2018-12-01

Large-Scale Non-Linear Dynamic Optimization For Combining Applications of Optimal Scheduling and Control

Logan Daniel Beal
Brigham Young University

Large-Scale Non-Linear Dynamic Optimization for Combining Applications of Optimal Scheduling and Control

Logan Daniel Beal

A dissertation submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Chemical Engineering

Brigham Young University

ABSTRACT

Large-Scale Non-Linear Dynamic Optimization
for Combining Applications of Optimal
Scheduling and Control

Logan Daniel Beal
Department of Chemical Engineering, BYU
Doctor of Philosophy

Optimization has enabled automated applications in chemical manufacturing such as advanced control and scheduling. These applications have demonstrated enormous benefit over the last few decades and continue to be researched and refined. However, these applications have been developed separately with uncoordinated objectives. This dissertation investigates the unification of scheduling and control optimization schemes. The current practice is compared to early-concept, light integrations, and deeper integrations. This quantitative comparison of economic impacts encourages further investigation and tighter integration.

A novel approach combines scheduling and control into a single application that can be used online. This approach implements the discrete-time paradigm from the scheduling community, which matches the approach of the control community. The application is restricted to quadratic form, and is intended as a replacement for systems with linear control. A novel approach to linear time-scaling is introduced to demonstrate the value of including scaled production rates, even with simplified equation forms. The approach successfully demonstrates significant benefit.

Finally, the modeling constraints are lifted from the discrete-time approach. Time dependent constraints and parameters (like time-of-day energy pricing) are introduced, enabled by the discrete-time approach, and demonstrate even greater economic value. The more difficult problem calls for further exploration into the relaxation of integer variables and initialization techniques for faster, more reliable solutions. These applications are also capable of replacing both scheduling and control simultaneously.

A generic CSTR application is used throughout as a case study on which the integrated optimization schemes are implemented. CSTRs are a common model for applications in most chemical engineering industries, from food and beverage, to petroleum and pharmaceuticals. In the included case study results, segregated control and scheduling schemes are shown to be 30+% less profitable than fully unified approaches during operational periods around severe disturbances. Further, inclusion of time-dependent parameters and constraints improved the open-loop profitability by an additional 13%.

Keywords: optimization, advanced control, scheduling, nonlinear programming, model predictive control

CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

ML	Machine Learning
ANN	Artificial Neural Networks
RTO	Real-Time Optimization
DRTO	Dynamic Real-Time Optimization
MDO	Multi-Disciplinary Optimization
LP	Linear Programming
QP	Quadratic Programming
NLP	Non-Linear Programming
MILP	Mixed-Integer Linear Programming
MINLP	Mixed-Integer Non-Linear Programming
QPQC	Quadratic Programming with Quadratic Constraints
AML	Algebraic Modeling Language
AD	Automatic (or Algorithmic) Differentiation
DAE	Differential and Algebraic Equations
ODE	Ordinary Differential Equations
PDE	Partial Differential Equations
MPC	Model Predictive Control
NMPC	Nonlinear Model Predictive Control
EMPC	Economic Model Predictive Control
MHE	Moving Horizon Estimation
SQP	Sequential Quadratic Programming
KKT	Karush-Kuhn-Tucker
BFGS	Broyden–Fletcher–Goldfarb–Shanno
CSTR	Continuously Stirred Tank Reactor
DO	Dynamic Optimization
MIDO	Mixed Integer Dynamic Optimization
MPEC	Mathematical Programming with Equilibrium Constraints
MPCC	Mathematical Programming with Complementarity Constraints
MV	Manipulated Variable
CV	Controlled Variable
SC	Scheduling and Control
DR	Demand Response
SISO	Single-Input Single-Output
LQR	Linear Quadratic Regulator
IP	Interior Point
AS	Active Set
SOC	Second-Order Correction
FBR	Fluidized Bed Reactor
ASU	Air Separation Unit
PFR	Plug Flow Reactor

CHAPTER 1. INTRODUCTION

1.1 Introduction

Computational power has increased dramatically in recent decades. In addition, there are new architectures for specialized tasks and distributed computing for parallelization. Computational power and architectures have expanded the capabilities of technology to new levels of automation and intelligence with rapidly expanding artificial intelligence capabilities and computer-assisted decision processing [2]. These advancements in technology have been accompanied by a growth in the types of mathematical problems that applications solve. Lately, machine learning (ML) has become the must-have technology across all industries, largely inspired by the recent public successes of new artificial neural network (ANN) applications. Increased computational resources have provoked the optimization community to expect more from computers. Examples such as extending Real-Time Optimization (RTO) to Dynamic Real-Time Optimization (DRTO) [3], energy-grid optimization [4], Multidisciplinary Design Optimization (MDO) [5], enterprise-wide optimization [6], and optimization under uncertainty [7] present much harder problems that yield more valuable solutions.

A subset of optimization that is particularly relevant to chemical engineering is termed dynamic optimization. This field encompasses all problems with time dependence. Applications have ranged from chemical production planning [8], energy storage systems [9,10], polymer grade transitions [3], cryogenic air separation [11], unmanned aerial vehicle path planning [12], and dynamic process model parameter estimation in the chemical industry [13].

As the capacity to solve increasingly larger problems improves, researchers continue to find more applications to optimize. Rather than simplifying and approximating problems to linear programming (LP) or quadratic programming (QP) for a solution that is “good enough”, practitioners have triggered a new era of larger-scale and combined optimization. Areas that had been previously dominated by heuristics can now be modeled and optimized. Problems that had been optimized through a series of simplifications and divisions to layers can now be tackled more directly. One such field that is relevant to chemical engineering manufacturing is the desegregation of heretofore separate scheduling and control optimizations.

Section 1.2 introduces the field of combining scheduling and control. The principle accompanying technologies that enable these advancements in large-scale optimization, such as combining scheduling and control, are algebraic modeling languages (AML) and solvers. AMLs and solvers each take the problem posed and transform it to a new paradigm to facilitate computational solutions, as shown in Figure 1.1. Understanding these technologies enables more complete and robust approaches to solving novel, difficult optimization problems. Section 1.3 addresses modeling frameworks that simplify the solver-engineer interface, enabling simpler, more robust problem solving. Section 1.4 explains the approach and core algorithms for state-of-the-art nonlinear solvers. Sections 1.5-1.7 then revisit combining scheduling and control and set up the remainder of the dissertation.

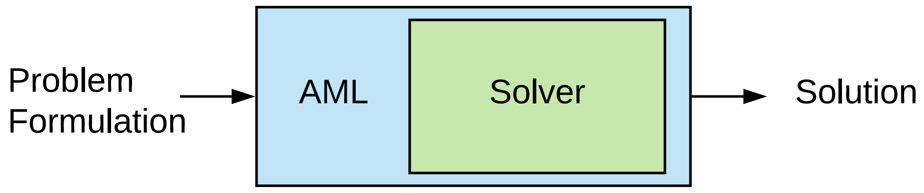


Figure 1.1: The interaction of tools in optimization problems.

1.2 Combining Scheduling and Control

Production scheduling and advanced control are terms which describe efforts to optimize chemical manufacturing operations. Production scheduling seeks to optimally pair production resources with production demands to maximize operational profit. Advanced controls seek to optimally control a chemical process to observe environmental and safety constraints and to drive operations to the most economical conditions. Model predictive advanced controls use process models to make predictions into a future horizon based on possible control moves to determine the optimal sequence of control moves to meet an objective, such as reaching an operational set point. In a multi-product continuous chemical process, steady-state operational set points or desired operational conditions over a future time horizon are determined by the production schedule, determining at which times, in what amounts, and in which sequence certain products should be produced.

Figure 1.2 illustrates the current industrial implementation of these applications. The scheduler considers economic information, creates an optimal schedule, and passes ordered product setpoints to the controller. The controller has no knowledge of economics and cannot communicate back to the scheduler. However, unlike the scheduler, the controller has knowledge of system dynamics and receives feedback from the physical process.

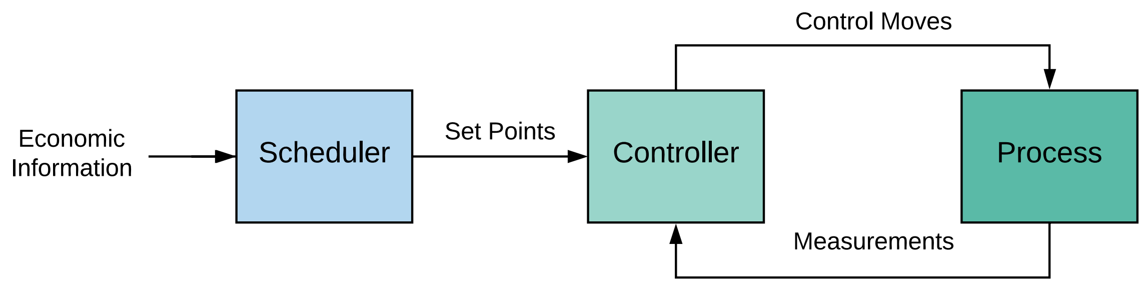


Figure 1.2: Interactions and data flow among optimization applications.

As scheduling and advanced control are closely interrelated, both seeking to optimize chemical manufacturing efficiency over future time horizons, their integration has been the subject of significant recent investigation. Multiple review articles have been published on the integration of scheduling and control [14–17]. As schedules are informed of process dynamics as dictated by control structure and process nonlinearities, schedules produced become more aligned with actual process operations and schedule efficacy improves [18]. Conversely, when scheduling and advanced control are separated, coordination of closed-loop responses to process disturbances is lost, unrealistic set points may be passed from scheduling to advanced controls, and advanced control may seek to drive the process to sub-optimal operating conditions due to a lack of communication [16, 18, 19]. In the presence of a process disturbance, for example, advanced controls may attempt to return to a set point determined prior to the disturbance, whereas a recalculation of the schedule from the measured disturbed state, with knowledge of process dynamics and controller behavior, may show a different schedule and set point to be most economical [18, 20].

One setback to the integration of scheduling and control is computational difficulty. Advanced controls, particularly model predictive controls, utilize dynamic process models in dynamic optimization problems, forming LP or nonlinear programming (NLP) optimization problems (depending on the type of model used) with high detail over short horizons. Scheduling involves discrete or binary decisions, such as assigning particular products to production at given times, over long horizons. This gives rise to mixed-integer programming problems for scheduling. When scheduling and control are combined, the computational burden of mixed-integer programming is combined with the LP or NLP dynamic optimization problems. This further requires blending the dynamics typically limited to the short horizons of control, with the long time scales of scheduling, as shown in Figure 1.3. Additionally, dynamic optimization control problems are not required to be solved only once, as in an iteration of advanced on-line control, but for each grade transition during a production schedule. The integrated scheduling and control (ISC) problem has been shown to be computationally difficult, and much research has been invested in decomposition and reduction of computational burden for the problem [21–25].

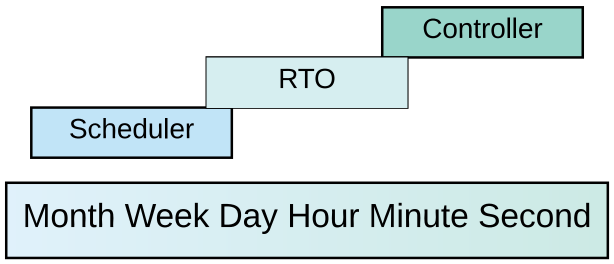


Figure 1.3: Time scales of manufacturing optimization applications.

Reduction of the computational burden of integrated scheduling and control is especially important for enabling on-line implementations. It has been shown repeatedly in simulation that closed-loop on-line implementations of integrated scheduling and control are critical to recalculating optimal scheduling and control when faced with process disturbances [18, 20]. Additionally, it has been demonstrated that on-line closed-loop integrated scheduling and control is vital to optimally responding to variable market conditions, including demand fluctuations and price fluctuations [18, 26, 27]. As mentioned in review articles on integrated scheduling and control, a key motivator for integration is the reduced time-scale at which market conditions fluctuate [14, 15]. A reduced time-scale for market fluctuations implies that the time-scale at which the economically optimal schedule and associated optimal control profile fluctuates is likewise reduced. Thus, on-line recalculation to respond to market condition updates is critical to integrated scheduling and control, and computational burden reduction to enable such implementation is a salient topic for researchers.

On-line responsiveness to volatile market conditions can improve process economics by updating or changing the existing schedule [18]. The majority of integrated scheduling and control formulations have used cyclic schedules [20–24, 28–36]. However, it has been suggested that a *dynamic* cyclic schedule may improve process economics [37]. Beal *et al.* suggest that a dynamic cyclic schedule can increase the flexibility of scheduling beyond the rigidity of a cyclic product grade wheel. A dynamic cyclic schedule can dynamically change the sequence and duration of production of products on the grade wheel based on process disturbances, sudden surges in demand for specific products, or time-dependent constraints such as operator availability for equipment handling. Economic benefit from dynamic cyclic scheduling has been demonstrated in previous work [18, 20]. Recent developments in integrated scheduling and control with discrete representations of time do away with the idea of a cyclic schedule as the schedule is determined as a sum total of the binary production variables at each discrete point in time during a prediction horizon [38, 39]. In these works, the number of products manufactured, the selection of manufactured products, sequence, and timing are all solved simultaneously with process dynamic models, resulting in a computationally heavy formulation. Evidence of benefit from noncyclic scheduling has also been demonstrated in other fields, such as cluster-tool robot scheduling for semiconductor fabrication [40–45].

1.3 Modeling Tools

Algebraic modeling languages (AML) facilitate the interface between advanced solvers and human users. High-end gradient-based solvers require extensive information about the problem, including variable bounds, constraint functions and bounds, objective functions, and first and second derivatives of the functions, all in consistent array format. AMLs simplify the process by allowing the model to be written in a simple, intuitive format. The modeling language accepts a model (constraints) and objective to optimize. The AML handles bindings to the solver binary, maintains the required formatting of the solvers, and exposes the necessary functions. The necessary function calls include constraint residuals, objective function values, and derivatives. Most modern modeling languages leverage automatic differentiation (AD) [46] to facilitate exact gradients without explicit derivative definition by the user.

In general, an AML is designed to solve a problem in the following form:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & c(x) = 0 \\ & d(x) \leq 0 \\ & x_L \leq x \leq x_U \end{aligned} \tag{1.1}$$

where x is a vector of the decision variables, with upper and lower bounds x_U and x_L, respectively. $x_L \in (-\infty, \infty)^n$ and $x_U \in (-\infty, \infty)^n$ with $x_L^i \leq x_U^i$. The objective function $f : \mathbb{R}^n \to \mathbb{R}$ and the equality constraints $c : \mathbb{R}^n \to \mathbb{R}^m$, where $m \leq n$, are twice continuously differentiable.

1.3.1 Dynamic Optimization

Dynamic optimization is a unique subset of optimization algorithms that pertain to systems with time-based differential equations. Dynamic optimization problems extend algebraic problems of the form in Equation 1.1 to include the possible addition of the differentials $\frac{dx}{dt}$ in the objective function and constraints, as shown in Equation 1.2.

$$\begin{aligned}
\underset{x}{\text{minimize}} \quad & f\left(x, \frac{\mathrm{d}x}{\mathrm{d}t}\right) \\
\text{subject to} \quad & c\left(x, \frac{\mathrm{d}x}{\mathrm{d}t}\right) = 0 \\
& d\left(x, \frac{\mathrm{d}x}{\mathrm{d}t}\right) \leq 0 \\
& x_L \leq x \leq x_U
\end{aligned} \tag{1.2}$$

Differential algebraic equation (DAE) systems are solved by discretizing the differential equations to a system of algebraic equations to achieve a numerical solution. Some modeling languages are capable of natively handling DAEs by providing built-in discretization schemes. The DAEs are typically solved numerically and there are a number of available discretization approaches. Historically, these problems were first solved with a direct shooting method [47]. Direct shooting methods are still used and are best suited for stable systems with few degrees of freedom. Direct shooting methods eventually led to the development of multiple shooting, which provided benefits such as parallelization and stability [48]. For very large problems with multiples degrees of freedom, "direct transcription" (also known as "orthogonal collocation on finite elements") is the state-of-the-art method [49]. Some fields have developed other unique approaches, such as pseudospectral optimal control methods [50].

Dynamic optimization problems introduce an additional set of challenges. Many of these challenges are consistent with those of other forms of ordinary differential equation (ODE) and partial differential equation (PDE) systems; only some challenges are unique to discretization in time. These challenges include handling stiff systems, unstable systems, numerical versus analytical solution mismatch, scaling issues (especially with increased discretization), the number and location in the horizon of discretization points, and the optimal horizon length. Some of these challenges, such as handling stiff systems, can be addressed with the appropriate discretization scheme. Other challenges, such as the necessary precision of the solution and the location of discretizations of state variables, are better handled by a knowledgeable practitioner to avoid excessive computation.

Popular practical implementations of dynamic optimization include model predictive control (MPC) [51] (along with its nonlinear variation NMPC [52] and the economic objective alternative EMPC [53]), moving horizon estimation (MHE) [54] and dynamic real-time optimization (DRTO) [55]. Each of these problems is a special case of Equation 1.2 with a specific objective function. For example, in MPC, the objective is to minimize the difference between the controlled variable set point and model predictions, as shown in Equation 1.3.

$$\min_{x} \quad \left\| x - x_{sp} \right\| \tag{1.3}$$

where x is a state variable and x_{sp} is the desired set point or target condition for that state. The objective is typically a 1-norm, 2-norm, or squared error. EMPC modifies MPC by maximizing profit rather than minimizing error to a set point, but uses the same dynamic process model, as shown in Equation 1.4.

$$\max_{x} \quad \text{Profit} \tag{1.4}$$

MHE adjusts model parameters to minimize the difference between measured variable values (x_{meas}) and model predictions (x), as shown in Equation 1.5.

$$\min_{x} \quad \left\| x - x_{meas} \right\| \tag{1.5}$$

1.3.2 Differential Equations in Optimization

Including differential equations is problems solved by optimizers designed to work with only algebraic contraints presents a unique challenge. There are two main approaches to this issue: sequential and simultaneous techniques.

Sequential

Sequential methods separate the problem in Equation 1.2 into the standard algebraic optimization routine (Equation 1.1) and a separate differential equation solver, where each problem is solved sequentially. This method is popular in fields where the solution of differential equations is extremely difficult. By separating the problems, the simulator can be fine-tuned, or wrapped in a "black box." Since the sequential approach is less reliable in unstable or ill-conditioned problems, it is often adapted to a "multiple-shooting" approach to improve performance. One benefit of the sequential approach is a guaranteed feasible solution of the differential equations, even if the optimizer fails to find an optimum.

Simultaneous

The simultaneous approach, or direct transcription, minimizes the objective function and resolves all constraints (including the discretized differential equations) simultaneously in the large-scale and sparse solvers by transforming the problem to algebraic form (see Equation 1.6). Thus, if the solver terminates without reaching optimality, it's likely that the equations are not satisfied and the dynamics of the infeasible solution are incorrect—yielding the solution worthless rather than just suboptimal. However, since simultaneous approaches do not waste time accurately simulating dynamics that are thrown away in intermediary iterations, this approach tends to be faster for large problems with many degrees of freedom [56]. A common discretization scheme for this approach is orthogonal collocation on finite elements. Orthogonal collocation represents the state and control variables with polynomials inside each finite element. This is a form of implicit Runga-Kutta methods, and thus it inherits the benefits associated with these methods, such as stability. Simultaneous methods require efficient large-scale NLP solvers and accurate problem information, such as exact second derivatives, to perform well.

$$\min_{u,z} \quad \sum_{i}^{n} J(z_i, u_i) \tag{1.6a}$$

$$0 = f(z_i, u_i) \;\forall\, i \in n \tag{1.6b}$$

$$0 \leq g(z_i, u_i) \;\forall\, i \in n \tag{1.6c}$$

$$\text{Collocation Equations} \tag{1.6d}$$

where n is the number of time points in the discretized time horizon, $z = \left[\frac{\mathrm{d}x}{\mathrm{d}t}, x\right]$ is the combined state vector, and the collocation equations are added to relate differential terms to the state values. The collocation equations and derivation are detailed in [57].

Semi-Sequential Intermediates

Most modeling languages only include standard variables and constraints, where all algebraic constraints and their associated variables are solved implicitly through iterations of the optimizer. A unique variable type that applies the benefits of a sequential approach to algebraic constraints is available in some packages and are termed Intermediates. Intermediates, and their associated equations, are like variables except that they are defined and solved explicitly and successively substituted at every solver function call. Intermediate values, first derivatives, and second derivatives are substituted into other successive Intermediates or into the implicit equations. This is done outside of the solver in order to reduce the number of algebraic variables while maintaining the readability of the model.

In very large-scale problems, removing a portion of variables from the matrix math of implicit solutions can reduce matrix size, keeping problems within the efficient bounds of hardware limitations. This is especially the case with simultaneous dynamic optimization methods, where a set of model equations are multiplied over all of the collocation nodes. For each variable reduced in the base model, that variable is also eliminated from every collocation node. Intermediate variables essentially blend the benefits of sequential solver approaches into simultaneous methods.

1.3.3 Available Software Packages

There are many software packages and modeling languages currently available for optimization and optimal control. This section, while not a comprehensive comparison, attempts to summarize some of the distinguishing features of each package.

Pyomo [58] is a Python package for modeling and optimization. It supports automatic differentiation and discretization of DAE systems using orthogonal collocation or finite-differencing. The resulting nonlinear programming (NLP) problem can be solved using any of several dozen AMPL Solver Library (ASL) supported solvers.

