17th European Symposium on Computer Aided Process Engineering – ESCAPE17
V. Plesu and P.S. Agachi (Editors)

Large-scale Nonlinear Programming: An Integrating Framework for Enterprise-Wide Dynamic Optimization

Lorenz T. Biegler

Chemical Engineering Department, Carnegie Mellon University, Pittsburgh, PA 15213
E-mail: biegler@cmu.edu

Abstract

Integration of real-time optimization and control with higher level decision making (scheduling and planning) is an essential goal for profitable operation in a highly competitive environment. While integrated large-scale optimization models have been formulated for this task, their size and complexity remains a challenge to many available optimization solvers. On the other hand, recent development of powerful, large-scale solvers leads to a reconsideration of these formulations, in particular, through development of efficient large-scale barrier methods for nonlinear programming (NLP). As a result, it is not unrealistic to solve NLPs on the order of a million variables, for instance, with the IPOPT algorithm. More recently, IPOPT has been embodied as an object oriented code that exploits problem structure, takes advantage of parallelism, and allows configuration of different "internal decomposition strategies" without compromising its fast convergence properties. Finally, an NLP sensitivity extension was added to this code that allows the fast approximate solution of perturbed NLP problems. These developments are demonstrated on dynamic optimization formulations that integrate real-time optimization with on-line calculations required by model predictive control. This allows on-line computations to be drastically reduced, even when large nonlinear optimization models are considered.

Keywords: Model Predictive Control, NLP, Sensitivity, On-line Optimization

1. Introduction

For over two decades, *real-time optimization* has evolved as a standard practice in the chemical and petroleum industry. The ability to optimize predictive models provides a major step towards linking on-line performance to higher-level corporate planning decisions. As described in [1,2], these decisions form a hierarchy as seen in the Figure 1, with levels of decision-making that include planning, scheduling, site-wide and real-time optimization, model predictive control and regulatory control. In this pyramid, note that the frequency of decision-making increases from top to bottom, while the impact and importance of decision-making increases from bottom to top. Moreover, mathematical models and optimization problems have been developed for all but the bottom-most level. Planning and scheduling decision models are often characterized by linear models with many discrete decisions. These are usually represented as mixed integer linear programs (MILPs), and occasionally mixed integer nonlinear programs (MINLPs) that capture key nonlinear elements. On the other hand, site-wide and real-time optimization require nonlinear process models which usually reflect steady-state performance of the plant, while model predictive control (MPC) is often characterized by linear dynamic models. In severely nonlinear processes, nonlinear model predictive control (NMPC) is needed and this requires a more expensive modeling effort to solve dynamic optimization problems on-line [3].

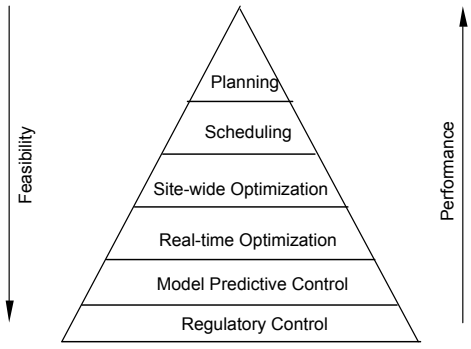

Figure 1. Decision Hierarchy for Enterprise Wide Optimization

Communication and interaction among levels requires that decisions made at higher levels be *feasible* at lower levels. Moreover, the *performance* described by lower level models must be reflected accurately in decisions made at higher levels. Clearly, the strongest communication and interaction is through direct integration of optimization formulations between two or more levels. Such integration has been described for planning and scheduling [1]. Similarly, site-wide and real-time levels can be integrated through compatible steady-state optimization models. However, integrated real-time optimization and MPC often suffers inconsistencies due to mismatch of linear dynamic and steady state nonlinear models, and also because of conflicting objectives [4].

