXiv:1406.3997 [math.OC].

Bunin GA, François G, Srinivasan B, Bonvin D. Input filter design for feasibility in constraint-adaptation schemes. In: 18th World Congress of the International Federation of Automatic Control (IFAC); 2011. p. 5585–90.

Chachuat B, Srinivasan B, Bonvin D. Adaptation strategies for real-time optimization. Comput Chem Eng 2009;33:1557–67.

Conn AR, Gould NIM, Toint PL. Trust-region methods. SIAM; 2000.

Costello S, François G, Bonvin D, Marchetti A. Real-time optimization when the plant and the model have different inputs. In: Dynamics and control of process systems (DYCOPS); 2013.

Faulwasser T, Bonvin D. On the use of second-order modifiers for real-time optimization. In: 19th World Congress of the International Federation of Automatic Control (IFAC); 2014.

François G, Bonvin D. Use of convex model approximations for real-time optimization via modifier adaptation. Ind Eng Chem Res 2013;52:11614–25.

Gao W, Engell S. Iterative set-point optimization of batch chromatography. Comput Chem Eng 2005;29:1401–9.

Larson JM. Derivative-free optimization of noisy functions. University of Colorado; 2012, Ph.D. thesis.

Marchetti A, Chachuat B, Bonvin D. Modifier-adaptation methodology for real-time optimization. Ind Eng Chem Res 2009;48:6022–33.

Marchetti A, Chachuat B, Bonvin D. A dual modifier-adaptation approach for real-time optimization. J Process Control 2010;20:1027–37.

Navia D, Marti R, Sarabia D, Gutirrez G, de Prada C. Handling infeasibilities in dual-modifier methodology for real-time optimization. In: 8th IFAC symposium on advanced control of chemical processes; 2012. p. 537–42.

Quelhas A, Castro N, Pinto J. Common vulnerabilities of RTO implementations in real chemical processes. Can J Chem Eng 2013;91:652–68.

Roberts P. Algorithms for integrated system optimisation and parameter estimation. Electron Lett 1978;14:196–7.

Rodger E. Dual modifier adaptation methodology for the on-line optimization of uncertain processes. McMaster University; 2010, Master's thesis.

Serralunga FJ, Mussati MC, Aguirre PA. Model adaptation for real-time optimization in energy systems. Ind Eng Chem Res 2013;52:16795–810.