JuMP [59] is a modeling language for optimization in the Julia language. It supports solution of linear, nonlinear, and mixed-integer problems through a variety of solvers. Automatic differentiation is supplied, but as of this writing, JuMP does not include built-in support for differential equations.

Casadi [60] is a framework that provides a symbolic modeling language and efficient automatic differentiation. It is not a dynamic optimization package itself, but it does provides building blocks for solving dynamic optimization problems and interfacing with various solvers. Interfaces are available in MATLAB, Python, and C++.

GAMS [61] is a package for large-scale linear and nonlinear modeling and optimization with a large and established user base. It connects to a variety of commercial and open-source solvers, and programming interfaces are available for it in Excel, MATLAB, and R. Automatic differentiation is available.

AMPL [62] is a modeling system that integrates a modeling language, a command language, and a scripting language. It incorporates a large and extensible solver library, as well as fast automatic differentiation. AMPL is not designed to handle differential equations. Interfaces are available in C++, C#, Java, MATLAB, and Python.

The gProms package [63] is an advanced process modeling and flow-sheet environment with optimization capabilities. An extensive materials property library is included. Dynamic optimization is implemented through single and multiple shooting methods. The software is used through a proprietary interface designed primarily for the process industries.

JModelica [64] is an open-source modeling and optimization package based on the Modelica modeling language. The platform brings together a number of open-source packages, providing ODE integration through Sundials, automatic differentiation through Casadi, and NLP solutions through IPOPT. Dynamic systems are discretized using both local and pseudospectral collocation methods. The platform is accessed through a Python interface.

ACADO [65] is a self-contained toolbox for optimal control. It provides a symbolic modeling language, automatic differentiation, and optimization of differential equations through multiple shooting using the built in QP solver. Automatic C++ code generation is available for online predictive control applications, though support is limited to small to medium-sized problems. Interfaces are available in MATLAB and C++.

DIDO [66] is an object-oriented MATLAB toolbox for dynamic optimization and optimal control. Models are formulated in MATLAB using DIDO expressions, and differential equations are handled using a pseudospectral collocation approach. At this time, automatic differentiation is not supported.

GPOPS II [67] is a MATLAB-based optimal control package. Dynamic models are discretized using hp-adaptive collocation, and automatic differentiation is supported using the ADiGator package. Solution of the resulting NLP problem is performed using either the IPOPT or SNOPT solvers.

PROPT [68] is an optimal control package built on top of the TOMLAB MATLAB optimization environment. Differential equations are discretized using Gauss and Chebyshev collocation, and solutions of the resulting NLP are found using the SNOPT solver. Derivatives are provided through source transformation using TOMLAB's symbolic differentiation capabilities. Automatic scaling and integer states are also supported. Access is provided through a MATLAB interface.

PSOPT [69] is an open-source C++ package for optimal control. Dynamic systems are discretized using both local and global pseudospectral collocation methods. Automatic differentiation is available by means of the ADOL-C library. Solution of NLPs is performed using either IPOPT or SNOPT.

In addition to those listed above, many other software libraries are available for modeling and optimization, including AIMMS [70], CVX [71], CVXOPT [72], YALMIP [73], PuLP [74], POAMS, OpenOpt, NLPy, and PyIpopt.

GEKKO

As part of the work in this dissertation, a new modeling platform, with special emphasis on dynamic optimization, was released as "GEKKO" [75]. GEKKO fills the role of a typical AML, but extends its capabilities to specialize in dynamic optimization applications. As an AML, GEKKO provides a user-friendly, object-oriented Python interface to develop models and optimization solutions. Python is a free and open-source language that is flexible, popular, and powerful. IEEE Spectrum ranked Python the #1 programming language in 2017. As a Dynamic Optimization package, GEKKO accommodates DAE systems with built-in discretization schemes and facilitates popular applications with built-in modes of operation and tuning parameters. For differential and algebraic equation systems, simultaneous and sequential methods are both built in to GEKKO. Modes of operation include data reconciliation, real-time optimization, dynamic simulation, moving horizon estimation, and nonlinear model predictive control. The back-end compiles the model to an efficient low-level format and performs model reduction based on analysis of the sparsity structure (incidence of variables in equations or objective function) of the model.

1.4 Non-Linear Programming Solvers

In the same manner that understanding physical and organic chemistry enables chemicals engineers to design better manufacturing processes, an understanding of NLP solver algorithms enable better optimization solutions. The paradigms of combined scheduling and control presented in this dissertation become increasingly more beneficial, but also more complex and difficult to solve. Later approaches are specifically tailored to take advantage of the solver methodologies explained in this section.

NLP addresses problems of form in Equation 1.1, repeated here:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & c(x) = 0 \\ & d(x) \leq 0 \\ & x_L \leq x \leq x_U \end{aligned}$$

Sequential Quadratic Programming (SQP) solvers require two phases for efficient solutions. The inner subroutine is a Quadratic Programming (QP) problem solving the Karuhn-Kush Tucker conditions of an approximation of a quadratic objective function and constraints. The solution to the approximation requires an outer subroutine to direct the line search and find the solution to the NLP. Algorithmic implementations and mathematical theory determine the efficiency of the algorithm, especially for large-scale problems. Depending on the specifics of a proposed problem, algorithms perform differently.

Interior-point solvers utilize a barrier function added to the objective function to eliminate inequality constraints. Interior-point solvers use Newton's Method to solve a QP comprised of a quadratically approximated objective function and linearly approximated equality constraints. The algorithm converges as the barrier parameter is relaxed completely [76, 77].

Active-set solvers define a subset of the constraints as "active", and employ on this subset to find the optimal value. Active-set solvers have been implemented in algorithms similar to interior point solvers where an approximated QP is solved using Newton's Method. The solution of the QP produces a new active set for the following iteration. The active set eventually represents the true active set at the solution, and the algorithm converges [78].

Interior point and active set methods excel in different circumstances. Interior point methods can handle problems with over a million variables [2], while excellent active-set SQP methods, such as SNOPT, are best suited for less than a few thousand degrees of freedom [79]. Active set methods are better suited to a warm-started problem while interior point methods can better handle poor initialization [80]. Interior point methods typically require fewer iterations than active set methods for difficult problems. However, each iteration is more difficult because all of the constraints are considered while the active set only includes a portion of the constraints in calculating the step [81]. Several recent papers show the difference in convergence ease and speed. Middleton et al. [82], Ternet *et al.* [83], and Bartlett *et al.* [84] report interior point using between 12.5 and 467 percent the CPU time that active set uses for their problems. Mishra *et al.* [85] report their interior point solver taking significantly less iterations than an active set solver. Clearly these CPU times vary significantly based on a variety of factors including number of variables, initial guess, and problem structure.

SQP solvers are a common active set method and were introduced by Wilson [86]. Determining the active set for each iteration is mostly heuristics. Oberlin *et al.* [87] give a few examples of determining an active set. The reduced-gradient approach to the active set further divides the variables into *nonbasic*, *basic* and *superbasic* sets [88].

All solving schema incorporating approximations require globalization strategies to find the minimum of the actual problem. The solution of the approximated problem provides an accurate local solution, but does not guarantee progression towards the actual minimum. Globalization strategies direct each iteration toward the solution of the actual problem. Notably, filter methods, a globalization strategy for nonlinear programming, were introduced by Fletcher *et al.* [89]. Filter methods have been applied in both active set and interior point methods successfully [77, 90–94].

1.4.1 Basic Interior Point Algorithm

Interior point methods have been widely used for years and are still being improved by researchers today [77, 94–100]. Interior point methods begin by introducing slack variables to Equation 1.1 that allow inequality constraints, $d(x) \leq 0$, to be reformulated into equality constraints:

$$d(x) + s = 0, \quad s \geq 0 \tag{1.7}$$

Since s is a regular, bounded variable, it is treated as part of x. For simplicity, assume x is bounded only on one side (the extension to two-sided bounds is obvious). The *interior point* or *barrier* method then adds a barrier term to the objective function to force x into the interior. The problem then becomes:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \phi(x) := f(x) - \mu \sum_i ln(x_i) \\ \text{subject to} \quad & c(x) = 0 \\ & x \geq 0 \end{aligned} \tag{1.8}$$

The barrier term $\mu \sum_i ln(x_i)$ penalizes the objective function, which slides the solution away from the boundary, beyond which the solution is undefined. This forces each iteration to remain in the feasible region. Forcing a feasible path results in a strength of the interior-point algorithm: better performance with poor initialization.

The barrier parameter μ sequentially decreases toward zero to allow x to reach the boundary without penalty or numerical difficulties. The sequential decreasing highlights a weakness of this method: extra iterations are always required to reach inequality bounds until the barrier is sufficiently reduced.

The Karush-Kuhn-Tucker (KKT) conditions for Problem 1.8 include the following primal-dual equations:

$$\begin{aligned} \nabla\phi(x) + \nabla c(x)\lambda &= 0 \\ c(x) &= 0 \end{aligned} \tag{1.9}$$

However, it is known that $\nabla\phi(x) = \nabla f(x) + \nabla\mu\sum_i ln(x_i)$. Since the barrier term always assumes the same form, its gradient extracts to:

$$\nabla\left(\mu\sum_i ln(x_i)\right) = \mu/x := z$$

We can then modify the KKT conditions to implement z with a linear equation,

$$\begin{aligned}\nabla f(x) + \nabla c(x)\lambda - z = 0\\ c(x) = 0\\ XZe - \mu e = 0\end{aligned} \tag{1.10}$$

Here, $\lambda \in \mathbb{R}^m$ is a vector of Lagrange multipliers for the equality constraints while $z \in \mathbb{R}^n$ serves as Lagrange multipliers for the bound constraints. X and Z represent matrices with x and z down the diagonal, respectively. When $\mu = 0$, these become the KKT conditions of the general case.

For termination and barrier term reduction, the optimality error is defined as:

$$E_\mu := \max\left\{\frac{\|\nabla f(x) + \nabla c(x)\lambda - z\|_\infty}{s_d}, \|c(x)\|_\infty, \frac{\|XZe - \mu e\|_\infty}{s_c}\right\} \tag{1.11}$$

where s_d and s_c are extra parameters to avoid numerical difficulties, especially when the multipliers λ and z are large. With this optimality error, the algorithm terminates successfully when

$$E_0(x_*, \lambda_*, z_*) \leq \varepsilon_{tol} \tag{1.12}$$

The termination condition assumes $\mu = 0$. The optimality error also determines when the barrier parameter μ should be decreased:

$$E_{\mu_j}(x_{*,j+1}, \lambda_{*,j+1}, z_{*,j+1}) \leq \kappa_\varepsilon \mu_j \tag{1.13}$$

where $\kappa_\varepsilon > 0$ is a tuning parameter of the solver. When the conditions of Equation 1.13 are met, μ updates:

$$\mu_{j+1} = \max\left\{\frac{\varepsilon_{tol}}{10}, \min\left\{\kappa_\mu \mu_j, \mu_j^{\theta_\mu}\right\}\right\} \tag{1.14}$$

where $\kappa_\mu \in (0,1)$ and $\theta_\mu \in (1,2)$ are additional tuning parameters. Equation 1.14 ensures μ doesn't decrease enough to cause numerical difficulties while it decreases μ linearly, then superlinearly. This demonstrates the extra iterations required near the end of the interior point method.

Newton Step

Iteration proceeds with Newton steps toward a solution of the KKT conditions (Eq. 1.10). In matrix form, this is presented as:

$$\begin{bmatrix} W_k & A_k & -I \\ A_k^T & 0 & 0 \\ Z_k & 0 & X_k \end{bmatrix} \begin{pmatrix} d_k^x \\ d_k^\lambda \\ d_k^z \end{pmatrix} = -\begin{pmatrix} \nabla f(x_k) + A_k\lambda_k - z_k \\ c(x_k) \\ X_k Z_k e - \mu_j e \end{pmatrix} \tag{1.15}$$

Where $A_k := \nabla c(x_k)$ and W_k is the Hessian $\nabla^2_{xx}\mathscr{L}(x_k, \lambda_k, z_k)$,

$$\mathscr{L}(x, \lambda, z) := f(x) + c(x)^T\lambda - z \tag{1.16}$$

However, instead of solving this nonsymmetric problem, the Newton step can be restructured to solve the symmetric problem shown in Equation 1.17 and later find z using Equation 1.18. In Equation 1.17, $\Sigma_k := X_k^{-1} Z_k$. Despite this restructuring, this linear solve is a weakness of interior point methods because all constraints are always included, even if an inequality constraint is far from the optimal point. Including every constraint makes the matrix larger, slowing down the factorization, which is already a limiting factor in solution speed.

$$\begin{bmatrix} W_k + \Sigma_k & A_k \\ A_k^T & 0 \end{bmatrix} \begin{pmatrix} d_k^x \\ d_k^\lambda \end{pmatrix} = -\begin{pmatrix} \nabla\phi_{\mu_j}(x_k) + A_k\lambda_k \\ c(x_k) \end{pmatrix} \tag{1.17}$$

$$d_k^z = \mu_j X_k^{-1} e - z_k - \Sigma_k d_k^z \tag{1.18}$$

The search direction then calculates the variable values for the next iteration:

$$x_{k+1} := x_k + \alpha_k d_k^x \tag{1.19a}$$

$$\lambda_{k+1} := \lambda_k + \alpha_k d_k^\lambda \tag{1.19b}$$

$$z_{k+1} := z_k + \alpha_k^z d_k^z \tag{1.19c}$$

$\alpha_k, \alpha_k^z \in (0,1)$ scale the step size to a feasible direction. For this, we introduce a "fraction-to-the-boundary" parameter $\tau_{min} \in (0,1)$ which is a function of the barrier parameter μ.

$$\tau_j = \max\{\tau_{min}, 1-\mu_j\} \tag{1.20}$$

The maximum α_k is calculated:

$$\alpha_k^{max} := \max\left\{\alpha \in (0,1] : x_k + \alpha d_k^x \geq (1-\tau_j)x_k\right\} \tag{1.21a}$$

$$\alpha_k^z := \max\left\{\alpha \in (0,1] : z_k + \alpha d_k^z \geq (1-\tau_j)z_k\right\} \tag{1.21b}$$

α_k^z is used in the step, but α_k can be further decreased within the line search as $\alpha_{k,l} = 2^{-l}\alpha_k^{max}$ with $l = 0,1,2,...$ where l counts the iterations of the line search, described in Section 1.4.3. First, we mention that z is corrected at the end of each iteration:

$$z_{k+1}^i \leftarrow \max\left\{\min\left\{z_{k+1}^i, \frac{\kappa_\Sigma \mu_j}{x_{k+1}^i}\right\}, \frac{\mu_j}{\kappa_\Sigma x_{k+1}^i}\right\}, \quad i = 1,...,n \tag{1.22}$$

with $\kappa_\Sigma \geq 1$ to ensure that Σ_k doesn't deviate arbitrarily much from $\mu_j X_k^{-2}$. Thus, each component σ_{k+1} of Σ_{k+1} stays within

$$\sigma_{k+1}^i \in \left[\frac{\mu_j}{\kappa_\Sigma (x_k^i)^2}, \frac{\mu_j \kappa_\Sigma}{(x_k^i)^2}\right] \tag{1.23}$$

1.4.2 Basic Active Set Algorithm

The active set method offers a different approach to dealing with inequality constraints. The active set assumes a "set" of constraints are "active" and then solves the problem with just the active constraints. If the inactive constraints are not violated and the active Lagrange multipliers remain negative, then the step is accepted and the algorithm continues to the next iteration. Otherwise the active set is updated and the search direction is recalculated. Many researchers are actively advancing this algorithm as well [93, 101–103].

SQP

The most common implementations of active-set solvers use the SQP algorithm. SQP active set methods apply the active set in the QP subproblem, with "minor" iterations until an active set is found that meets the QP optimality conditions. The QP solution is passed back to the "major" iteration for a nonlinear line search and refinement of the active set.

The formulation for an active set problem is shown in Equation 1.24. Like the interior point method, the inequality constraints can be converted to equality constraints with slack variables, as show in Equation 1.7. Conditions for active inequality constraints are then applied through the respective slack variables. Again, the algorithm is presented with one-sided boundaries for simplicity.

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & c(x) = 0 \\ & 0 \leq u \perp (s) \geq 0 \end{aligned} \tag{1.24}$$

$y \perp z$ denotes the complementarity conditions that $y_i = 0$ (inclusive) or $z_i = 0$ for all elements i of these vectors.

The Karush-Kuhn-Tucker (KKT) conditions for Problem 1.24 are below,

$$\begin{aligned} \nabla f(x) + \nabla c(x)\lambda - Eu_A &= 0 \\ c(x) &= 0 \\ E^T s &= 0 \end{aligned} \tag{1.25}$$

Here, $u \in \mathbb{R}^n$ is a vector of the variable bound multipliers. E determines the active variable bounds; $E_i = 1$ if constraint i is active or $E_i = 0$ otherwise.

Newton Step/QP Solver

Iteration proceeds with Newton steps toward a solution of the KKT conditions (Eq. 1.25). In matrix form, this is presented as:

$$\begin{bmatrix} H_k & A_k & -E \\ A_k^T & 0 & 0 \\ -E^T & 0 & 0 \end{bmatrix} \begin{pmatrix} d_k^x \\ d_k^\lambda \\ d_k^{u_A} \end{pmatrix} = - \begin{pmatrix} \nabla f(x_k) + A_k \lambda_k \\ c_A(x_k) \\ E^T(-s_k) \end{pmatrix} \tag{1.26}$$

Where c_A is the set of active constraints, $A_k := \nabla c_A(x_k)$ and H_k is the Hessian $\nabla_{xx}^2 \mathscr{L}(x_k, \lambda_k)$. By removing the inactive constraints, the active-set Newton step becomes computationally lighter than that of the interior point method.

However, the inner QP solver of the active set method has its own optimality conditions, shown in Equation 1.27 where $\hat{x}_k$, $\hat{s}_k$, and $\hat{\lambda}_k$ are the solution to the QP. The variable s represents the slack variables of the constraints.

$$c(x_k) + A_k(\hat{x}_k - x_k) = \hat{s}_k \tag{1.27a}$$

$$\nabla f(x_k) + H_k(\hat{x}_k - x_k) = A_k^T \hat{\lambda}_k \tag{1.27b}$$

$$\hat{\lambda}_k \geq 0, \quad \hat{s}_k \geq 0, \quad \hat{\lambda}_k \hat{s}_k = 0 \tag{1.27c}$$

The Newton step satisfies Equations 1.27a-1.27b, but may violate Equation 1.27c, in which case the active set must be readjusted and the Newton step must be recalculated with the new active set. The possibility of multiple QP iterations (called "minor iterations") is a weakness of the active set method which sometimes counters the benefit of the smaller matrix factorization.

Determining the Active Set

Inside the QP, the active set is determined for the approximated problem to avoid extra function calls. In the outer loop, function calls are necessary for the line search. These function calls are then used to refine the active set to the unapproximated problem. Most methods to determine the active set for commercial active set solvers are proprietary. This makes it inconvenient to build off of others' work. One method introduces Ψ in Equation 1.28 to determine the active set from the violation and multipliers of the slack variables to avoid unnecessary switching of the active set.

$$\Psi(x,u,\lambda) = \left\| \begin{bmatrix} \nabla_x \mathscr{L}(x,u,\lambda) \\ c(x) \\ min(\lambda,-s) \end{bmatrix} \right\|_\infty \tag{1.28}$$

If the slack variable of constraint i, $s_i < -\Psi$, then constraint i is made active. If the bound multiplier u_i for slack variable s_i is sufficiently negative ($u_i < -\Psi$), constraint i is made inactive. Ψ decreases with termination conditions so it will be 0 at the solution. While the iteration is far from the optimal point, it is less concerned with minor deviations.

1.4.3 Globalization and Line-Search Methods

Constrained optimization requires achieving two, sometimes competing, objectives: minimizing the objective function and satisfying constraints. Balancing these two requirements is difficult and leads to the introduction of globalization techniques. The most common techniques are the merit function and the filter method.

Filter Method

For a given step to pass, it must meet certain criteria. First, the next step is checked against a filter $\mathscr{F} \subseteq \{(\theta,\phi) \in \mathbb{R}^2 : \theta \geq 0\}$. If the point is in the filter ($(\theta(x_k(\alpha_{k,l})), \phi_{\mu_j}(x_k(\alpha_{k,l}))) \in \mathscr{F}_k$), it is rejected. This filter prevents cycling between previously-explored regions. The filter initializes at the beginning and after each update to the barrier parameter μ with:

$$\mathscr{F}_0 := \left\{ (\theta, \phi) \in \mathbb{R}^2 : \theta \geq \theta^{max} \right\} \tag{1.29}$$

If the step is not caught in the filter, the sufficient decrease condition is checked (Eq. 1.30). If Eq. 1.30 holds, and the Armijo condition (Eq. 1.32) holds, the point passes. If 1.30 does not hold, but the switching condition (Eq. 1.31) holds, the point also passes. Otherwise, the point is not acceptable.