To resolve these two levels, we explore simultaneous dynamic optimization formulations that combine these two levels. In the next section we briefly describe the optimization problem and summarize the *direct transcription* approach used to solve it. Section 3 then discusses an on-line NMPC strategy based on NLP sensitivity, which easily extends to more general dynamic optimization schemes. In this section we also present a recent case study on grade transition for a realistic LDPE process. Section 4 summarizes these concepts and outlines areas for future work.

2. Fast, Large-scale Dynamic Optimization

Consider the following dynamic optimization problem:

$$\begin{aligned} &Min \quad \Phi(z(t_f), y(t_f), u(t_f), p) \\ &s.t. \quad F(dz/dt; z(t); y(t); u(t); t; p) = 0, z(0) = z_0 \\ &G_s(z(t_s); y(t_s); u(t_s); t_s; p)) = 0 \\ &z^L \le z(t) \le z^U, y^L \le y(t) \le y^U \\ &u^L \le u(t) \le u^U, p^L \le p \le p^U, t_f^L \le t_f \le t_f^U \end{aligned} \tag{1}$$

where Φ is a scalar objective function at final time, t_f, F is the differential-algebraic equation (DAE) model, assumed to be index 1, G_s are additional point conditions at times t_s, $z(t)$ are differential state profile vectors, $y(t)$ are algebraic state profile vectors, $u(t)$ are control state profile vectors and p is a time-independent parameter vector.

In this formulation, the continuous time problem is converted into an NLP by approximating all of the state and control profiles as a family of polynomials on finite elements. This is known as the *direct simultaneous* or *direct transcription* approach, where the discretizations stem from implicit Runge-Kutta formulae. Here monomial representations are used for the differential variables, and control and algebraic profiles are approximated using Lagrange polynomials. Once discretized, a large NLP is constructed of the form:

$$Min\ f(x),\ s.t.\ c(x) = 0,\ x_L \le x \le x_U \tag{2}$$

where x represents all discretized variables and $c(x) = 0$ represents the discretized equations.

Direct transcription methods offer a number of advantages for challenging dynamic optimization problems. First, control variables can be discretized at the same level of accuracy as the differential and algebraic state variables. The KKT conditions of (2) can be shown to be consistent with the variational

conditions of (1) and finite elements allow for discontinuities in control profiles. Next, collocation formulations allow problems with unstable modes to be handled in an efficient and well-conditioned manner, as the NLP formulation inherits stability properties of boundary value solvers. Finally, dynamic optimization using collocation methods has been used for a wide variety process applications including batch process optimization, batch distillation, crystallization, dynamic data reconciliation and parameter estimation, nonlinear model predictive control, polymer grade transitions and process changeovers, and reactor design and synthesis. A review of this approach can be found in [5].

On the other hand, these strategies lead to large nonlinear programs, often with many thousands of variables, constraints and, possibly, degrees of freedom. These can be addressed efficiently by the IPOPT algorithm [6]. This algorithm follows a barrier approach, where the bound constraints in (2) are replaced by logarithmic barrier terms and added to the objective function to give the equality-constrained NLP:

$$\begin{aligned} \underset{x\in\Re^n}{Min} \quad & \varphi_\mu(x) = f(x) - \mu[\sum_{i=1}^{n}\ln(x - x_L) + \sum_{i=1}^{n}\ln(x_U - x)] \\ \text{s.t.} \quad & c(x) = 0 \end{aligned} \tag{3}$$

Solving a sequence of these problems with the barrier parameter, $\mu \rightarrow 0$ allows us to recover the solution of (2). Problem (3) is solved by writing the KKT conditions of (3) in primal-dual form:

$$\begin{aligned} \nabla f(x) + A(x)\lambda - \nu^U + \nu^L &= 0 \\ c(x) &= 0 \\ V_U(x_U - x) &= \mu e \\ V_L(x - x_L) &= \mu e \end{aligned} \tag{4}$$

where $A(x) = \nabla c(x)$ and λ, ν_L, ν_U are the KKT multipliers for the equality constraints and variable bounds in (2). IPOPT solves (4) by applying Newton's method and solving the sparse, symmetric linear system at each iteration, ℓ.