The Sufficient Decrease Condition indicates an improvement, with sufficient progress, toward either a decrease in the objective function ϕ_{μ_j} or the constraint residuals $\theta(x) := \left\| c(x) \right\|$. $\gamma_\theta, \gamma_\phi \in (0,1)$.

$$\theta(x_k(\alpha_{k,l})) \leq (1 - \gamma_\theta)\theta(x_k) \tag{1.30a}$$

$$\text{or} \quad \phi_{\mu_j}(x_k(\alpha_{k,l})) \leq \phi_{\mu_j}(x_k) - \gamma_\phi \theta(x_k) \tag{1.30b}$$

The Switching Condition only accepts progress in the barrier function ϕ_{μ_j} if the constraint residuals are sufficiently low ($\theta^{min} \in (0, \infty)$).

$$\theta(x_k) \leq \theta^{min} \tag{1.31a}$$

$$\text{and} \quad \nabla \phi_{\mu_j}(x_k)^T d_k^x < 0 \tag{1.31b}$$

$$\text{and} \quad \alpha_{k,l} \left[-\nabla \phi_{\mu_j}(x_k)^T d_k^x \right]^{s_\phi} > \delta \left[\theta(x_k) \right]^{s_\theta} \tag{1.31c}$$

Since the switching conditions can accept progress towards a feasible, non-optimal point, the Armijo Condition verifies a decrease in the objective function to avoid the non-optimal points.

$$\phi_{\mu_j}(x_k(\alpha_{k,l})) \leq \phi_{mu_j}(x_k) + \eta_\phi \alpha_{k,l} \nabla \phi_{\mu_j}(x_k)^T d_k^x \tag{1.32}$$

If, after each iteration, either Equation 1.31 or Equation 1.32 does not exclusively hold, the filter updates to mark the region of x_k as taboo:

$$\mathscr{F}_{k+1} := \mathscr{F}_k \cup \left\{ (\theta, \phi) \in \mathbb{R}^2 : \theta \geq (1 - \gamma_\theta)\theta(x_k) \quad \text{and} \quad \phi \geq \phi_{\mu_j}(x_k) - \gamma_\phi \theta(x_k) \right\} \tag{1.33}$$

Merit Function

The merit function technique defines a merit function $m(x)$ as shown in Equation 1.34.

$$m(x,\sigma) = f(x) + \sigma_k \| c(x) \| \tag{1.34}$$

In this approach, steps are only acceptable when they decrease the merit function. This is simple to implement and effectively restructures the problem to unconstrained optimization with objective $m(x,\sigma)$; violation of the constraints is treated as a penalty of the objective.

The difficulty of this approach is determining the correct value of σ. If the magnitude of the constraint violation is insufficiently large compared to the objective function, the constraints are not respected. The function becomes more accurate as $\sigma \to \infty$. However, larger multiplier values can result in numerical difficulties as the Newton's step matrix becomes ill-conditioned.

Compared to filter methods, merit functions encourage a safe, restricted path to the solution. Ofttimes, the freedom presented by the filter method allows the solver a simpler path to the solution. Overall, filter methods have proven a worthy alternative to penalty and merit function [104].

Line Search

In QP problems, the Newton step always provides a good solution since the quadratic problem is represented exactly. In NLP problems, the QP approximation of the Newton step is not always accurate and the resulting step can at times yield a less optimal point. However, it is known that a better point exists somewhere along the search direction. Therefore, NLP solvers utilize a line-search technique to search along the vector of the step to identify a useful step size.

Sometimes, a full optimization problem is conducted along the vector to identify the minimum within the subspace. This approach is usually deemed excessive. Instead, a rough optimization seeks only to identify a point that is "good enough", as defined by the globalization technique implemented. A common approach is to successively halve the step length l times until the criteria as met, as shown in Equation 1.35 where l is the number of backtracking steps taken.

$$d^x_{k,l} = 2^{-l} d^x_k \tag{1.35}$$

1.4.4 Recent Advancements in Nonlinear Programming

Research in active set solvers has recently focused on solving larger systems and improving convergence. An active-set solver has been shown in a recent paper [105] to solve a smooth problem of up to 200,000,000 nonlinear constraints with a moderate number of variables. An active set algorithm not requiring derivative information for bound-constrained optimization has been developed using self-correcting geometry [106]. In addition, other research has used an active-set solver to improve the convergence of an interior point solver and identify the correct active set of the problem [107].

Interior Point solver research gained attention from the development of the state-of-the-art solver IPOPT in a 2004 paper by Waechter [108]. Since the advent of IPOPT, interior point solvers have been improved through regularization techniques [109, 110]. Wan [111] presents a novel method for structured regularization that detects dependent constraints during the LBL^T decomposition and perturbs only the necessary constraints. Prior to IPOPT [112], improvement in interior point methods were developed using Mehotra's corrector-predictor algorithm [113].

Of the different components of solvers in literature, globalization strategies appear most frequently. Mayne and Polak implement a search arc rather than a search direction [114]. Fletcher and Leyffer developed the penalty function free filter concept [89] and implemented efficiently with a line-search technique [108, 115]. A trust region with memory of previous iterates has been implemented [116] to improve convergence.

Avoiding the Maratos effect (the inability of a filter or merit function to accept full Newton Steps close to a strict local minimum) has received marked attention in the literature. A trust region framework development and observation of the Maratos effect for trust regions are put forth by Byrds [117]. Non-monotone line searches to avoid the Maratos Effect have also been studied. Chamberlain *et al.* implement a non-monotone watchdog technique to avoid the Maratos effect [118]. Other non-monotone line search strategies include the generalization of Armijo's condition combined with a watchdog technique [119] and the combination of Mayne and Polak's search arc with the watchdog technique [120].

More recently, research in globalization strategies has focused on integrating multiple strategies. A non-monotone trust region with a line search technique [121] combines two globalization strategies to make a more efficient and robust algorithm. A similar trust region and line search integrated approach decreases computational expense [122]. Another recent article implements a spectral gradient method to determine step size in a line search [123].

Most optimization algorithms approximate nonlinear constraints linearly to solve the Newton Step. The more nonlinear the constraints are, the less accurate the Newton Step is towards the actual solution. To solve highly nonlinear equations, higher order methods have received attention in the literature [124]. Higher order and correction steps used to solve non-linear problems have shown improved convergence in convex and non-convex scenarios. Kchouk and Dussault propose a generalization of higher order and iterative methods [125, 126]. Using a comparison of convergence to cost, the above authors determine that higher order methods were more efficient compared to Newton's Method for large systems of nonlinear equations.

Higher order methods have been shown to converge faster than Newton's method for unconstrained optimization close to the solution [127]. Accelerated gradient methods calculate a step and then go slightly farther to achieve faster convergence. Ghadimi *et al.* extend the accelerated gradient from smooth convex applications to non-convex nonlinear optimization [128]. The extension achieves optimal convergence for convex problems and improves the best rate of convergence for non-convex problems. Another iterative method called the predictor-corrector method has been applied to Halley's method [129] to improve convergence rate. Other research has developed high order convergent iterative schemes not requiring second derivative information by using numerical methods [130, 131]. One drawback to higher order methods is low computational efficiency in calculating higher order derivatives. The authors of [132] determine that the ratio of the number of arithmetic operations for Halley's step and Newton's Step is independent of the number of unknowns for Hessian matrices with skyline or envelope structure.

Recent literature for solver algorithms focuses on improving individual components of solver performance. Many authors have exploited problem structure to improve algorithmic efficiency. For problems with block diagonal Hessians typical in optimal control, updating blocks of the Hessian using SR1 updates when possible and implementing an inertia control algorithm in the factorization increase computational efficiency [133]. A visual comparison method for algorithms and a Broyden–Fletcher–Goldfarb–Shanno (BFGS) SQP algorithm has been developed [134]. The BFGS SQP algorithm outperforms SNOPT. Another computationally expensive part of solvers is the feasibility restoration phase. If a globalization strategies fail to find any acceptable step length, a trial point outside of the vicinity is chosen through a restoration phase. Li and Yang [135] develop a line-search technique that functions without requiring a filter, penalty function, or restoration phase. Their algorithm implements a novel pivoting algorithm for the active set. Other authors have combined solver methods using an augmented Lagrangian for equality constraints and a barrier function for inequality constraints [136]. In addition, approximating constraints using quadratic rather than linear approximations has been designed and implemented [137].

Solver algorithms can operate in the feasible space only or also in the infeasible space. In some problems, a solution in the feasible space does not exist. Finding an infeasible solution quickly can reduce a solver's iterations in finding a non-existing feasible solution. An algorithm has been created [138] to find infeasible solutions at the same convergence rate as feasible solutions using a SQP algorithm. However, solvers operating only in the feasible region give feasible solutions at every iterate. Iteration details can provide good information about a solution if the solver stays in the feasible space. To provide this information, Zhu *et al.* have developed an efficient SQP algorithm iterating only in the feasible region [116].

Algorithm analysis has also appeared in literature. Information about worst-case and best-case efficiency has been examined. For example, algorithmic complexity of nonconvex diminishing step size algorithms are analyzed in [139].

1.5 Scheduling

Scheduling answers the questions of when to produce each product. Scheduling optimization algorithms are centered around market price (of both products and raw materials), market demand, deadlines and contractual agreements. The field of scheduling optimization is particularly difficult for a number of reasons. Primarily, the time-scales under consideration naturally include a lot of uncertainty. If any part of the schedule breaks down, subsequent deadlines are likely to be missed. There are many discrete events such as on-spec production or waste, or the availability of materials (such as storage tanks). Mixed-integer problems are magnitudes of order more difficult than continuous problems, especially mixed-integer non-linear programming (MINLP). To obtain reasonable schedules in a useful amount of time, schedulers have historically reduced their problems to mixed-integer linear problems. Further, practitioners deliberately ignore some scheduling considerations in the optimization, opting instead for human manipulation of optimization output for final schedules. A recent review paper on industrial implementations of scheduling states that the optimized schedule should be output in a user-friendly format and to assume that the scheduler will manually adjust the results [140]. The human-in-the-loop also factors in emotional consideration that are difficult to quantify such as gut feelings, reliability of certain units, relationships with customers, etc.

As part of the model simplification, only steady-state process models are considered. In batch processes, this reduces "stages" of the process to a uniform block. In continuous process, this considers only steady-state production. This dissertation explores continuous chemical manufacturing. By ignoring process dynamics, the scheduler loses two benefits: knowledge of transitions between products and only one possible steady-state production condition per product. Transitions play an important role in scheduling because they are effectively down-time generating only waste/off-spec products. By not considering this down time, the scheduler is overly-optimistic and always wrong. By only considering one steady-state, the scheduler loses the possibility of increased production rates under favorable conditions.

There are two main paradigms to representing time in scheduling optimization: continuous-time (or "slot-based" schedules) or discrete-time schedules. In continuous-time schedules, production of a given product is assigned a "slot". The beginning and ending time of each slot is determined by the solver and is linked to the slot times of adjacent slots. Any value of time is possible, thus time is "continuous". In advanced slot-based scheduling, the number of slots is also optimized in an outer loop. This is a favorable approach because the number of degrees of freedom is equal to the number of production slots (effectively the length of time for each slot) and slot lengths have infinite options. The alternative approach discretizes time to a fixed number of points. The solver then selects which product is produced during each time interval. In this approach there are degrees of freedom for each discrete time point and production length is fixed to discrete intervals matching time discretization. With this perspective, continuous-time scheduling appears to be superior. However, [141] found that the arbitrary formulation of slot-based schedules proved more difficult for solvers as the problem increased in complexity, whereas the discrete-time approach difficulty scaled more reasonably.

1.6 Control

The primary objective of controllers is to maintain process conditions at a pre-determined set of conditions, or “set points”, as shown in Equation 1.3. Steady-state set point are often determined by RTO to produce the product selected by the scheduler. The challenges of advanced control are dominated by the time scales of the real system. Controllers must be kept on-line and must therefore be robust and reliably solve fast enough for control moves to be relevant. Control faces challenges in process disturbances, inexact process models (plant-model mismatch) and messy measurements.

The core of MPC is the model of the system, defined by the engineer. A more accurate model of the system yields better results. With an inaccurate model, the optimizer will find the very best control move for an irrelevant system. However, high-fidelity models, such as those used in high-end simulators, are too slow to be useful in real-time. Practitioners must judge the appropriate level of detail to include in MPC models, knowing that some model mismatch will be compensated through feedback loops. If a model is not sufficiently robust, it could return an infeasible result and the control is then worthless on a live system. Sometimes safeguards can account for the occasional error, but frequent failures are unacceptable to process operators.

MPC typically has many degrees of freedom since the problem is dynamic and discretized through time. Each manipulated variable adds a degree of freedom at every discrete time point.

1.7 Combining Scheduling and Control

By applying modern tools to their fullest capacity, we find added benefits by solving increasingly larger problems. These advanced techniques are more efficient, result in less waste and are more profitable. With an understanding of modern control and scheduling approaches, and the underlying tools of modeling languages and solvers, this dissertation explores approaches to combine scheduling and control together. Through unification, scheduling gains knowledge of process dynamics and the controller gains knowledge of multiple products and economics. Together, we find the unified problem addresses defects and performs better than is possible through the current scheme.

This field is difficult because of the massive scale of the problem. Scheduling inherently requires long timescales, while control requires detailed process dynamics. Mixing the two creates a large problem of detailed process dynamics over long times. However, successful implementations are demonstrated to be of significant benefit.

1.8 Outline

The remainder of this dissertation explores the combining of scheduling and control. Chapter 2 motivates the field by exploring the benefits of combining scheduling and control. Various degrees of integration are postulated and implemented. Simulated disturbances highlight the weaknesses of the current, segregated approach. The economic benefits of the progressively integrated scheduling and control are quantified in simulated case studies.

Chapter 3 explores integration of scheduling and control in discrete-time. This chapter focuses on fast, robust solutions by limiting the dynamics to a linear model and keeping the problem within quadratic equations. Changing production rates are included through time-scaling. This approach is applicable to mostly-linear problems, or those currently using linear MPC.

Chapter 4 expands the discrete-time approach of Chapter of 3 to a full nonlinear problem. By allowing nonlinear constraints, the full benefits of the discrete-time approach are exploited through time-based constraints and parameters for the most profitable solutions.

1.9 Novel Contributions

The following contributions to the optimization community are presented in this dissertation:

- Present a computationally light decomposed integration method employing continuous-time scheduling and NMPC
- Propose a benchmark problem with four stages of integration and three relevant disturbances
- Implement the benchmark study and quantitatively demonstrate the benefit of tighter integration of scheduling and control

- Present a paradigm that fully unifies scheduling and control to one optimization problem with a discrete-time formulation that both generates a schedule and maintains closed-loop control
- Apply the discrete-time approach with dynamic linear models
- Present time-scaling algorithm to enable the algorithm with varying production rates
- Achieve fast solutions with Quadratic Programming with Quadratic Constraints (QPQC) framework
- Present various continuous-relaxation approaches of binary variables in discrete-time with sample performance metrics
- Implement discrete-time approach in full-nonlinear form to enable dynamic constraints and parameters
- Present initialization techniques to promote convergence, including novel application of feedback linearization

---

Portions of this chapter are derived from the following published works:

Beal, L. D. R., Hill, D. C., Martin, R. A., and Hedengren, J. D., 2018. "Gekko optimization suite." *Processes,* **6**(8)

Petersen, D., Beal, L. D. R., Prestwich, D., Warnick, S., and Hedengren, J. D., 2017. "Combined noncyclic scheduling and advanced control for continuous chemical processes." *Processes,* **5**(4)

CHAPTER 2. ECONOMIC BENEFIT FROM PROGRESSIVE INTEGRATION OF SCHEDULING AND CONTROL FOR CONTINUOUS CHEMICAL PROCESSES

This chapter is composed of the following articles published in the Processes Journal Special Issue “Combined Scheduling and Control”:

Beal, L. D. R., Petersen, D., Pila, G., Davis, B., Warnick, S., and Hedengren, J. D., 2017. “Economic benefit from progressive integration of scheduling and control for continuous chemical processes.” *Processes,* **5**(4)

Petersen, D., Beal, L. D. R., Prestwich, D., Warnick, S., and Hedengren, J. D., 2017. “Combined noncyclic scheduling and advanced control for continuous chemical processes.” *Processes,* **5**(4)

2.1 Introduction

Production scheduling and advanced process control are related tasks for optimizing chemical process operation. Traditionally, implementation of process control and scheduling are separated; however, research suggests that opportunity is lost from separate implementation [14, 143, 144]. Many researchers suggest that economic benefit may arise from integrating production scheduling and process control [19, 27, 30, 31, 145–147]. Though integration may provide economic benefit, scheduling and control integration presents several challenges which are outlined in multiple reviews on integrated scheduling and control (ISC) [14–17, 140]. Some of the major challenges to integration mentioned in review articles include time-scale bridging, computational burden, and human factors such as organizational and behavioral challenges.

2.1.1 Economic Benefit from Integrated Scheduling and Control

Many complex, interrelated elements factor into the potential benefit from the integration of scheduling and control, including the following [14, 15]:

1. Rapid fluctuations in dynamic product demand
2. Rapid fluctuations in dynamic energy rates
3. Dynamic production costs
4. Benefits of increased energy efficiency
5. Necessity of control-level dynamics information for optimal production schedule calculation

In the current economic environment, demand and selling prices for the products and inputs of chemical processes can change significantly over the course of not only months and years, but on the scales of weeks, days, and hours [14, 15, 140]. Energy rates often fluctuate hourly, with peak pricing during peak demand hours and rate cuts during off peak hours (sometimes even negative rate cuts occur during periods of excess energy production) [15]. An optimal schedule is intrinsically dependent upon market conditions such as input material price, product demand and pricing, and energy rates [140]. Therefore, when market conditions change, the optimal production sequence or schedule may also change. Since the time scale at which market factors fluctuates has decreased, the time scale at which scheduling decisions must be recalculated should also decrease [14, 140].

Frequent recalculation of scheduling on a time scale closer to that of advanced process control (seconds to minutes) leads to a greater need to integrate process dynamics into the scheduling problem [14]. According to a previous review [15], process dynamics are important for optimal production scheduling because (i) transition times between any given products are determined by process dynamics and process control (ii) process dynamics may show that a calculated production sequence or schedule is operationally infeasible (iii) process disturbances may cause a change in the optimal production sequence or schedule.

2.1.2 Previous Work

Significant research has been conducted on the integration of production scheduling and advanced process control [14, 15]. This section summarizes evidence for economic benefit from integration, upon which this work builds. Previous research showing the benefits of combined scheduling and control is explored and previous research done to show the economic benefits of combined over segregated scheduling and control is examined. The reviewed articles are summa-

Table 2.1: Economic Benefit of Integrated Scheduling and Control (ISC) Over Segregated Scheduling and Control (SSC)

Author	Compare ISC/SSC	Batch(B) or Continuous(C)	Example Application(s)
Baldea *et al.* [21]		C	CSTR
Baldea *et al.* [22]		C	MMA
Baldea [148]		C	chemical processes as energy storage
Beal *et al.* [39]		C	CSTR
Beal *et al.* [37]		C	CSTR
Cai *et al.* [149]		B	Semiconductor production
Capon-Garcia *et al.* [19]		B	2 different batch plants
Chatzidoukas *et al.* [150]	X	C	gas-phase polyolefin FBR.
Chatzidoukas *et al.* [151]	X	C	catalytic olefin copolymerization FBR
Chu & You [24]		C	MMA
Chu & You [23]		C	CSTR
Chu & You [152]		B	parallel polymerization & purification (RTN)
Chu & You [153]	X	B	5-unit batch process
Chu & You [154]	X	B	sequential batch process
Chu & You [25]		B	reaction, filtration, reaction tasks
Chu & You [155]		B	8-unit batch process
Chu & You [156]	X	B	8-unit batch process
Dias *et al.* [157]		C	MMA
Du *et al.* [35]		C	CSTR & MMA
Flores-Tlacuahuac & Grossmann [29]		C	CSTR
Flores-Tlacuahuac [30]		C	Parallel CSTRs
Gutiérrez-Limón *et al.* [158]		C	CSTR
Gutiérrez-Limón *et al.* [159]		C	CSTR & MMA
Gutiérrez-Limón *et al.* [160]		C	CSTR
Koller & Ricardez-Sandoval [161]		C	CSTR
Nie & Bieglier [147]	X	B	flowshop plant (reactor, filter, distillation)
Nie *et al.* [162]		BC	parallel polymerization & purification
Nystrom *et al.* [8]		C	industrial polymerization process
Nystrom *et al.* [145]		C	industrial polymerization process
Patil *et al.* [163]		C	CSTR & HIPS
Pattison *et al.* [26]	X	C	ASU model
Pattison *et al.* [27]		C	ASU model
Prata (2008) *et al.* [164]		C	medium industry-scale model
Terrazas-Moreno *et al.* [165]		C	MMA (with one CSTR) & HIPS
Terrazas-Moreno *et al.* [28]		C	HIPS & MMA
Terrazas-Moreno *et al.* [31]		C	HIPS & MMA
You & Grossmann [166]		C	medium & large polystyrene supply chains
Zhuge & Ierapetritou [20]		C	CSTR & PFR.
Zhuge & Ierapetritou [32]		B	simple and complex batch processes
Zhuge & Ierapetritou [33]		C	SISO & MIMO CSTRs
Zhuge & Ierapetritou [34]	X	C	CSTR & MMA

rized in Table 2.1. This work focuses on research demonstrating benefit over a baseline comparison of segregated scheduling and control (SSC).

Integrating Process Dynamics into Scheduling

Mahadevan *et al.* suggest that process dynamics should be considered in scheduling problems. To avoid the computational requirements of mixed-integer nonlinear programming (MINLP), they include process dynamics as costs in the scheduling problem [167]. Flores-Tlacuahuac and Grossman implement process dynamics into scheduling directly in a mixed-integer dynamic optimization (MIDO) problem with a continuous stirred tank reactor (CSTR). Chatzidoukas *et al.* demonstrated the economic benefit of implementing scheduling in a MIDO problem for polymerization, solving product grade transitions along with the scheduling problem [150]. Economic benefit has also been shown for simultaneous selection of linear controllers for grade transitions and scheduling, ensuring that the process dynamics from the controller selection are accounted for in the scheduling problem [151]. Terrazas-Moreno *et al.* also demonstrate the benefits of process dynamics in cyclic scheduling for continuous chemical processes [28]. Capon-Garcia *et al.* prove benefit of implementing process dynamics in batch scheduling via a MIDO problem [19]. MIDO batch scheduling optimization with dynamic process models is shown to be more profitable than a fixed-recipe approach. Chu and You also demonstrate enhanced performance from batch scheduling with simultaneous solution of dynamic process models over a traditional batch scheduling approach [153, 154, 156]. Economic benefit from integrating process dynamics into batch and semi-batch scheduling has also been demonstrated via mixed-logic dynamic optimization in state equipment networks and solution with Benders decomposition in resource task networks [147, 162]. Potential for economic benefit from integrating process dynamics into design, scheduling, and control problems has also been demonstrated [163, 165, 168]. Computational reduction of incorporating process dynamics into scheduling has been investigated successfully, maintaining benefit from the incorporation of process dynamics into scheduling while reducing dynamic model order [21, 22, 26, 27, 35].

Reactive Integrated Scheduling and Control

Research indicates that additional benefit arises from ISC responsive to process disturbances, which are a form of process uncertainty. This is in congruence with recent work by Gupta and Maravelias demonstrating that increased frequency of schedule rescheduling (on-line scheduling) can improve process economics [169–171]. Many previous works considering reactive ISC are outlined in Table 2.2. For a complete review of ISC under uncertainty, the reader is directed to a recent review by Dias and Ierapetritou [157]. Zhuge and Ierapetritou demonstrate increased profit from closed-loop implementation (over open-loop implementation) of combined scheduling and control in the presence of process disturbances [20]. The schedule is optimally recalculated when a disturbances is encountered. Zhuge and Ierapetritou also present methodology to reduce the computational burden of ISC to enable closed-loop on-line operation for batch and continuous processes. They propose using multi-parametric model predictive control for on-line batch scheduling and control [32], fast model predictive control coupled with reduced order (piece-wise affine) models in scheduling and control for continuous processes [33], and decomposition into separate problems for continuous processes [34]. Chu and You demonstrate the economic benefit of closed-loop moving horizon scheduling with consideration of process dynamics in batch scheduling [25]. Chu and You also investigate the reduction of computational burden to enable on-line closed-loop ISC for batch and continuous processes. They investigate utilization of Pareto frontiers to decompose batch scheduling into an on-line mixed-integer linear programming (MILP) problem and off-line dynamic optimization (DO) problems [152]. Investigation of a solution via mixed-integer nonlinear fractional programming and Dinkelbach's algorithm coupled with decomposing into an on-line scheduling and controller selection and off-line transition time calculation [24].

Closed-loop reactive ISC responds to process uncertainty in a reactive rather than preventative manner [174]. Preventative approaches to dealing with process uncertainty in ISC have also been investigated. Chu and You investigated accounting for process uncertainty in batch processes in a two-stage stochastic programming problem solved by a generalized Benders decomposition [154]. The computational requirements of the problem prevent on-line implementation. Dias and Ierapetritou demonstrate the benefits of using robust model predictive control in ISC to optimally address process uncertainty in continuous chemical processes [157].

Table 2.2: Works Considering Reactive ISC

Authors	Product Price Disturbance	Product Demand Disturbance	Process Variable Disturbance	Other Disturbances
Baldea (2017) [148]			X	
Cai *et al.* (2012) [149]			X	
Chu & You (2012) [24]			X	
Du *et al.* (2015) [35]				
Flores-Tlacuahuac (2010) [30]		X		
Gutiérrez-Limón *et al.* (2016) [159]		X		
Kopanos & Pistikopoulos (2014) [172]			X	
Liu *et al.* (2012) [173]	X	X		
Patil *et al.* (2015) [163]			X	
Pattison *et al.* (2017) [27]		X	X	
Touretzky & Baldea (2014) [9]				Weather & energy price
You & Grossmann (2008) [166]		X		
Zhuge & Ierapetritou (2012) [20]			X	
Zhuge & Ierapetritou (2015) [33]			X	

Responsiveness to Market Fluctuations

As mentioned in Section 2.1.1, a major consideration affecting the profitability of ISC is rapidly fluctuating market conditions. If the market changes, the schedule should be reoptimized to new market demands and price forecasts. This is again congruent with recent work demonstrating benefit from frequent re-scheduling [169–171]. Literature on ISC reactive to market fluctuations is relatively limited in scope. Gutierrez-Limon *et al.* demonstrated integrated planning, scheduling, and control responsive to fluctuations in market demand on a CSTR benchmark application [159, 160]. Pattison *et al.* investigated ISC with an air separation unit (ASU) in fast-changing electricity markets, responding optimally to price fluctuations [26]. Pattison *et al.* also demonstrated theoretical developments with moving horizon closed-loop scheduling in volatile market conditions [27]. Periodic rescheduling to account for fluctuating market conditions was implemented successfully on an ASU application.

2.1.3 Purpose of this Work

This work aims to provide evidence for the progressive economic benefits of combining scheduling and control and operating combined scheduling and control in a closed-loop responsive to disturbances over segregated scheduling and control and open-loop formulations for continuous chemical processes. This work demonstrates the benefits of integration through presenting four progressive stages of integration and responsiveness to disturbances. This work comprehensively demonstrates the progression of economic benefit from (1) integrating process dynamics and control level information into production scheduling and (2) closed-loop integrated scheduling and control responsive to market fluctuations. Such a comprehensive examination of economic benefit has not been performed to the authors' knowledge. This work also utilizes a novel, computationally light decomposed integration method employing continuous-time scheduling and nonlinear model predictive control (NMPC) as the fourth phase of integration. Although the phases of integration presented in this work are not comprehensively representative of integration methods presented in the literature, the concepts of integration progressively applied in the four phases are applicable across the majority of formulations in the literature.

2.2 Phases of Progressive Integration

This section introduces the four phases of progressive integration of scheduling and control investigated in this work. Each phase is outlined in the appropriate section.

2.2.1 Phase 1: Fully Segregated Scheduling and Control

A schedule is created infrequently (every 24 hours in this work) and a controller seeks to implement the schedule throughout the 24 hours with no other considerations. In this format, the schedule is open-loop whereas the control is closed-loop. The controller acts to reject disturbances and process noise to direct the process to follow the predetermined schedule (See Figure 2.1).

This work considers an NMPC controller and a continuous-time, slot-based schedule (Section 2.2.5). For this phase, the schedule is uninformed of transition times as dictated by process dynamics and control structure. All product grade transitions are considered to produce a fixed amount of off-specification material and to require the same duration.

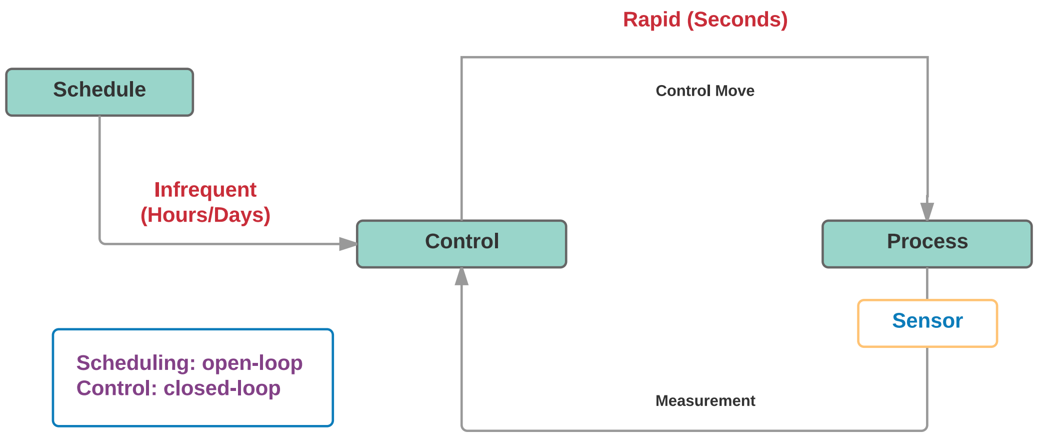


Figure 2.1: Phase 1: Open-loop scheduling determined once per day with no consideration of process dynamics. Closed-loop control implemented to follow the schedule.

2.2.2 Phase 2: Reactive Closed-loop Segregated Scheduling and Control

Phase 2 is a closed-loop implementation of completely segregated scheduling and control. The formulation for phase 2 is identical to that of phase 1 with the exception that the schedule is recalculated in the event of a process disturbance or market update.

2.2.3 Phase 3: Open-loop Integrated Scheduling and Control

For phase 3 the schedule is calculated infrequently, similar to phase 1 (every 24 hours in this work). However, information about the control structure and process dynamics in the form of transition times are fed to the scheduling algorithm to enable a more intelligent decision. Scheduling remains open-loop while the controller remains closed-loop to respond to noise and process disturbances while implementing the schedule.

This work considers a continuous-time schedule with process dynamics incorporated via transition times estimated by NMPC. Transitions between products are simulated with a dynamic process model and nonlinear model predictive controller implementation. The time required to transition between products is minimized by the controller, and the simulated time required to

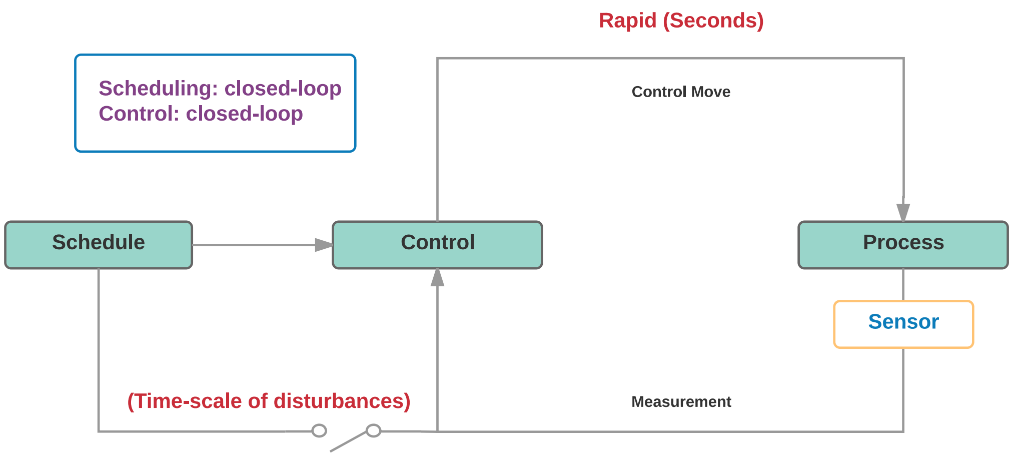


Figure 2.2: Phase 2: Dual-loop segregated scheduling and control. Scheduling is recalculated reactively in the presence of process disturbances above a threshold or updated market conditions. Closed-loop control implements the schedule in the absence of disturbances.

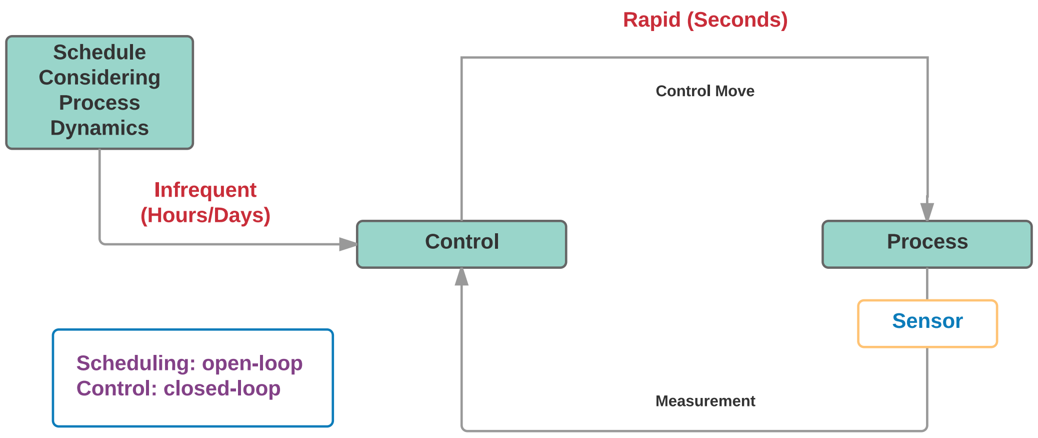


Figure 2.3: Phase 3: Open-loop scheduling determined once per day with consideration of process dynamics and control structure in the form of grade transition information. Closed-loop control implemented to follow the schedule.

transition is fed to the scheduler as an input to the continuous-time scheduling formulation (Section 2.2.5).

2.2.4 Phase 4: Closed-loop Integrated Scheduling and Control Responsive to Market Fluctuations

Phase 4 represents closed-loop implementation of ISC responsive to both market fluctuations and process disturbances. This work utilizes the formulation for computationally light on-line scheduling and control for closed-loop implementation further explored in [142]. As in phase 3, a continuous-time schedule is implemented with NMPC-estimated transition times as inputs to the scheduling optimization; however, the ISC algorithm is implemented not only once at the beginning of the horizon as in phase 3, but triggered by updated market conditions or process disturbances above a threshold. This enables the ISC algorithm to respond to fluctuations in market conditions as well as respond to measured process disturbances in a timely manner to ensure that production scheduling and control are updated to reflect optimal operation with current market conditions and process state.

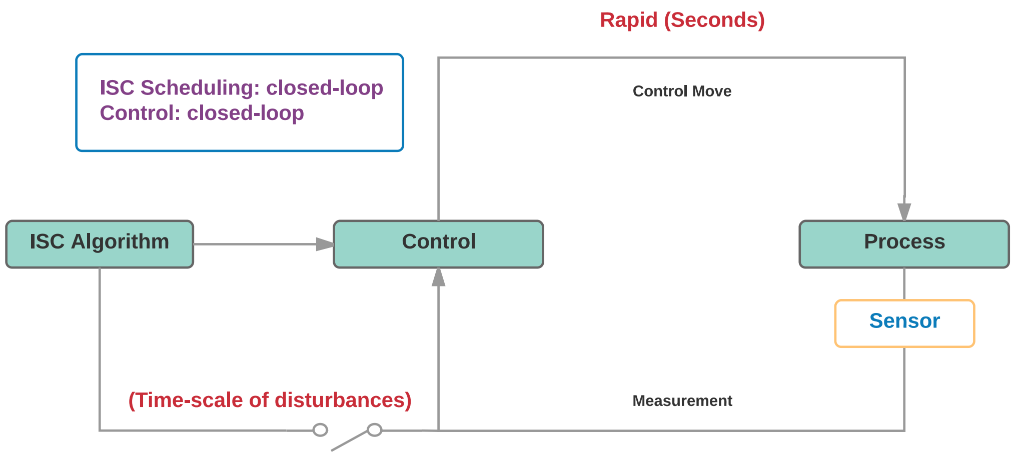


Figure 2.4: Phase 4: Closed-loop combined scheduling and control responsive to both process disturbances and updated market information.

This formulation builds on previous work which demonstrates the separability of the integrated scheduling and control problem into subproblems without the need for iterations [34] and builds on previous work which demonstrates the separation into MILP and dynamic optimization problems [8, 23, 145]. This formulation also builds on previous work which demonstrate benefit from shifting separable computational burden into off-line portions of the integrated problem [24,32,33,155].

The formulation for phase 4 builds on the work of Zhuge *et al.* [33], which justifies decomposing slot-based ISC into two subproblems: (1) NLP solution of transition times and transition control profiles and (2) MILP solution of the slot-based, continuous-time schedule. The approach expands the work of Zhuge *et al.* by combining a look-up transition time table with control profiles and transition times between known product steady-state conditions, calculated off-line and stored in memory, with transitions from current conditions to each product. The transitions from current conditions or most recently received process measurements are the only transition times and transition control profiles required to be solved at each iteration of combined scheduling and control (Eqs. 2.2-2.8). This reduces the on-line problem to few non-linear programming (NLP) dynamic optimization problems and a MILP problem only, eliminating the computational requirements of MINLP. This work also introduces the use of nonlinear models in this form of decomposition. Zhuge *et al.* use piecewise affine (PWA) models whereas this work harnesses full nonlinear process dynamics to calculate optimal control and scheduling.

This work also builds on the work of Pattison *et al.*, who demonstrate closed-loop moving horizon combined scheduling and control to respond to market updates [27]. This formulation, however, does not use simplified dynamic process models for scheduling, but rather maintains nonlinear process dynamics while reducing computational burden via problem decomposition into off-line and on-line components and further decomposition of the problem into computationally light NLP and MILP problems, solvable together without the need for iterative alternation [142].

The continuous-time scheduling formulation, as introduced in Section 2.2.5, will produce sub-optimal results if the number of products exceeds the optimal number of products to produce in a prediction horizon. The number of slots is constrained to be equal to the number of products, causing the optimization to always create n production slots and n transitions even in cases in which $< n$ slots would be most economical in the considered horizon for scheduling and control. To eliminate this sub-optimality, an iterative method is introduced to leverage the computational lightness of the MILP continuous-time scheduling formulation. The number of slots in the continuous-time schedule is selected iteratively based on improvement to the objective function (profit), beginning from one slot. As previously mentioned, transition times and control profiles between steady-state products are stored in memory, requiring no computation in on-line operation. Additionally, the transitions from current measured state to each steady-state product ($\tau_{0'i}$) are calculated once before iterations are initiated. Thus, the iterative method only iterates the MILP problem, not requiring any recalculation of grade transition NLP dynamic optimization problems. This decomposition is computationally light and allows for a fixed-horizon non-cyclic scheduling and control formulation. This non-cyclic fixed-horizon approach to combined scheduling and control enables response to market fluctuations in maximum demand and product price, whereas traditional continuous-time scheduling requires a makespan (T_M) to meet a demand rather than producing an optimal amount of each product within a given fixed horizon.

2.2.5 Mathematical Formulation

This continuous-time optimization used in phases 1-3 seeks to maximize profit and minimize grade transitions (and associated waste material production) while observing scheduling constraints. The objective function is formulated as follows:

$$\begin{aligned} \max_{z_{i,s},t_i^s,t_i^f \forall i,s} \quad & J = \sum_{i=1}^{n} \Pi_i \omega_i - \sum_{i=1}^{n} c_{storage,i}\omega_i \sum_{s=1}^{m} z_{i,s}(T_M - t_s^f) - W_\tau \sum_{s=1}^{m} \tau_s \\ \text{s.t.} \quad & \text{Eq. } 2.2 - 2.8 \end{aligned} \tag{2.1}$$

where T_M is the makespan, n is the number of products, m is the number of slots ($m = n$ in these cyclic schedules), $z_{i,s}$ is the binary variable that governs the assignment of product i to a particular slot s, t_s^s is the start time of the slot s, t_s^f is the end time of slot s, Π_i is the per unit price of product i, W_τ is an optional weight on grade transition minimization, τ_s is the transition time within slot s, $c_{storage,i}$ is the per unit cost of storage for product i, ω_i represents the amount of product i manufactured,

$$\omega_i = \sum_{s=1}^{m} \int_{t_s^s+\tau_s}^{t_s^f} z_{i,s}\, q\, dt \tag{2.2}$$

and where q is the production volumetric flow rate and τ_s is the transition time between the product made in slot s - 1 and product i made in slot s.

The values of the vector τ_s are determined by the optimization variables $z_{i,s}$. τ_s represents the transition time between the product made in slot s - 1 and product made in slot s. Thus, the value of τ_s is determined by the optimal values of $z_{j,s-1}$ and $z_{k,s}$ which determine which products are assigned to slots $s-1$ and s. For $z_{j,s-1}$ and $z_{k,x}$, τ_s would be equal to the calculated grade transition time from product j to product k. The possible values for τ_s are the grade transition times calculated via NLP optimal grade transition problems (Eq. 2.9). The time points must satisfy the precedence relations

$$t_s^f > t_s^s + \tau_s \qquad \forall s > 1 \tag{2.3}$$

$$t_s^s = t_{s-1}^f \qquad \forall s \neq 1 \tag{2.4}$$

$$t_m^f = T_M \tag{2.5}$$

which require that a time slot be longer than the corresponding transition time, impose the coincidence of the end time of one time slot with the start time of the subsequent time slot, and define the relationship between the end time of the last time slot (t_n^f) and the total makespan or horizon duration (T_M).

Products are assigned to each slot using a set of binary variables, $z_{i,s} \in \left\{0,1\right\}$, along with constraints of the form

$$\sum_{s=1}^{m} z_{i,s} = 1 \qquad \forall i \tag{2.6}$$

$$\sum_{i=1}^{n} z_{i,s} = 1 \qquad \forall s \tag{2.7}$$

which ensure that one product is made in each time slot and each product is produced once.

The makespan is fixed to an arbitrary horizon for scheduling. Demand constraints restrict production from exceeding the maximum demand (δ_i) for a given product, as follows:

$$\omega_i \leq \delta_i \qquad \forall i \tag{2.8}$$

The continuous-time scheduling optimization (or MILP problem) requires transition times between steady-state products ($\tau_{i'i}$) as well as transition times from the current state to each steady-state product if initial state is not at steady-state product conditions ($\tau_{0'i}$). These grade transitions comprise the separable dynamic optimization problems or NLP portion of the overall problem decomposition.