$$\begin{bmatrix} W_\ell + \Sigma & A_\ell \\ A_\ell^T & 0 \end{bmatrix} \begin{bmatrix} \Delta x_\ell \\ \lambda_\ell \end{bmatrix} = -\begin{bmatrix} \nabla\varphi_\mu(x_\ell) \\ c(x_\ell) \end{bmatrix}. \tag{5}$$

Here $V_U = diag\{\nu_U\}$ and $V_L = diag\{\nu_L\}$, W_ℓ is the Hessian of the Lagrange function, $\Sigma = (X_\ell - X_L)^{-1}V_L + (X_U - X_\ell)^{-1}V_U$, ($X = diag(x)$) is the barrier term correction and $\varphi_\mu(x_\ell)$ is the barrier function from (3). As a result, local convergence of the Newton method is fast and global convergence of IPOPT is

ensured by a novel filter line search strategy. More information on IPOPT, along with open source code, can be found in [6]. Moreover IPOPT has been applied to a number of direct transcription problems including an NMPC study applied to the unstable Tennessee Eastman process [7]. Here the on-line calculations required only 5-10 CPUs for each of a long series of NLPs with about 11000 variables and 660 degrees of freedom.

3. On-line Dynamic Optimization

Efficient NLP formulations and fast NLP solvers are not enough for large-scale on-line dynamic optimization. Process models in (1) continue to increase in size while on-line calculations remain time-critical with fixed intervals to update the model, as seen in Figure 2. Instead, we note that on-line solution of (2) is parametric in the state variable estimates (z_0) and this solution often varies only slightly between measurements $y(t_k)$ and $y(t_{k+1})$. As a result, we can apply two concepts:

- The on-line solution of (2) at t_{k+1} can be approximated with a perturbation of the previously converged solution, using the initial condition z_0 inferred from $y(t_{k+1})$
- Once an approximate solution is calculated at t_{k+1}, and u_{k+1} is injected into the plant, a fully converged solution can be determined *in background,* say *between* t_{k+1} and t_{k+2}.

Both of these *real-time iteration* concepts have been used to advantage in [2, 8, 10]. In particular, using direct transcription and IPOPT leads to an on-line solution approximated by an *NLP sensitivity step* from the primal-dual algorithm. Such steps are well characterized, lead to first order accurate solutions, and require only a fast, direct calculation from a sparse linear system [9]. As a result, they can lead to reduction of on-line optimization costs by *at least two orders of magnitude*.

As detailed in [10], the real-time iteration approach can be implemented in a number of ways. The *direct approach* is to calculate, from the solution of (1) at t_k, an approximate solution that perturbs the initial conditions of the differential variables, z_0, inferred from the measurement $y(t_{k+1})$. This approach solves the system, $K\Delta v = r$, where K is the matrix in (5), the right hand side r represents the perturbation of the initial condition z_0 and Δv is the change in the (primal and dual) approximate solution from t_k to t_{k+1}. Because this approach applies the matrix factorized from (5), it is quite fast. However, it retains the active constraint set from solution at t_k for t_{k+1} (which is incorrect and should be shifted by one step). This leads to a poor approximation of the optimal solution.

In contrast, the *second approach* recognizes the shift in the active set. From the measurement $y(t_{k+1})$, it implicitly performs a perturbation on z_1 (at t_{k+1}) and

back-calculates the value of z_0 (at t_k) to achieve this. This approach ensures the correct active set by requiring an additional equation $z_1 - z_1(t_{k+1}) = 0$, a slack variable s for z_0, and solution of the augmented system:

$$\begin{bmatrix} K & E_1 \\ E_2 & 0 \end{bmatrix} \begin{bmatrix} \Delta v \\ s \end{bmatrix} = - \begin{bmatrix} 0 \\ z_1^* - z_1(t_{k+1}) \end{bmatrix} \tag{6}$$

where z_1^* is from the previous solution at t_k. Using the previous factorization of K, (6) can be solved quickly with a Schur complement operation.