Transition times are estimated using NMPC via the following objective function:

$$\begin{aligned} \min_{u} \quad & J = (x - x_{sp})^T W_{sp} (x - x_{sp}) + \Delta u^T W_{\Delta u} + u^T W_u \\ \text{s.t.} \quad & \text{nonlinear process model} \\ & x(t_0) = x_0 \end{aligned} \tag{2.9}$$

where W_{sp} is the weight on the set point for meeting target product steady-state, $W_{\Delta u}$ is the weight on restricting manipulated variable movement, W_u is the cost for the manipulated variables, u is the vector of manipulated variables, x_{sp} is the target product steady-state, and x_0 is the start process state from which the transition time is being estimated. The transition time is taken as the time at which and after which $|x - x_{sp}| < \delta$, where δ is a tolerance for meeting product steady-state operating conditions. This formulation harnesses knowledge of non-linear process dynamics in the system model to find an optimal trajectory and minimum time required to transition from an initial concentration to a desired concentration. This method for estimating transition times also effectively captures the actual behavior of the controller selected, as the transition times are estimated by a simulation of actual controller implementation. This work uses $W_{\Delta u} = 0$ and $W_u = 0$.

Because product steady-state operating conditions can be known *a priori*, all grade transition times between production steady-state operating conditions can be calculated off-line and stored in memory in a grade transition time table: τ_{ss}. The on-line portion of the NLP subproblem comprises only of the calculation of τ_θ, the transition duration and corresponding optimal control profile from current measured state (x_θ) and each steady-state operating condition, or in other words, the vector of possible transitions for the first slot ($s = 0$). For example, consider the case of three products as described in Table 2.3. As product steady-state conditions are known *a priori* as shown in Table 2.3, the transition times between product steady-states can be calculated through NLP problems (Eq. 2.9) and stored in a grade transition time table prior to operation (Table 2.4). However, before operation, process state measurements (x_θ) cannot be known. For example, if in the three product case described in Table 2.3 a process disturbance was measured at $x_\theta = 0.19 mol/L$ during on-line operation, for an optimal rescheduling beginning from current state x_θ to be calculated, transition durations from x_θ to each operating steady-state would need to be estimated on-line by solving n separable NLP optimal grade transition problems, where n is the number of steady-state products considered. The results of these on-line optimal transition problems would be the vector τ_θ (Table 2.5). These transitions would be the possible values for τ_s for the first slot ($s = 0$) in Eq. 2.1 for the MILP rescheduling problem.

Table 2.3: Product Specifications

Product	C_A (mol/L)
1	0.10
2	0.30
3	0.50

Table 2.4: Transition Time Table (τ_{ss}) *

Start	End Product		
Product	1	2	3
1	0.00	0.71	1.20
2	0.45	0.00	0.71
3	0.94	1.57	0.00

* Transition times in hours.

With τ_{ss} and τ_θ grade transition information, the MILP problem is equipped to optimally select the production sequence and amounts for the prediction horizon based on product demands, prices, transition durations, raw material cost, storage cost, and other economic parameters. Even when a process disturbance is encountered and measured, the schedule can be optimally recal-

Table 2.5: Initial Transitions (τ_θ)

Product	τ_θ (hr)
1	0.31
2	0.43
3	0.96

culated from the measured disturbance, x_θ, via the incorporation of transitions from x_θ to each production steady-state condition (τ_θ).

2.3 Case Study Application

As shown in prior work, there are many different strategies for integrating scheduling and control. A novel contribution of this work is a systematic comparison of four general levels of integration through a single case study. In this section the model and scenarios used to demonstrate progressive economic benefit from the integration of scheduling and control for continuous chemical processes are presented.

2.3.1 Process Model

This section presents a standard CSTR problem used to highlight the value of the formulation introduced in this work. The CSTR model is applicable in various industries from food/beverage to oil and gas and chemicals. Notable assumptions of a CSTR include:

- Constant volume
- Well mixed
- Constant density

The model shown in Eqs. 2.10 to 2.11 is an example of an exothermic, first-order reaction of $A \Rightarrow B$ where the reaction rate is defined by an Arrhenius expression and the reactor temperature is controlled by a cooling jacket.

$$\frac{dC_A}{dt} = \frac{q}{V}(C_{A0} - C_A) - k_0 e^{-E_A/RT} C_A \tag{2.10}$$

$$\frac{dT}{dt} = \frac{q}{V}(T_f - T) - \frac{1}{\rho C_p} k_0 e^{\frac{-E_A}{RT}} C_A \Delta H_r - \frac{UA}{V \rho C_p}(T - T_c) \tag{2.11}$$

In these equations, C_A is the concentration of reactant A, C_{A0} is the feed concentration, q is the inlet and outlet volumetric flowrate, V is the tank volume (q/V signifies the residence time), E_A is the reaction activation energy, R is the universal gas constant, UA is an overall heat transfer coefficient times the tank surface area, ρ is the fluid density, C_p is the fluid heat capacity, k_0 is the rate constant, T_f is the temperature of the feed stream, C_{A0} is the inlet concentration of reactant A, ΔH_r is the heat of reaction, T is the temperature of reactor and T_c is the temperature of cooling jacket. Table 2.6 lists the CSTR parameters used.

Table 2.6: Reactor Parameter Values

Parameter	Value
V	$100m^3$
E_A/R	$8750K$
$\frac{UA}{V\rho C_p}$	$2.09s^{-1}$
k_0	$7.2e10s^{-1}$
T_f	$350K$
C_{A0}	$1mol/L$
$\frac{\Delta H_r}{\rho C_p}$	$-209\frac{Km^3}{mol}$
q	$100m^3/hr$

In this example, one reactor can make multiple products by varying the concentrations of A and B in the outlet stream. The manipulated variable in this optimization is T_c, which is bounded by $200K \leq T_c \leq 500K$ and by a constraint on manipulated variable movement as $\Delta T_c \leq 2K/min$.

2.3.2 Scenarios

The sample problem uses three products over a 24-hour horizon. The product descriptions are shown in Table 2.7, where the product specification tolerance (δ) is $\pm 0.05\, mol/L$.

Three scenarios are applied to each phase of progressive integration of scheduling and control:

1. Process disturbance (C_A)
2. Demand disturbance

Table 2.7: Product Specifications

Product	C_A (mol/L)	Max Demand (m^3)	Price (\$/m^3)	Storage Cost (\$/hr/m^3)
1	0.10	1000	22	0.11
2	0.30	1000	29	0.1
3	0.50	1000	23	0.12

3. Price disturbance

Scenarios A-C maintain the specifications in Table 2.7 but introduce process disturbances, demand disturbances, and price disturbances respectively (See Table 2.8). Scenario A introduces a process disturbance to the concentration in the reactor (C_A) of $0.15\,mol/L$, ramping uncontrollably over 1.4 hours. Scenario B introduces a market update with a 20% increase in demand for product 2. Scenario C shows a market update with fluctuations in selling prices for products 2-3. The starting concentration for each scenario is $0.10\,mol/L$, the steady-state product conditions for product 1.

Table 2.8: Scenario Descriptions

Scenario	**Time** (hr)	**Disturbance** Product 1	Product 2	Product 3
A	2.2-3.8	————	0.15 mol/L	————
B	3.1	+0 m^3	+200 m^3	+0 m^3
C	2.1	+0 $\$/m^3$	-9 $\$/m^3$	+6 $\$/m^3$

2.4 Results

The results of implementation of each phase for each scenario are discussed and presented in this section. Each problem is formulated in the Pyomo framework for modeling and optimization [58, 175]. Nonlinear programming dynamic optimization problems are solved via orthogonal collocation on finite elements [176] and the APOPT and COUENNE MINLP solvers are utilized to solve all mathematical programming problems presented in this work [177, 178]. For comparative purposes, profits are compared to those of Phase 3 due to its centrality in performance.

2.4.1 Scenario A: Process Disturbance

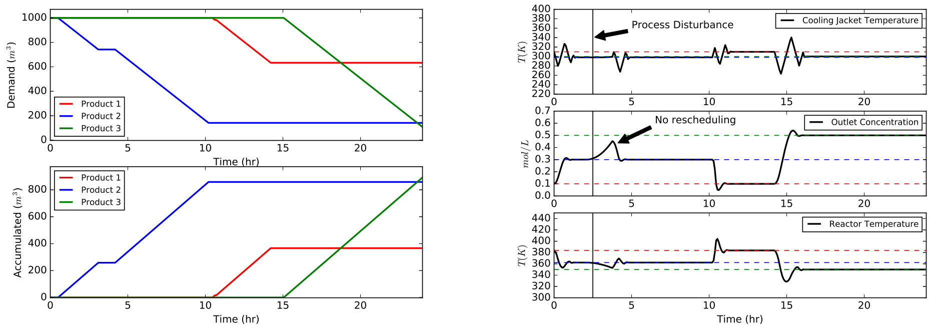
In Scenario A, Phase 1 has a poor schedule due to a lack of incorporation of process dynamics into scheduling. The durations of grade transitions, as dictated by process dynamics, are unaccounted for. However, the production amounts or production durations for each product are optimized based on selling prices. The order is selected based on storage costs, clearly leading to longer grade transitions than necessary. The schedule maximizes production of higher-selling products 2 and 3. Phase 1 does not recalculate the schedule after the process disturbance, holding to pre-determined transition timing.

Phase 2 follows the same pattern as Phase 1 due to its lack of incorporation of process dynamics. Phase 2 recalculates a schedule after the process disturbance, but because it does not account for process dynamics, it cannot determine that it would be faster to transition to Product 3 from the disturbed process state than to return to Product 2. Thus, the production sequence remains sub-optimal. However, the recalculated schedule enables more profitable Product 2 to be produced than in Phase 1 as the timing of transition to Product 1 is delayed due to the disturbance by the recalculation. Phase 2 illustrates benefit that comes from frequent schedule recalculation rather than from scheduling and control integration.

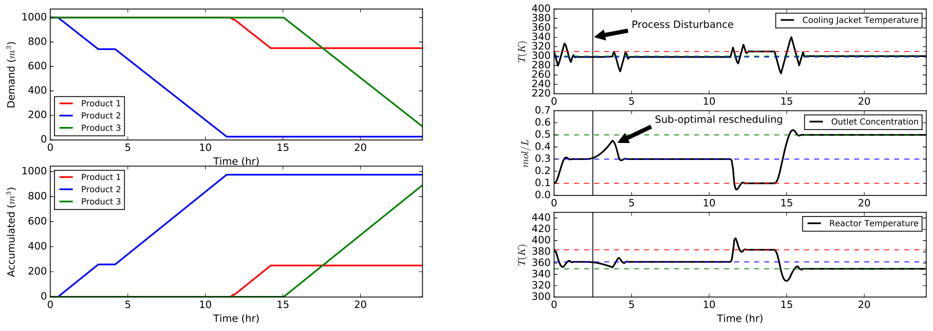
Phase 3 does not react optimally to the process disturbance because it has a fixed schedule, but its initial schedule is optimal due to the incorporation of process dynamics and the resultant minimization of grade transition durations. Phase 3 illustrates benefit originating solely from scheduling and control integration, without schedule recalculation. Phase 4 optimally reschedules with understanding of transition behavior from the disturbed state to each steady-state operating condition, transitioning to Product 2 immediately after the disturbance. Phase 4 demonstrates the premium benefits of both reactive or frequent rescheduling and from scheduling and control integration.

2.4.2 Scenario B: Market update containing demand fluctuation

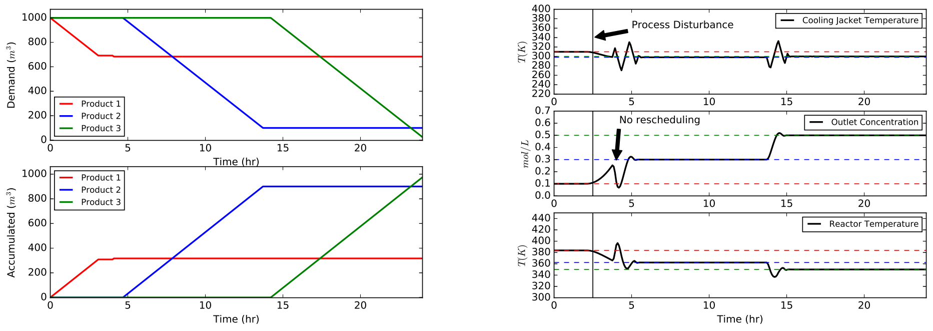
As in Scenario A, the production order for Phases 1-2 is sub-optimal due to a lack of incorporation of process dynamics in scheduling, or a lack of integration of scheduling and control. Phase 2 improves performance over Phase 1 by reacting to the market update and producing



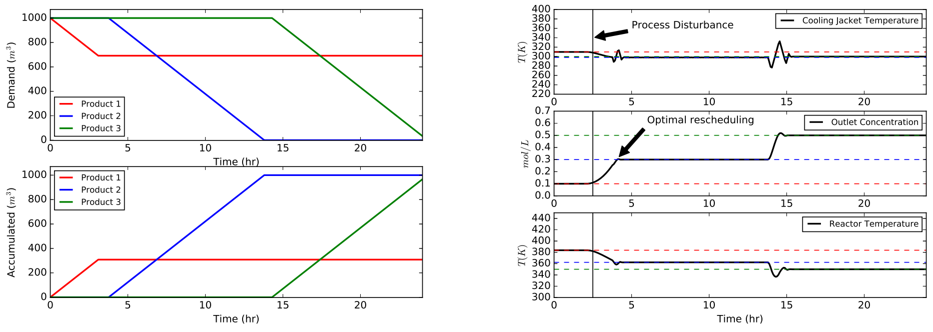
(a) Phase 1: Segregated, Fixed Schedule



(b) Phase 2: Segregated, Reactive Schedule



(c) Phase 3: Integrated, Fixed Schedule



(d) Phase 4: Integrated, Reactive Schedule

Figure 2.5: Scenario A: Process disturbance.

Table 2.9: Results: Scenario A

Phase	**Description**	**Profit**		**Production** (m^3)		
		(\$)	(%)	Product 1	Product 2	Product 3
1	Segregated, Fixed Schedule	3114	(-38%)	367	858	908
2	Segregated, Reactive Schedule	3942	(-21%)	317	900	992
3	Integrated, Fixed Schedule	4983	(+0%)	308	1000	983
4	Integrated, Reactive Schedule	7103	(+43%)	308	1000	983

more profitable Product 2, which had a surge in demand, illustrating again the benefits of reactive scheduling. Phase 3 integrates control with scheduling, resulting in an optimal initial schedule minimizing transition durations. The benefits from integrating scheduling and control (Phase 3) and the benefits of reactive scheduling (Phase 2) are approximately the same in Scenario B, differing in profit by only a negligible amount. However, incorporating both reactive scheduling and scheduling and control integration (Phase 4) leads to a large increase in profits. The initial and recalculated schedules in Phase 4 have optimal production sequence, utilizing process dynamics information to minimize grade transition durations. Additionally, recalculation of the integrated scheduling and control problem after the market update allows for increased production of the highest-selling product, leading to increased profit.

Table 2.10: Results: Scenario B

Phase	**Description**	**Profit**		**Production** (m^3)		
		(\$)	(%)	Product 1	Product 2	Product 3
1	Segregated, Fixed Schedule	6033	(-19%)	367	967	908
2	Segregated, Reactive Schedule	7446	(+0.1%)	133	1200	908
3	Integrated, Fixed Schedule	7441	(+0%)	317	1000	992
4	Integrated, Reactive Schedule	8676	(+17%)	308	1200	800

2.4.3 Scenario C: Market update containing new product selling prices

As in Scenarios A-B, the production order for Phases 1-2 is sub-optimal due to a lack of incorporation of process dynamics in scheduling. However, reactive rescheduling after the price fluctuation information is made available results in a large profit increase from Phase 1 to

Phase 2, demonstrating the strength of reactive scheduling even without scheduling and control integration.

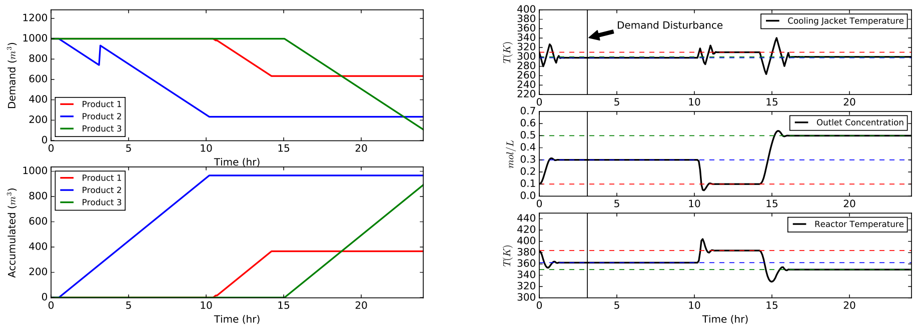
Phases 3-4 have an optimal production sequence due to the integration of scheduling and control, leading to higher profits than the corresponding segregated phases. This illustrates again the benefits of scheduling and control integration. Like Phase 2, Phase 4 reschedules when the updated market conditions are made available, producing less of product 2 and more of products 1 and 3 due to the price fluctuations. This leads to a leap in profit compared to Phase 3. Phase 4 with both scheduling and control integration and reactive or more frequent scheduling is again the most profitable phase.

Table 2.11: Results: Scenario C

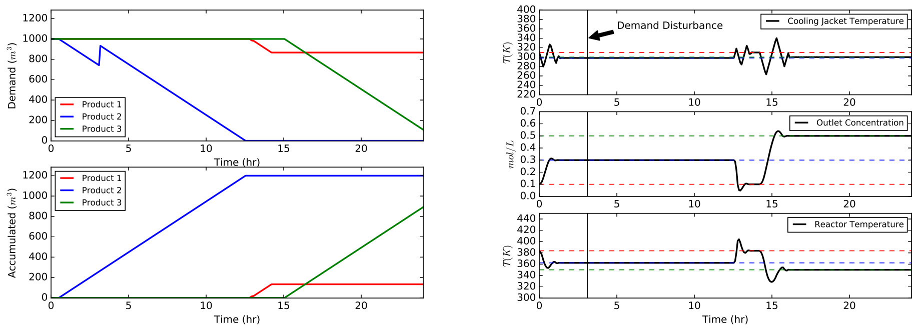
Phase	**Description**	**Profit**		**Production** (m^3)		
		(\$)	(%)	Product 1	Product 2	Product 3
1	Segregated, Fixed Schedule	3758	(-16%)	367	967	908
2	Segregated, Reactive Schedule	4879	(+9%)	967	367	908
3	Integrated, Fixed Schedule	4466	(+0%)	317	1000	992
4	Integrated, Reactive Schedule	5662	(+27%)	1000	317	992

2.5 Conclusions

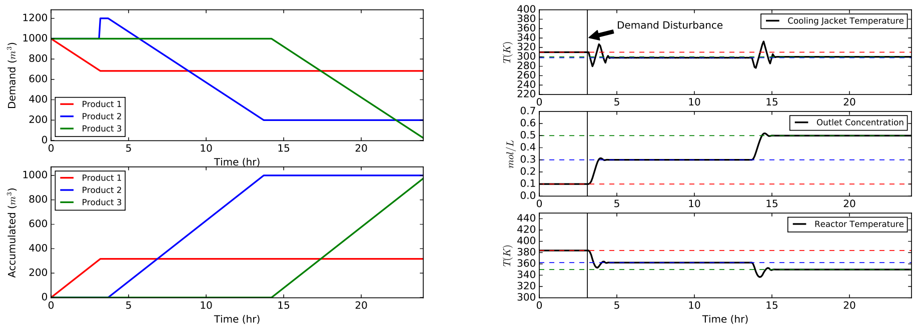
This work summarizes and reviews the evidence for the economic benefit from scheduling and control integration, reactive scheduling with process disturbances and market updates, and from a combination of reactive and integrated scheduling and control. This work demonstrates the value of combining scheduling and control and responding to process disturbances or market updates by directly comparing four phases of progressive integration through a benchmark CSTR application and three scenarios with process disturbance and market fluctuations. Both ISC and reactive rescheduling show benefit, though their relative benefits are dependent on the situation. More complete integration (applying ISC in closed-loop control, rather than just the scheduling) demonstrates the most benefit.



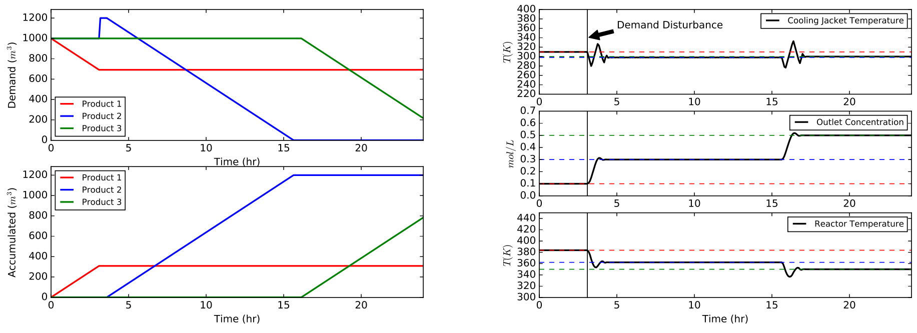
(a) Phase 1: Segregated, Fixed Schedule



(b) Phase 2: Segregated, Reactive Schedule

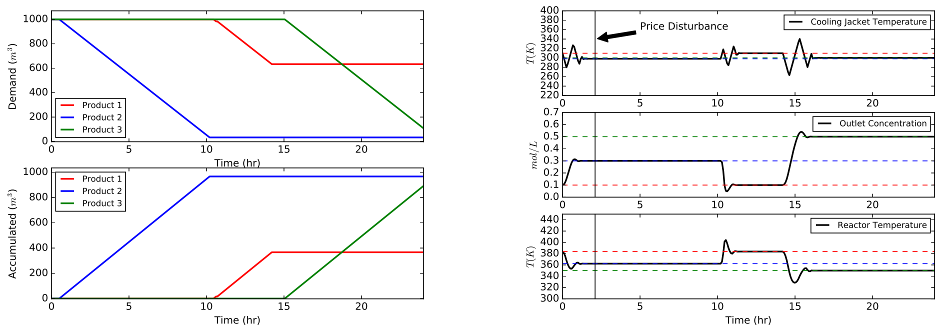


(c) Phase 3: Integrated, Fixed Schedule

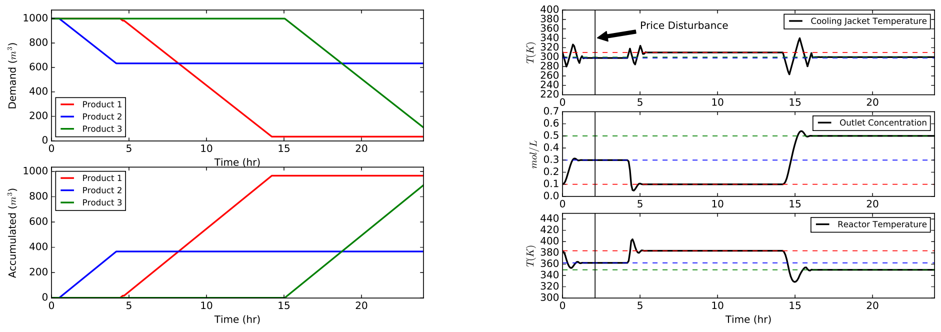


(d) Phase 4: Integrated, Reactive Schedule

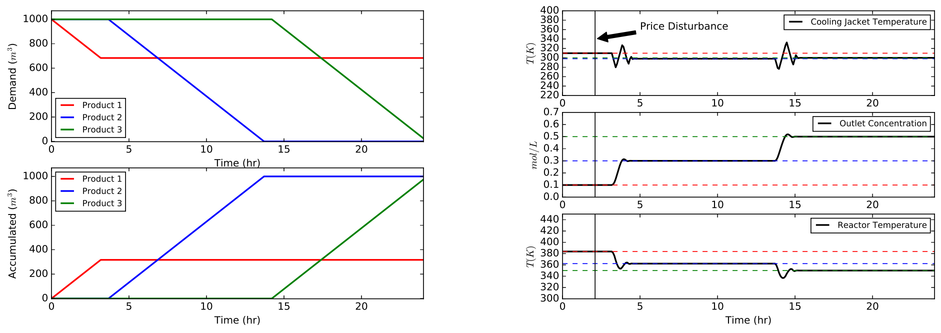
Figure 2.6: Scenario B: Market update (demand disturbance).



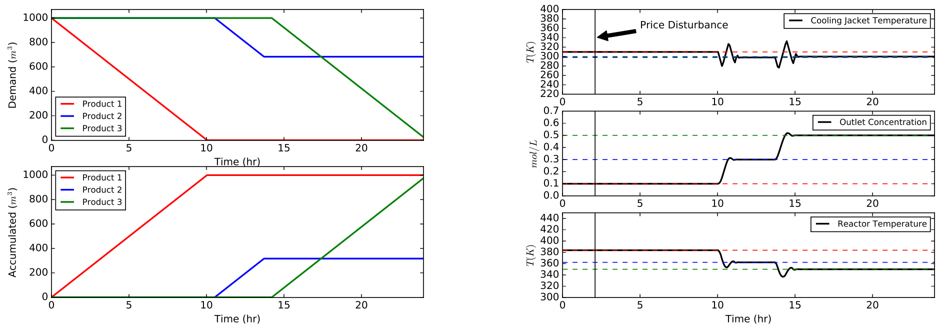
(a) Phase 1: Segregated, Fixed Schedule



(b) Phase 2: Segregated, Reactive Schedule



(c) Phase 3: Integrated, Fixed Schedule



(d) Phase 4: Integrated, Reactive Schedule

Figure 2.7: Scenario C: Market update (price disturbance).

CHAPTER 3. COMBINED MODEL PREDICTIVE CONTROL AND SCHEDULING WITH DOMINANT TIME CONSTANT COMPENSATION

This chapter was published as Beal, L. D., Park, J., Petersen, D., Warnick, S., and Hedengren, J. D., 2017. "Combined model predictive control and scheduling with dominant time constant compensation." *Computers & Chemical Engineering,* **104**, pp. 271–282

3.1 Introduction

Time-of-day energy pricing for electricity and natural gas pose a challenge and opportunity for industrial scale manufacturing processes. In many manufacturing processes in which Model Predictive Control (MPC) is well-established, such as downstream refining and petrochemicals, there is lost opportunity when advanced process control only operates at certain conditions but must be turned off when unit production is lowered [51]. A challenge with changing production rates is that the dynamics of a process often change dramatically with throughput. Empirical models identified at high throughput rates are often inaccurate and lead to poor control performance at low production rates. An opportunity with time-of-day pricing is to temporarily reduce the consumption of energy-intensive processes for periods when the energy costs are sufficiently high. During off-peak periods, the production rate is increased or more energy-intensive product grades are produced to take advantage of low energy costs. With typical daily cycles in energy costs, a cyclical operation forms that the original advanced control system may not be designed to follow. When production targets are not set to maximize to production constraints, MPC may switch to maximize energy efficiency or other secondary objectives. Set point targets traditionally come from a real-time optimization (RTO) application that optimizes a steady state operating point for the plant [144].

Segregated control and scheduling structure is historically due to computational factors that limit application complexity [14]. As a result, the control and scheduling fields have grown independently and without coordination, leading to a loss of opportunity from combining the applications [143, 144]. By combining process scheduling of set points with control, the inefficiencies of application layering are avoided. One such inefficiency that results from application layering is the infeasibility on the control layer of individual solutions pass from the supervisory layer [19]. Scheduling applications frequently do not consider grade transition times because of the large combinatorial look-up table that would be required to consider all possible transitions. Additionally, objectives of individual solutions can oppose each other. For example, the controller does not consider the most economical route to reach a target set point given by the scheduler or steady-state optimizer [16].

The computational barriers for combined control and scheduling are diminished with improved computer hardware and adaptation of algorithms to the hardware. Algorithmic barriers are being overcome with a number of key contributions that are opening several fronts of development [57, 180]. Hardware or network resources such as multi-core, cloud-based, and graphics processing units (GPUs) provide access to previously inaccessible computing power. However, advanced architectures such as GPUs for optimization impose some limitations on the type of problems that can be solved because the algorithms have not yet been adapted to take full advantage of the architecture [181].

Economic MPC (EMPC) [53, 182] uses an objective function that maximizes a profit function rather than targeting a set point as in standard MPC. Including the profit function directly in the MPC application ensures that decisions are directly driven by economic considerations. The profit function also provides guidance on product scheduling, although work on EMPC up to this point has focused on single products. The drawback of this approach is that EMPC generally requires a short time horizon such that the longer horizon required for scheduling constraints and objectives cannot be met [53]. MPC for supply chain management [183] is an alternative strategy that extends the control horizon to schedule product movement through a distribution network.

Dynamic Real-Time Optimization (DRTO) also has an economic objective function but augments a steady state Real-Time Optimization (RTO) with select differential equations that capture the salient and dominant dynamics of a process [3, 140, 184]. One drawback of RTO is that the process must be at steady-state [185] to perform data reconciliation. RTO has traditionally been applied to processes that do not have grade transitions but are dominated by changing economics, disturbances, and slow dominant dynamics. With dynamics included, DRTO can be solved more frequently than RTO applications and can be solved during periods of transient disturbances, during startup, or during shutdown periods. RTO calculations are typically performed every hour to every day while DRTO optimizes the transition between steady-state conditions. Like EMPC, DRTO does not manage multiple sequential product campaigns as a scheduler.

Complete integration of scheduling and control requires an extended prediction horizon to plan the production sequence as well as near-term control actions. Two integration approaches are referred to as top-down (add control and dynamics to a scheduling application) or bottom-up (add scheduling to a control algorithm) [14]. An early top-down implementation includes differential and algebraic equations in the scheduling application [29]. Another method is the scale-bridging model (SBM) in which a simplified model of process dynamics is embedded in the scheduling application [21,22,35]. A benefit of this method is disturbance rejection [186]. Algorithms include Benders' decomposition [23] for problems that have a large-scale structured form and Dinkelbach's algorithm [24] for non-convex problems that require global optimization methods. Applications of combined scheduling and control include batch processes [19, 147], polymer reactors [164], parallel Continuously Stirred Tank Reactors (CSTRs) [30], and an electrical grid that responds to current and future price signals [187].

Variable electrical pricing incentivizes reduced consumption during peak hours [188]. It is desirable to match generation to consumption, but the adoption of more renewable energy requires producers and consumers to respond to price signals [189–191]. Energy producers may expose consumers to time-of-day pricing to discourage consumption during peak hours [192]. Scheduling operation of chemical processing [4, 193], oil refining [194], and air separation [195] are some examples of industrial units that can shed electrical load during peak hours, typically in the middle of the day. Many cooling-limited processes also operate more efficiently at night [193]. Periodic constraints can be used to optimize a typical daily cycle.

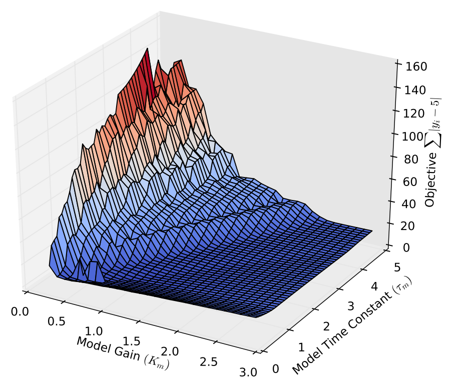
Prior work in scheduling and control integration has been centered around slot-based, continuous-time scheduling formulations. The benefit of discrete-time formulation has been shown [193]. However, the non-linear discrete-time formulation proved computationally difficult. The purpose of this work is to restrict the dynamic model to linear form while capturing benefits of the integration of scheduling and control. There is a large installed base of advanced controls that utilize linear models [51]. A unique aspect of this work is a time-scaling algorithm that adjusts the linear dynamic model based on residence time calculations with a theoretical foundation for first-order systems. The time-scaling approximation is applied to higher order, finite impulse response models that are common in industrial practice. These linear models are used in the combined scheduling and control application.

3.2 Time-Scaling with First-Order Systems

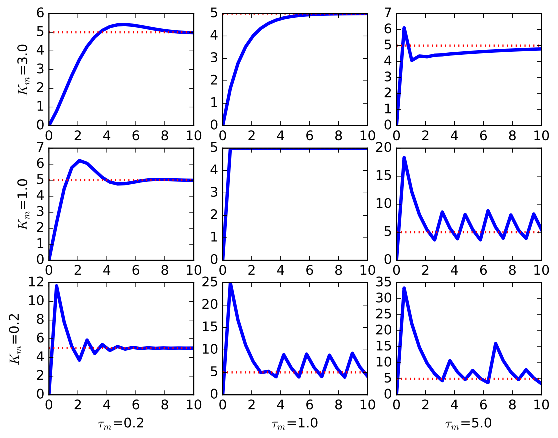
It is well known that linear MPC performance degrades with changes in the actual process time constant or gain [196]. This effect has been quantified for MPC where there is model mismatch in the time constant or gain of a first order system. A simple example is where the actual system is described by a single differential equation as $\tau_p \frac{dy}{dt} = -y + K_p u$ with a process time constant of $\tau_p = 1$ and a process gain of $K_p = 1$. An MPC controller with objective $\sum_{i=1}^{20} |y_i - 5|$ drives the response from a set point of 0 to 5. The controller model is similar to the process but with variable model time constant of τ_m and gain K_m in the equation $\tau_m \frac{dy}{dt} = -y + K_m u$. Common industrial practice is that acceptable MPC performance can be achieved with gain mismatch less than 30% ($0.7 \leq K_m \leq 1.3$) and time constant mismatch less than 50% ($0.5 \leq \tau_m \leq 1.5$) [196]. The dominant time constant for many industrial processes is characterized by the volume (V) divided by the volumetric flowrate (q) as $\tau_p = {}^{V}/_{q}$. The explicit solution to the first order equation is given by Equation 3.1.

$$y[k+1] = \exp\left(\frac{-\Delta t}{\tau_p}\right) y[k] + \left(1 - \exp\left(\frac{-\Delta t}{\tau_p}\right)\right) K_p \, u[k] \tag{3.1}$$

where y is the output, u is the input, k is the discrete time step, and t is the time. The integrated sum of absolute errors is computed for combinations of K_m and τ_m, each between 0.1 and 5.0. The 3D contour plot shows the control performance over the range of time constant and gain mismatch (see Figure 3.1(a)). A mismatch in the gain (x-axis) and time constant (y-axis) are plotted versus the error. The vertical axis (z-axis) is the integrated absolute value of the objective function for a set point change from 0 to 5. A lower sum of absolute errors equates to better control performance with a minimum at $K_m = 1$ and $\tau_m = 1$ (no model mismatch) as shown in Figure 3.1(b).



(a) MPC objective function

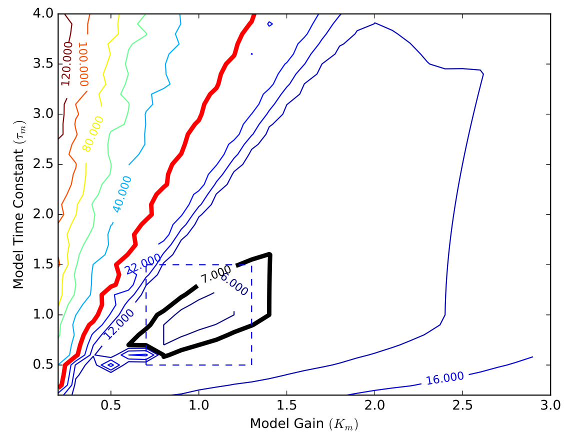


(b) Control performance sensitivity.

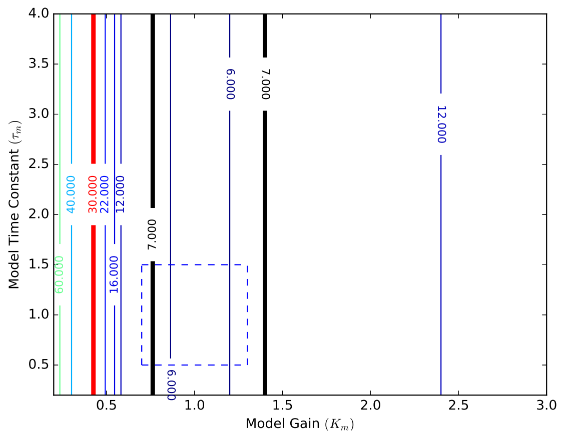
Figure 3.1: Performance degradation of MPC with model mismatch.

Especially poor performance occurs when the model has a higher time-constant (slower) than the actual process. The model in MPC predicts that changes happen slower than are actually the case, leading to a controller that is more aggressive. This aggressiveness translates into overshoot of the set point or even instability. Likewise, a model with a lower gain than the actual process also exhibits poor control performance. The model predicts that larger changes in the Manipulated Variable (MV) are required to drive the process to the new set point. In reality, a smaller adjustment is required and the over-reaction of the controller leads to overshoot and possibly instability. A contour plot of the performance profile gives insight on the performance as shown in Figure 3.2(a). Figure 3.2(b) shows the performance with the time-scaled model. The parallel contour lines show that there is no performance degradation of the controller when τ_m changes such as a production rate decrease or increase. The ability of the MPC to function over all production rates is required for processes that respond to utility price or product demand signals. This simple example shows the increased effectiveness over a wider range of operating conditions with time-scaling while still preserving the linear model.

This result nearly agrees with industry observations, shown with dashed box in Figure 3.2. Another feature of this result is that the combination of high mismatch in time-constant (τ_m) and low mismatch in gain (K_m) combine to form a region of poor control performance. Acceptable or marginal performance is also possible if both τ_m and K_m are both too high or too low. If there is



(a) MPC without time-scaling



(b) MPC with time-scaling

Figure 3.2: Contour plot of performance degradation of MPC. An objective below 7 is acceptable, between 7-30 is marginal, and above 30 (red line) is poor performance.

low mismatch in time-constant (τ_m) and high mismatch in gain (K_m), the controller degradation is manifest as sluggishness and only incremental moves in the MV but not instability.

The time scaling approach adjusts either the controller cycle time or the discrete model time step based on the change in unit throughput q relative to the nominal throughput $\bar{q}$. $\bar{\tau}_p$ is the nominal time constant associated with $\bar{q}$. The modified process time constant is $\tau_p = q \times \left({}^{\bar{\tau}_p}/_{\bar{q}}\right)$ which now has a linear relationship to q. If the process model is not easily adjusted, the cycle time Δt of the controller is adjusted to $\Delta t \times \left({}^{q}/_{\bar{q}}\right)$ to compensate for the changing process dynamics. For first-order systems, this gives an exact representation of the nonlinear dynamics without modifying the original linear model.

3.3 Selective Time-Scaling

Multi-variate and higher-order systems may have certain MV to CV relationships that are known to scale with changing unit throughput while others are invariant to throughput changes. Prior work has focused on decomposition of fast and slow dynamics [197] or variable time-delay of measurements [54, 198]. For systems with multiple MVs and CVs, only the relationships that are sensitive to throughput are scaled. These can be identified with a dynamic process simulator or else by repeating plant identification tests at low and high production rates. A method to scale higher order systems is to transform the linear time-invariant (LTI) model into discrete form. In discrete form, the sampling time is scaled by $\left(\bar{q}/q\right)$ and resampled to preserve the overall model sampling time. As an example of this time scaling approach, consider the 7^{th} order system given by Equation 3.2 as a transfer function in terms of Laplace variable s.

$$G(s) = \frac{CV(s)}{MV(s)} = \frac{1.5}{\left(s^2+0.6s+1\right)\left(0.5s+1\right)^5} \tag{3.2}$$

Suppose that the dynamics of this system depend on the production rate and that the feed rate to the unit is reduced to half of the rate where the model is originally identified. When a time-scaling transformation of $q/\bar{q} = 2$ is applied, the new transfer function is also a 7^{th} order system but with shifted dynamics. The steady state gain of the transfer function is preserved with this method of dynamic transformation. The resulting transfer function is Equation 3.3.

$$G(s) = \frac{CV(s)}{MV(s)} = \frac{1.5}{\left(4s^7+21.2s^6+47s^5+57s^4+42s^3+20s^2+6.2s+1\right)} \tag{3.3}$$

The continuous transfer function is first converted to discrete form with a sufficiently small sampling time. For this case, a sampling time of 0.5 *sec* is chosen for the continuous to discrete transformation. The discrete model sampling time is set to 2 x 0.5 *sec* $=$ 1.0 *sec* based on the time-scaling factor and then the model is resampled to 0.5 *sec* to be consistent with other unscaled MV/CV models. Continuous to discrete transformations and discrete model resampling for LTI models are standard methods [199] and are not repeated here for the sake of brevity. A graphical demonstration of this method is shown in Figure 3.3.

3.4 Scheduling and Control Formulation

An innovation of this work is to combine scheduling and control with linear models and quadratic objective functions for fast solution with Quadratic Programming (QP) [200] or Nonlinear Programming (NLP) solvers. There are multiple methods for formulating tiered price structures. The focus of this work is on creating mathematical expressions that have continuous first and second derivatives and fit within the QP or a QP with Quadratic Constraint (QPQC) framework. In contrast, most modern scheduling applications utilize integer variables, often as binary decision variables. Including integers requires MILP or MINLP solvers, which are significantly slower and more complex.

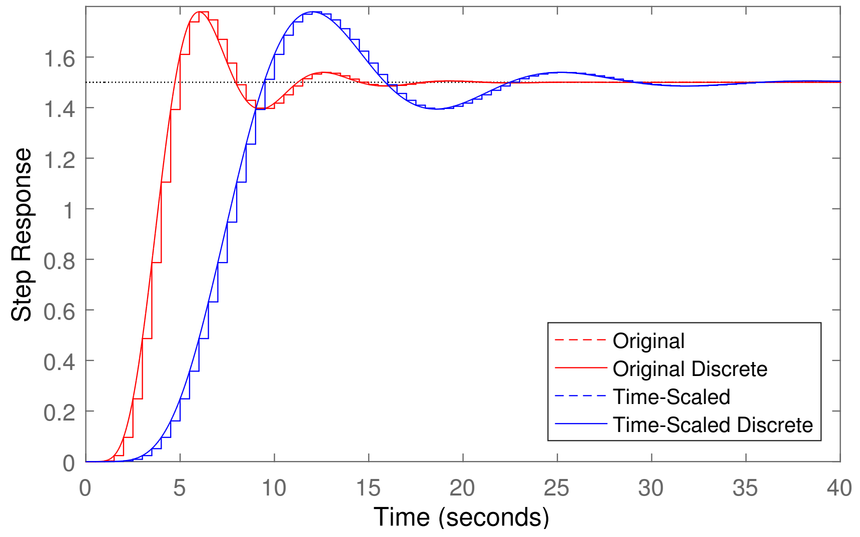


Figure 3.3: Time-scaling of a 7th order system when the feed rate is reduced to half.