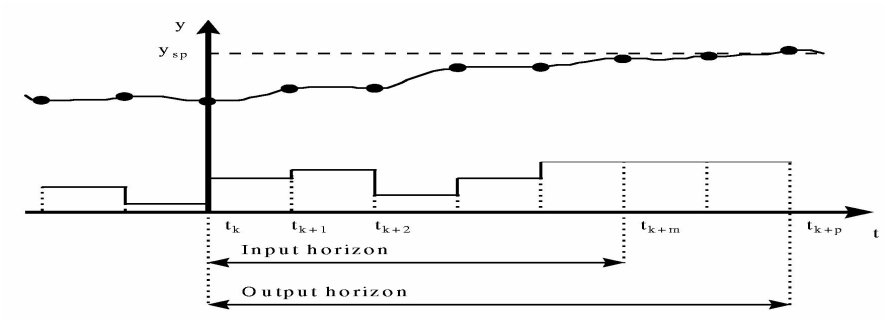

Figure 2. Time Horizon for On-line Dynamic Optimization

IPOPT has features that allow both approaches and it exploits the similarities between the Newton and sensitivity steps. After solving the NLP in background, it retains the previous matrix factorization from (5) and finds the approximate solution of the perturbed problem with a single back-solve in (6). More details of this approach are given in [10].

3.1. LDPE Case Study

The above computational framework is demonstrated on a grade transition scenario for a large-scale high-pressure low density polyethylene (LDPE) process described in [10] and shown in Figure 3. The on-line grade transition combines NMPC with dynamic real-time optimization to minimize the transition time and reduce off-spec product. The process model consists of a simplified polymerization reactor, a number of compression stages and separators, and several time delays. Product quality is estimated from the butane content in the low pressure recycle stream, while butane feed and the purge stream flowrate are the control variables. In addition, uncertainties in the time delays lead to deviations of the measurements from the process model. The resulting dynamic optimization problem (1) contains 294 differential and 64

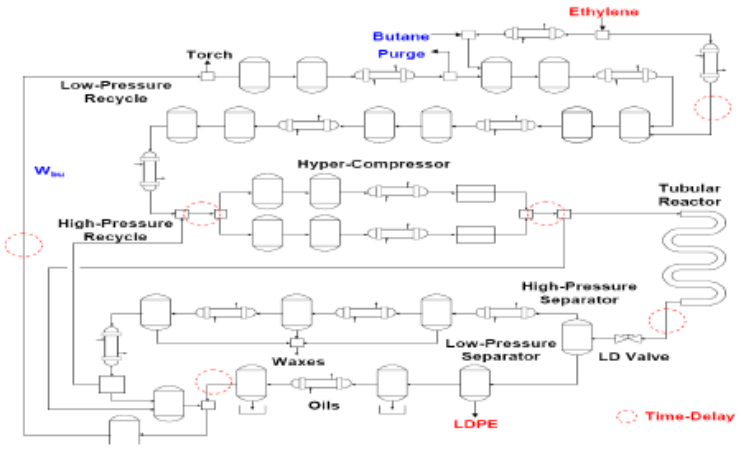

Figure 3. LDPE Process Flowsheet

algebraic state variables. Applying direct transcription leads to a large-scale NLP problem (2) with 27,135 constraints, 9360 lower bounds, 9360 upper bounds and 30 degrees of freedom. With a 3.0 GHz Pentium IV processor using 1 GB of RAM, complete solution of this problem requires about 350 CPUs, 34 CPUs for the factorization of matrix K and 0.95 CPUs for a single back-solve with this matrix. In contrast, the approximated solution requires an on-line calculation *of only 1.0 CPUs*. The results of the approximate and complete solution profiles are shown in Figure 4. The output product quality shown in the top graphic shows identical performance for both approaches while the control variables in the middle and bottom graphic are quite similar, despite a drastic computational reduction with the real-time iteration approach.