3.4.1 MPCC Steps for Product Pricing

The method uses a continuous formulation to logical decisions. It uses a Mathematical Programs with Complementarity Constraints (MPCC) formulation to avoid binary variables that would otherwise accommodate tiered pricing structures [201]. This formulation uses a step function MPCC, using constraints $v_0 y = 0$ and $v_1(1-y) = 0$. v_0 and v_1 are positive slack variables. The complementarity constraints force variables to their bounds – resulting in y being a binary variable at its bounds $[0,1]$.

This condition can be difficult for solvers to find a solution so the condition is typically either solved as an equivalent inequality $(v_0 y \leq 0)$, a relaxed inequality $(v_0 y \leq \varepsilon)$, or included in the objective function $(\min\ v_0 y)$ [202]. The step function MPCC that turns y from 0 to 1 at switching point x_p is shown in Equation 3.4a-3.4d.

$$\min_{v_0,v_1,y} v_0 y + v_1 (1-y) \tag{3.4a}$$

$$x - x_p = v_1 - v_0 \tag{3.4b}$$

$$v_1, v_0 \geq 0 \tag{3.4c}$$

$$0 \leq y \leq 1 \tag{3.4d}$$

To preserve the QP structure, the complementarity constraints are included in the objective function. If the complementarity constraints are included as inequalities a QPQC or NLP solver must be used. With sufficient weighting, the complementarity constraints in the objective function are zero at a final optimal solution but not necessarily along the search path to the solution. Being zero at the solution, the complementarity constraints do not influence other objective terms such as minimizing energy consumption or maximizing profit. The MPCC switching conditions are combined to create a tiered product pricing structure as shown in Figure 3.4.

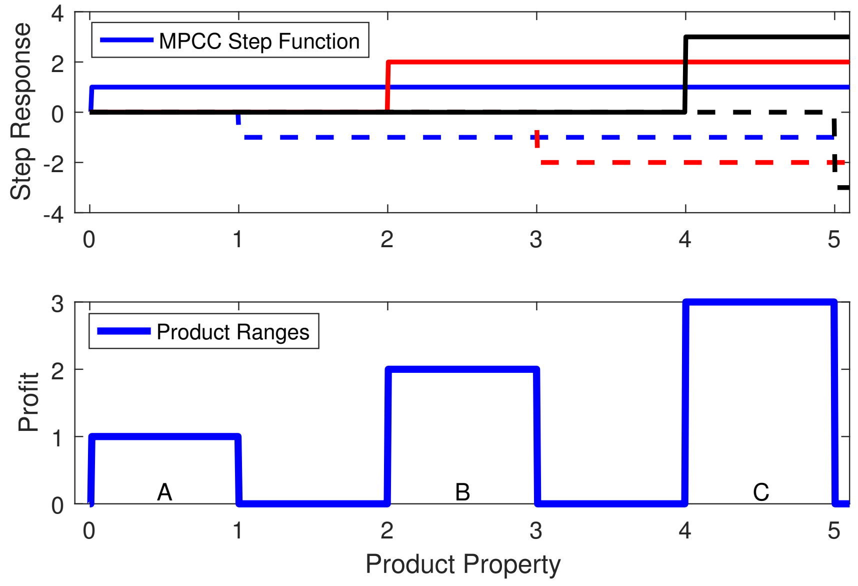


Figure 3.4: Individual MPCC step functions are combined to create a continuous differentiable expression of switching conditions.

Each product has a different value and potentially a different width of specification limits. The positive and negative step functions at different switching points are combined with pricing information to create an objective function with multiple products. The individual steps are shown in the upper subplot of Figure 3.4 while the summation of all steps is shown in the lower subplot. Product A ranges from 0 to 1 and has a price of 1 per unit production. Product B ranges from 2 to 3 and has a price of 2 per unit production. Product C ranges from 2 to 3 and has the highest price of 3 per unit production. The facility has a capacity of two units of production per hour and the minimum amount of production for each product over a 7 hour time window is 2 units of A, 5 units of B, and 3 units of C. Because C is the most valuable product, any spare capacity should favor the production of C. In switching between products, a schedule should account for transition material between grades that does not have value. The speed of a transition is limited by the maximum move rate of the Manipulated Variable (MV) of 1.6 per hour and the dynamics of the process. The dynamics of the process are simply a linear first-order system as $\tau \frac{d(CV)}{dt} = -CV + MV$ with a time constant τ of 1.0.

The grade-specific objective function with a quadratic objective, linear equation, and simple inequality constraints is added to the process model to optimize the timing of grade transition switches. A complete statement of the QP optimization problem is shown in A. The series of three products demonstrates a simple combined control and scheduling problem as shown in Figures 3.5 and 3.6. The cases have different initial conditions and constraints but the same underlying model. The first case has an initial condition for CV of 0.0 at the lower specification limit for product A while the second case starts at CV of 4.5.

The optimizer minimizes the amount of products A and B while meeting the minimum requirements before transitioning to the next grade. This allows the schedule to favor the production of C, the most profitable product. By minimizing the amount of products A and B, the scheduler creates a plan that produces an extra unit of product C. The combined controller and scheduler anticipates grade transitions by shifting the production specification to the upper limit to minimize the transition time to the next grade. This is apparent at time 1.0 hr and 4.2 hr where the product is already transitioning to the new grade just as the minimum required amount for products A and B are produced. The combined scheduler and controller adjusts both the control actions as well as

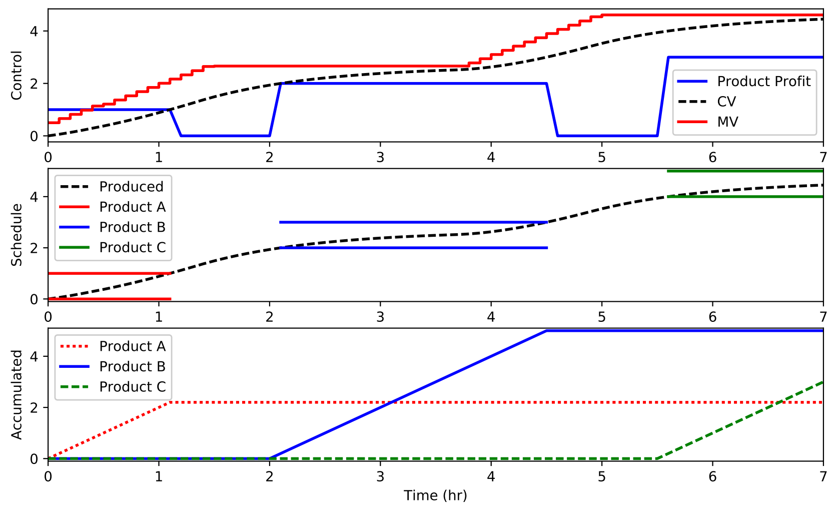


Figure 3.5: The control and scheduling optimization are combined to determine optimal MV movement along with the optimal order and quantity of production for each grade.

the order in which the products are produced. In the next example, the initial condition is shifted to a different starting location to demonstrate this feature of the approach.

The computational time is an important factor for control problems where the algorithm must return a solution within the cycle time necessary to rejected disturbances and maintain process stability. The fully discretized combined scheduling and control problem with 3 products and discrete time points has 2730 variables, 1820 equations, and 910 degrees of freedom. The problem is solved on a Dell R815 server with an AMD Opteron 6276 Processor and 64 GB of RAM. The problem requires 8.6 s to converge with the APOPT solver (version 1.0) or 4.0 s with the IPOPT solver (version 3.10). MPC commonly uses a warm-start from a prior solution to improve efficiency. With a warm-start from the prior time-step, the optimizer requires 0.5 s with APOPT and 3.0 s with IPOPT. Active-set solvers (such as APOPT) and interior point solvers (such as IPOPT) are both effective for large-scale MPC applications although interior point solvers have better performance as degrees of freedom increase [57].

3.4.2 Dynamic Cyclic Schedule

A common method to optimize a schedule in current practice is to place products on a fixed grade wheel where each product is visited sequentially in a rotation. The cyclic schedule visits all grades in a forward and then a reverse order to end up at the original grade and begin the cycle again. If there is a sudden demand for a particular grade out of sequence, the product wheel is still typically traversed in the grade wheel order but there may be a minimal amount of the less desirable grades in favor of the desired grade production. In some processes, such as polyolefin production, there are multiple versions of the grade wheel. One version of the grade wheel may include only commodity products while other wheels or sub-wheels may be a more complex cycles that include less frequently produced products.

The contribution of this work is the discrete time, extended controller and scheduler that is also able to produce a cyclic schedule. However, this cyclic schedule is not fixed but adjusts the sequence of products automatically when updated economic or constraint information is available. The schedule and control action may update every controller cycle (e.g. every minute). Some constraints for scheduling are periodic boundary constraints where the final condition must be equal to the initial condition, contracted quantities that must be produced by a certain date or time, particular equipment limitation, or time-of-day transition constraints. An example of a time-of-day constraint is that certain grade transitions need start-up or shut-down of auxiliary equipment. A constraint may be that the grade transition should happen only during a weekday day-time shift where there is adequate operator support.

The prior example problem is augmented with intermediate production targets and a periodic constraint. The scheduler and controller, shown in Figure 3.6 produces an optimized schedule and control actions. It is a dynamic rather than a static grade wheel because the order of product production and quantity is re-optimized every cycle of the controller. The product order or quantity may change based on changing customer demand, price signals for electricity or feed costs, or disturbances that drive the system to a different state.

The optimized solution meets constraints as well as maximizes the production of product C, the highest value product. The intermediate constraints originate from contracted delivery times and storage capacity of a particular product. In this case, the storage capacity of all products is less than 5 units of production. This requires two deliveries of product B that are scheduled at

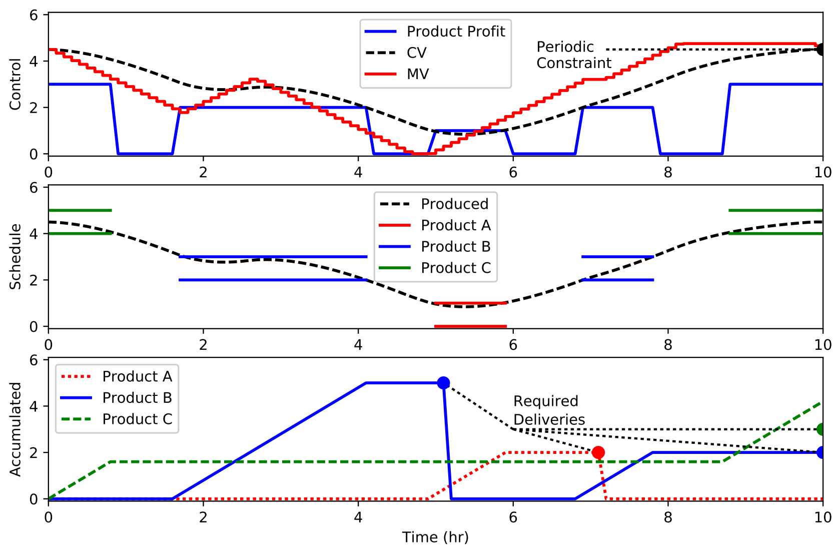


Figure 3.6: Optimized schedule with periodic condition, intermediate production targets, and final production targets. The schedule and control actions adjust to meet constraints and maximize profit (product *C*).

two equally spaced intervals of 5 hrs. The production of product *C* exceeds the required delivery amount because it is the highest value product. Although the initial condition is at product *C*, the controller immediately targets product *B* to meet the required delivery of 5 units of production at 5 hrs. Without the periodic constraint, production of product *C* would be maximized before transitioning because the scheduler evaluates the alternative that a transition back to product *C* results in lost profit potential. However, with the required transition back to product *C*, the scheduler puts excess production of product *C* at the end.

While this method is capable of producing cyclic schedules, the optimizer should begin from current conditions rather than steady-state product conditions to fully integrate control and scheduling. Cyclic schedules combined with online control may lead back to off-spec conditions because of disturbances or because the controller is in a transition. Instead, this method is better suited to a different set of constraints – production amounts and due dates. These constraints give more freedom to the optimizer so the economic objective will improve or be equal to the solution with periodic constraints.

As with adding any constraint, there is potential to make the problem infeasible. Multi-objective optimization statements, such as those used to refine a dynamic grade wheel sequence, can also be posed with a rank-ordered set of constraints with an explicit prioritization of objectives. The ℓ_1-norm dead-band formulation is discussed in more detail in [57] and related to combined scheduling and control production targets in Section 3.4.3. Posing the constraints as multi-objective penalties versus hard constraints allows the problem to remain feasible yet still meet the most important objectives in order of priority.

3.4.3 Acceptable Range of Production Quantity

One drawback to the prior examples is that all spare production capacity is typically placed on the highest value product. Over-production of any product can have the effect of lowering the selling price because of supply and demand market forces. In scheduling, there is often a range of production quantity that is acceptable instead of just a single hard limit. To accommodate this, the scheduling and control algorithm can use an ℓ_1-norm objective function to give a target region for the production quantity, rather than one specific hard limit. Equation 3.5 shows a generalized ℓ_1-norm control formulation used in this work.

$$\begin{aligned} \min_{x,CV,MV} \quad & \Phi = w_{hi}^T e_{hi} + w_{lo}^T e_{lo} + x^T c_x + MV^T c_{MV} + \Delta MV^T c_{\Delta MV} \\ \text{s.t.} \quad & 0 = f\left(\frac{dx}{dt}, x, \frac{dCV}{dt}, CV, MV\right) \\ & e_{hi} \geq x - d_{hi} \\ & e_{lo} \geq d_{lo} - x \end{aligned} \tag{3.5}$$

In this formulation, Φ is the objective function, x is the production quantity per grade. Parameters w_{lo} and w_{hi} are penalty matrices for solutions outside of the production target region. The slack variable e_{lo} and e_{hi} define the error of the dead-band low and high limits. Parameters c_x, c_{MV}, and $c_{\Delta MV}$ are cost vectors for the production quantity (positive values minimize production within region, negative values maximize production within region), the MVs (positive values minimize MV quantities such as energy use, negative values maximize MV quantity such as pump speed correlated to higher efficiency), and change of MVs, respectively. The function f is an open-equation set of equations as functions of x, CV, MV, and time derivatives of x and CV. The demand targets d_{lo} and d_{hi} define lower and upper target limits for production.

This range formulation is not used in this work but is presented to demonstrate one of many ways that this problem structure can be expanded to meet various scheduling needs.

3.5 Application: Continuously Stirred Tank Reactor

A continuously stirred tank reactor (CSTR) is a common benchmark used in nonlinear model predictive control and scheduling applications [14, 185, 193]. This nonlinear application is used to demonstrate the strength of the approach to time-scale based on throughput and combined scheduling & control.

The linear time-scaled model changes dynamics based on reactor flow rate, allowing linear MPC to be used instead of nonlinear MPC. The multi-product scheduling objective is used instead of a simple target tracking to combine the scheduling and control into one application. Rapid convergence is ensured with a linear model and quadratic objective because the problem is convex and because QP solvers are efficient for large-scale systems.

The CSTR application is highly nonlinear because of an exothermic reaction that has the potential to cause rapid reaction of stored reactant and thereby cause a temperature run-away. The CSTR shares characteristics of many industrial processes such as polymer reactors or many refining processes but with much simpler mathematics that are amenable to demonstrating a new approach for control and scheduling. The liquid full reactor is used to convert compounds $A \Rightarrow B$ with constant liquid density (ρ) and heat capacity $\left(C_p\right)$ as shown in Figure 3.7.

The reaction kinetics are first order and irreversible. Reaction of A to B is exothermic with the potential for temperature run-away because of the exponential dependence of reaction rate on temperature, typical of an Arrhenius form for reaction rates. The reactor is well-mixed with reactor concentration and temperature equally distributed and also equal to the outlet measured values. The reactor temperature is regulated with a cooling jacket liquid temperature, T_c. The cooling jacket temperature is normally regulated by adjusting the rate of cooling or the coolant flow rate but in this model the jacket temperature is assumed to be controlled directly and the dynamics are approximated by a maximum rate of change.

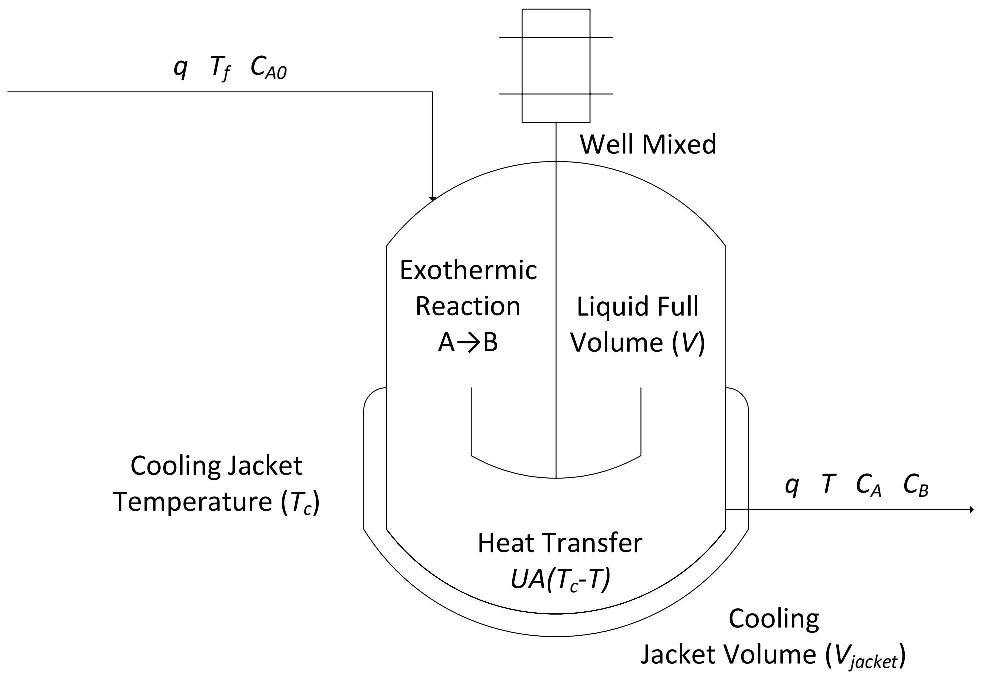

q T_f C_{A0}
Well Mixed
Exothermic Reaction A→B
Liquid Full Volume (V)
Cooling Jacket Temperature (T_c)
q T C_A C_B
Heat Transfer $UA(T_c\text{-}T)$
Cooling Jacket Volume (V_{jacket})

Figure 3.7: Diagram of the well-mixed and liquid-full CSTR. The $A \Rightarrow B$ reaction is exothermic and controlled by a cooling jacket fluid.

The dynamics of the CSTR are dictated by a species and energy balance as shown in Equations 3.6-3.7 and in Figure 3.8.

$$V \frac{dC_A}{dt} = q(C_{A0} - C_A) - k_0\, V\, e^{\frac{-E_A}{RT}}\, C_A \tag{3.6}$$

$$\rho C_p V \frac{dT}{dt} = q\rho C_p (T_f - T) - k_0\, V\, e^{\frac{-E_A}{RT}}\, C_A \Delta H_r - UA\,(T - T_c) \tag{3.7}$$

where V is the volume of the reactor, C_A is the concentration of reactant A, q is the volumetric flowrate, C_{A0} is the inlet concentration of reactant A. The energy balance includes terms UA as an overall heat transfer coefficient times the tank surface area, C_p as the reactor fluid heat capacity, ρ as the fluid density, T_f as the temperature of the feed stream, T as the temperature of reactor, and T_c as the temperature of cooling jacket fluid. Terms related to the reaction include ΔH_r as the heat of reaction, E_A as the activation energy, R as the universal gas constant, and k_0 as the pre-exponential factor. Table 3.1 lists the CSTR parameters and the associated values.

A regression is shown with varying orders for an Output Error (OE) time-series model in Figure 3.8. Second and third order models have nearly the same fit to the nonlinear regression

Table 3.1: Reactor Initial Conditions and Parameter Values

States	Description	Initial Condition
C_A	Concentration of reactant A	$0.1\ \frac{mol}{L}$
C_B	Concentration of product B	$0.9\ \frac{mol}{L}$
T	Reactor temperature	$386.82\ K$
Manipulated Variables	Description	Initial Value
T_c	Cooling jacket temperature	$300\ K$
Parameters	Description	Value
q	Volumetric flowrate	$12\ m^3/hr$
V	Tank volume	$40m^3$
C_{A0}	Feed concentration of reactant A	$1\frac{mol}{L}$
UA	Overall heat transfer coefficient	$5000\ \frac{W}{K}$
C_p	Heat capacity of reactor fluid	$0.239\ \frac{J}{kg\ K}$
ρ	Density of reactor fluid	$1000\ \frac{kg}{m^3}$
T_f	Feed temperature	$350\ K$
ΔH_r	Heat of reaction (exothermic)	$-11.95\frac{MJ}{mol}$
k_0	Pre-exponential factor, rate constant	$1.8e10\frac{1}{hr}$
E_A/R	Activition energy divided by R	$8750K$

while a first order model is insufficient in capturing the process dynamics. A second simulation is performed with the production rate reduced from 12 m^3/hr to 6 m^3/hr.