4. Conclusions

Integration of optimal decision-making for operations is an important task for today's highly competitive chemical industry. In particular, tying higher-level decisions to optimal on-line operations is essential. This paper considers the challenges at *real-time optimization* and *model predictive control* levels as they both comprise large-scale NLP problems, often with conflicting models and objectives. We show that *direct transcription* and fast NLP solvers, like *IPOPT*, allow the integration of real-time optimization and model predictive control through a single dynamic optimization formulation. In addition, concepts from *real-time iteration* allow on-line calculations to be drastically reduced with almost all of the optimization calculations performed in a background step. This is shown on a large LDPE process where on-line optimization computations were reduced by over 300 times, with negligible loss of performance.

Future work deals with application of real-time iteration concepts to moving horizon estimation and the optimization of on-line multi-stage operations. These represent challenging large-scale NLPs which require specialized decomposition strategies that can be exploited by IPOPT. In addition, a particular stability result of *real-time iteration* has been developed in [8]. We intend to explore these nominal and robust stability properties further through a Lyapunov analysis along with recently developed robustness properties for Nonlinear Model Predictive Control.

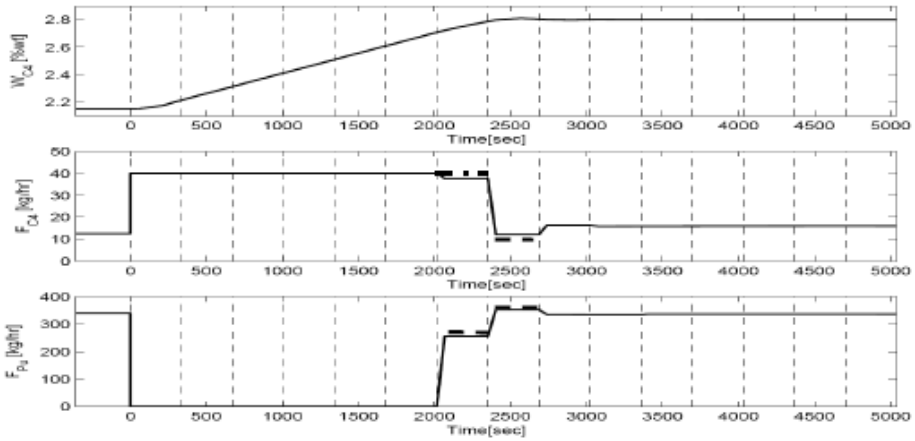

Figure 4. Output (top) and control trajectories for complete dynamic optimization (solid lines) and approximate optimization (dashed lines)

References

1. Grossmann, I. E., *AIChE J.,* 51, 7, p. 1846, 2005
2. Kadam, J., W. Marquardt, in *Assessment and Future Directions of NMPC*, Findeisen, Allgöwer, Biegler (eds.), to appear. Springer, Berlin, 2006
3. R. D. Bartusiak, in *Assessment and Future Directions of NMPC*, Findeisen, Allgöwer, Biegler (eds.), to appear. Springer, Berlin, 2006
4. Yip, W. S. and T. Marlin, *Comp. Chem. Engr.*, 28, 267, 2004
5. Kameswaran, S. and L. T. Biegler, *Comp. Chem. Engr.*, 30, p. 1560, 2006
6. Wächter, A., and L. T. Biegler, *Math. Prog.*, 106, 1, pp. 25-57, 2006, https://projects.coin-or.org/Ipopt
7. Jockenhövel, T., L. Biegler, A.Wächter, *Comp. Chem. Engr.*, **27**, 1513, 2003
8. M. Diehl, H.G. Bock, J.P. Schlöder. *SIAM J. Cont. Opt.*, 43, 5:1714, 2005.
9. V. Fiacco. *Introduction to Sensitivity and Stability Analysis in Nonlinear Programming*. Academic Press, New York, 1983.
10. 10.Zavala, V., C. D. Laird and L. T. Biegler, Fast NMPC IFAC Workshop, Grenoble (2006), http://dynopt.cheme.cmu.edu/papers/preprint/paper22.pdf