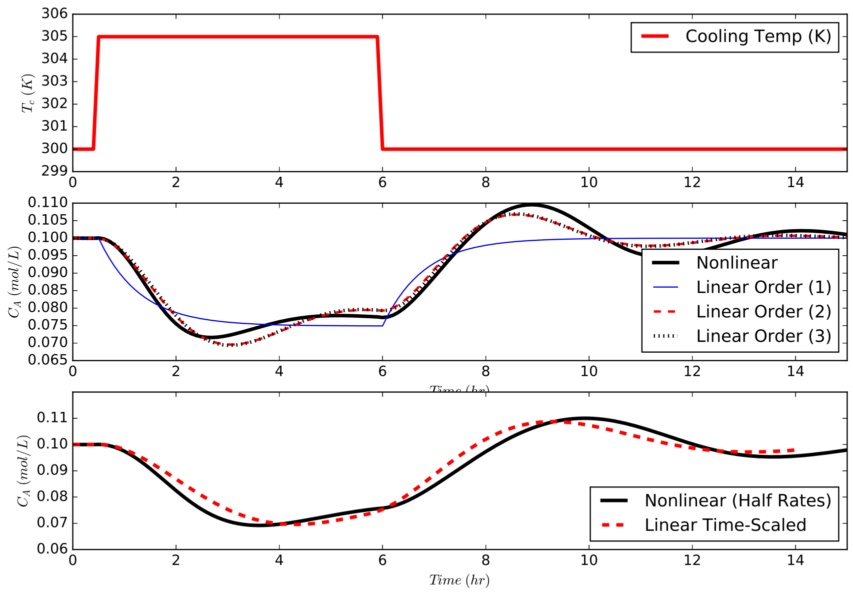


Figure 3.8: Step tests in the jacket cooling and linear model regression of the step response.

The concentration response of the reactor at half production rate is shown in the bottom subplot of Figure 3.8 with the second order time-scaled model that originally fit with simulated data from the full production rate simulation. The time-scaled approach is effective at capturing the essential process dynamics without re-fitting a process model at low rates.

One reactor makes multiple products by varying the concentrations of A and B in the reactor. The cooling jacket temperature T_c is the manipulated variable in this optimization. The production rate of the reactor changes throughout a typical 24-hour period because of time-of-day pricing that necessitates a cut-back of production during peak energy prices and when cooling capacity is limited. The production rate is modified by adjusting the total flow through the reactor (q) between half-rates at 6 $\frac{m^3}{hr}$ and full-rates at 12 $\frac{m^3}{hr}$.

There is demand for three products with quantities that must be met in the schedule over a 48-hour horizon. The product descriptions and quantities are shown in Table 3.2.

Table 3.2: Product Summary with Demand and Price

Product	C_A (mol/L)	Demand (m^3)	Price (\$/m^3)
P_1	0.12 ± 0.01	120	9
P_2	0.25 ± 0.01	130	11
P_3	0.35 ± 0.01	150	6

The most valuable is product P_2 while the least valuable product is P_3. Although P_3 has the lowest price, it also has the highest required quantity. Spare capacity in the production facility favors product P_2. A potential drawback to always switching to P_2 at the end of a campaign is that there is lost material during the transition back to P_2. An improved strategy is to make excess P_2 when the schedule requires it to meet a minimum target demand instead of transitioning back to P_2 near the end of the time period. The combined control and scheduling solution is shown in Figure 3.9 over a 48 hour time period with 6 minute time intervals. The problem is discretized with orthogonal collocation on finite elements. Each 6 minute segment is integrated with Radau quadrature. The resulting QP or QPQC problem is solved with a nonlinear programming solver with a simultaneous solution of the objective and equations. If specific control action is needed at more frequent intervals, the first steps of the horizon could be adjusted to meet a required controller cycle time. This would develop a near-term move plan and simultaneously solve the scheduling optimization problem with one application. The feed flow rate is decreased to half each day between the hours of 08:00 and 18:00 as is done with some energy intesive processes that exploit time-of-day electricity pricing. The addition of production rate as a decision variable and the associated cooling contraints is outside the scope of this work because the model becomes nonlinear. The simultaneous control and scheduling of production rate and product grade sequence is the topic of a future publication (see [193] for preliminary results). The top sub-plot is the sequence of control moves to drive the system to produce on-spec products and transition between products. The middle sub-plot shows the grade specifications and the concentration in the reactor. The bottom sub-plot is the total production of each grade with the minimum required as indicated by the circle markers at hour 48. The production rate is non-zero during transition periods because the total rate includes production of off-spec as well as on-spec grade material.

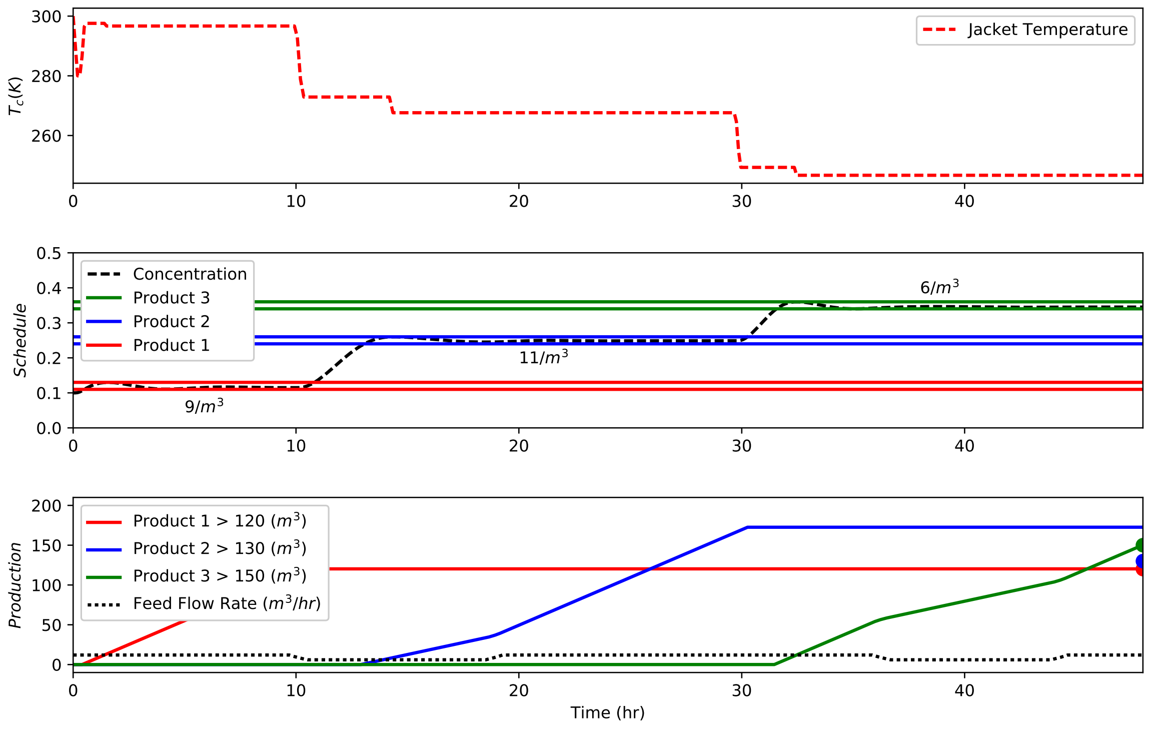


Figure 3.9: Combined control and schedule optimization results.

The control influences the scheduling solution and the scheduling solution gives the controller target values. The controller adjusts the jacket temperature (T_c) to minimize the transition time between grade specifications and remain within the grade limits. The controller response improves with knowledge of the scheduling targets because pro-active pre-transition movement shows that the product specification are pushed to an upper limit right before the transition begins. One factor that affects the schedule is the speed of the transition from one product to another. The control dynamics are due to limitations in the manipulated variable movement and the process dynamics affecting the speed of attaining concentration range. The speed of the control response is factored into the schedule as lost production time when transition (off-spec) material is produced. Blending of transition material is one strategy to dilute the off-test material in prime products. This strategy is not considered in this approach but could be included as a constraint on the amount of reblend fraction that is allowed in the final product.

The scheduling profit function is an application and adaptation of Equations 3.4a-3.4d. Figure 3.10 displays the slack and step function for each of the product limits along with the overall profit function.

The slack variables and complementarity conditions combine to create discrete steps with a function that has continuous first and second derivatives. For this problem, the complementarity constraints were included as constraints and in the objective function to assist the optimizer and ensure binary decision variables. The transition time to each product is not specified beforehand but is a result of the optimization solver finding a sequence of grade transitions that maximize overall profit. The individual steps of Figure 3.10 occur when one of the product grade limits is crossed. One of each of the slack variables for each step is non-zero before or after the step. The slack variables guide the solver with a continuous form of a discrete grade switching function. The overall solution is intuitive because the optimizer produces excess product P_2 (highest volumetric price) but meets only the minimum production quantities for the lower value products (P_1 and P_3).

A weakness of this method is that the CSTR is nonlinear but the solution is computed with a linearized model. Linearization neglects features of the process that may be optimized if a suitable process model more accurately represented the physical process. A key contribution of this work is that the solution to the combined control and scheduling problem is relatively computationally

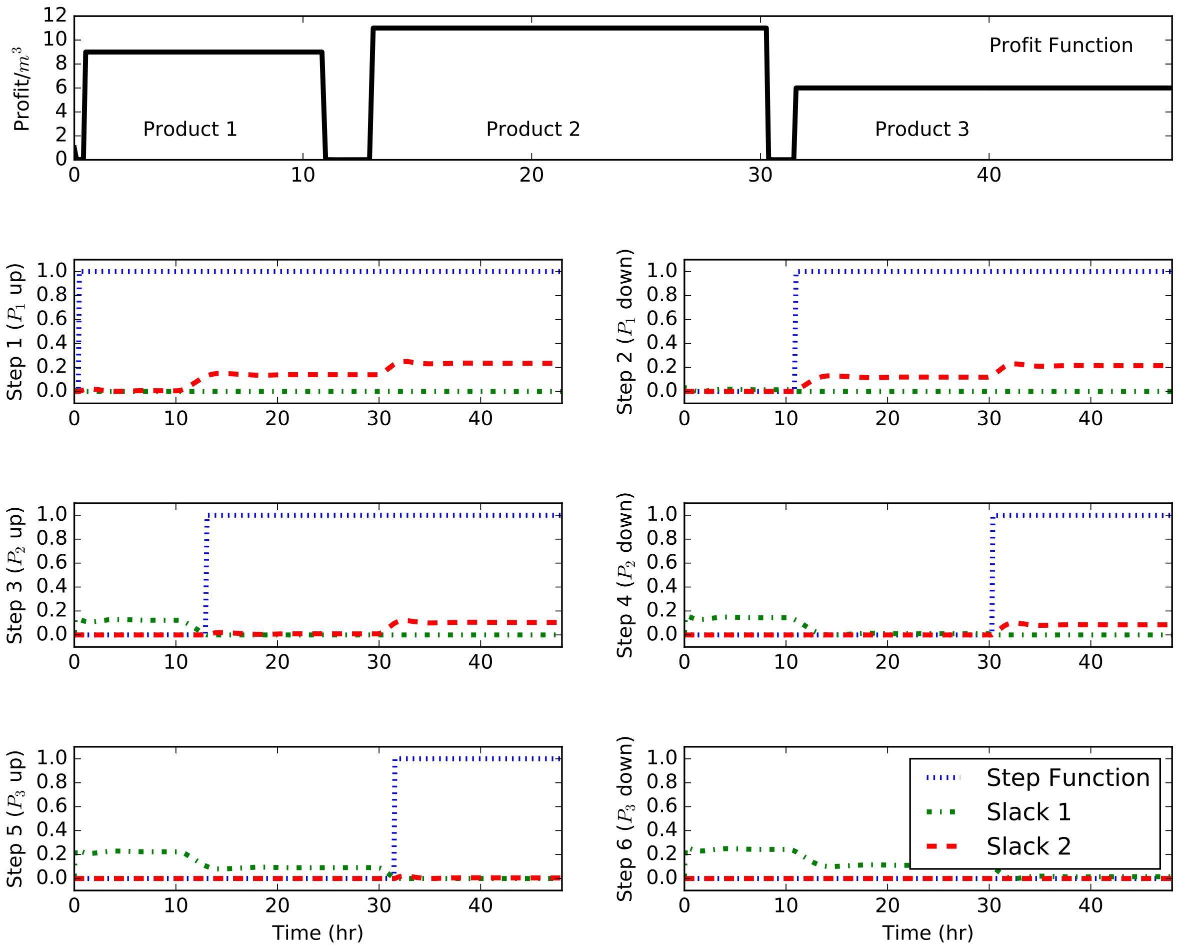


Figure 3.10: Profit function and individual step functions for each product.

inexpensive in comparison to a full nonlinear solution. In this case, the linearized model with time-scaling has a total of $22,048$ variables and $16,640$ equations and is solved on a Dell R815 with an AMD Opteron 6276 Processor and 64 GB of RAM. After initialization, the problem requires 23.7 s to converge with the IPOPT solver. This is representative of the cycle time of the application as it repeatedly solves to reject disturbances and as new objective information or demand constraint information is available. This speed, coupled with the inclusion of a process model sufficient for control, allows this formulation to be utilized for on-line control, on top of providing an advanced schedule. This is the epitome of combining scheduling and control – a fully unified optimization that can replace both layers.

A nonlinear version of this application is future work that will be reported in a subsequent publication. A key difference with the full nonlinear solution is that the solution in this work is several orders of magitude faster because of the quadratic objective and linear time-scaled model. Although combined scheduling and control with linear models or feedback linearization does not always accurately predict a highly nonlinear system, the linearized solution is a valuable starting point to initialize a nonlinear solution. Also, many processes are not highly nonlinear and this approach is likely suitable for systems that are already controlled with linear MPC.

3.6 Conclusion

A combined scheduling and control application is enabled by an MPCC objective function, discrete-time, and linear time-scaling of process dynamics based on production rate changes. The objective of this work is to extend traditional linear MPC applications with a scheduling objective that allows for rapid convergence for real-time applications. One drawback of this work is that nonlinearities are not included in the application. These nonlinearities are the subject of future work. The method is tested on a CSTR application that includes three grades over a 48 hour time horizon with 6 minute time intervals. The embedded controller simulates realistic transition times between each of the products. The scheduling objective determines the order and quantity of production at each grade even with half-rate reduction during peak electricity demand. The formulation is sufficiently fast enough, and includes enough process dynamics, to be utilized in on-line control. This presents a fully unified optimization that fulfills the roles of, and can replace, both control and scheduling for a comparable system. Although the method is demonstrated on the CSTR application, this formulation can be applied to other systems by replacing the model, pricing structure, and constraints of the scheduler.

CHAPTER 4. INTEGRATED SCHEDULING AND CONTROL IN DISCRETE-TIME WITH DYNAMIC PARAMETERS AND CONSTRAINTS

This chapter was published as Beal, L. D., Petersen, D., Grimsman, D., Warnick, S., and Hedengren, J. D., 2018. "Integrated scheduling and control in discrete-time with dynamic parameters and constraints." *Computers & Chemical Engineering,* **115**, pp. 361 – 376

4.1 Introduction

Current process control and optimization strategies are typically divided into major sections including base layer controls, advanced controls, real-time optimization, scheduling, and planning [144]. Each of these levels works at a different time scale, ranging from milliseconds to seconds for base controls, up to weeks or months at the planning level.

Each of these levels receives a minimal amount of information to fulfill an objective to simplify models and decrease computation time. However, the lack of information communicated between the levels creates lost opportunities. For example, scheduling problems have historically focused on the quantity and time-line of product manufacturing, without much knowledge of the dynamics of the manufacturing process. Thus, the "optimal" solution determined in scheduling is sometimes impossible to implement in practice within the required time to transition between products in continuous manufacturing [19]. Further, the objectives of different manufacturing layers can sometimes counter each other. For example, a control goal to reach a set point could potentially conflict with a scheduling goal to maximize profits [16].

This segregated manufacturing structure is largely an artifact of the development of process control and computational limits during these developmental periods [14]. Thus, each level has developed within an isolated domain, without much inter-level coordination, sometimes at the expense of truly optimal solutions [14].

4.1.1 Economic Model Predictive Control and Dynamic Real Time Optimization

With ever-increasing computational power, the segregation of optimization is being re-analyzed through efforts such as model predictive control for supply-chain management [183], combined nonlinear estimation and control [180, 203], dynamic real-time optimization (DRTO) [3, 140, 184], and economic model predictive control (EMPC) [53, 204]. These past efforts have proven valuable in practice [144].

DRTO has an economic objective function similar to that of a scheduler. DRTO is solved more frequently than scheduling problems and leverages the predictive power inherent in a dynamic first-principles model to calculate intermediate set points used by MPC for optimal product transitions [3, 53].

EMPC mixes the benefits of the optimization layers with an objective function centered around profit or reducing operating expenses, rather than reaching a setpoint, and is therefore similar to a scheduler. However, EMPC uses a very short time horizon like MPC [53]. However, both EMPC and DRTO only consider one product at a time, and therefore do not replace a scheduler.

Other researchers are more fully integrating control and scheduling (SC) in an attempt to achieve even more optimal solutions. Suggestions for, and early implementation of, fully combined scheduling and control go back at least a decade [29].

4.1.2 Integrated Scheduling and Control and Computational Capacity

The benefits of integrated scheduling and control have been explored extensively in recent work, and algorithms and technology which further enable these large-scale problems have been steadily advancing during recent years. Today, both computing power and algorithms have advanced so far that nonlinear programming problems (NLPs) with over a million variables can be solved [2]. Not only have solvers grown in capacity, but also in speed [184]. These developments and potential for future growth in algorithm capacity and speed prompt investigation of a new paradigm of integrated scheduling and control which could lead to improved process solutions through more complex problem formulations.

Previous Work

Extensive research has been performed in the new field of integrated scheduling and control. Multiple review articles have been written on the topic of integrated scheduling and control, ranging from reviews on integration feasibility to addressing uncertainty in the integrated problem [15–17, 157, 205]. Some researchers have investigated incorporating explicit process dynamics in the scheduling model with differential and/or algebraic constraints [29], even for multi-product parallel continuous stirred tank reactors (CSTRs) [30]. Multi-objective optimization approaches have also been investigated for combined scheduling and control of CSTRs [158]. Another approach called the scale-bridging model (SBM) uses a simplified model that encompasses most of the important dynamics that can be used in the scheduling framework [21, 22, 35].

Predictive control system integration into refinery scheduling models has been investigated [206]. Integrating scheduling and control has been shown to optimize transition times in a polymerization reactor model, although the optimization problem grows rapidly with an increasing number of products [164]. Combined scheduling and controller selection for optimal grade transitions in polymerization processes has also been investigated [146, 151]. Closed-loop implementation of simultaneous scheduling and control has been shown to effectively re-calculate an optimal schedule in the presence of significant disturbances [20]. Fast model predictive control [33] and dual feedback structures [25] have been proposed to reduce the computational requirements of on-line, closed-loop implementation of combined scheduling and control. A traveling salesman approach has also recently been suggested [207].

Some researchers employ decomposition techniques to the SC problem. Past research in this field includes segregating production sequence and product demand [34], applying Benders' decomposition framework to particular problems [23, 153, 154], using Dinkelbach's algorithm to find a global optimum in on-line implementations [24], and using Lagrangian heuristic decomposition to reduce the computational burden of the combined problem [31].

Some researchers have explored the integration of scheduling and control in batch processes. The possibilities of direct inclusion of process dynamics into batch scheduling was first discussed over a decade ago [208]. Recent research continues to explore integrating process dynamics into batch scheduling [19]. Multi-parametric model predictive control [32], state equipment networks [147], and two-phase (off-line and on-line) architectures [209] have been applied to integrate scheduling and control for batch processes. Chu and You investigated scheduling and control in batch processes, investigating moving horizon approaches [25], decomposition through surrogate modeling [155], and decomposition into a bi-level problem solvable with a game theory approach [156].

Work has also been done to integrate design, scheduling, and control [161, 163, 165] as well as to integrate planning, scheduling, and control [159, 160]. Several review articles have outlined organizational and other challenges to integrating scheduling and control in chemical processes [14–16,157]. This work considers the simultaneous integration of scheduling and control for multi-product continuous chemical processes.

4.1.3 Demand Response

Demand response (DR) is an illustrative example of the benefit of considering dynamic constraints and parameters in SC optimization for chemical processes. As the electrical grid transitions to a smart grid and dynamic electricity prices become available, stakeholders are being empowered to perform more informed energy transactions [187, 189, 190]. Along with residential and commercial systems, industrial systems are among those that can increasingly take advantage of the variable price of electricity [210].

Demand response seeks to manage both volatile demand and renewable energy in order to increase efficiency of the electrical grid. DR incentivizes consumers to behave in ways that benefit the electrical grid as well as themselves by utilizing variable pricing to reduce consumption during peak hours when the reliability of the grid is jeopardized [188]. Generation should match consumption in order to maintain grid reliability [194]. DR is a major reason why variability of energy prices is expected to increase [192]. Industrial manufacturing processes can benefit from DR by decreasing energy consumption when the cost of electricity is high and increasing consumption when electricity costs are low. The possibility of demand-side pricing and constraints in energy markets creates new opportunities to achieve economic benefit from SC in chemical processes [15].

Although residential consumers make up the largest portion of electrical grid consumers, tremendous opportunities exist for industrial participants [194]. Previous efforts to quantify the benefits of DR for the industrial sector include petroleum refining [194], chemical processing [4], gas production (Air Separation Unit) [195], aluminum smelting [211], and steel production [212]. The idea that chemical processes can be used as a "battery" to store energy from the grid generated during non-peak hours in the form of chemical products was introduced by Baldea [148]. He discusses methods to enable chemical process operators to interact optimally with utility operators to enable effective DR. Mendoza-Serrano and Chmielewski [194] introduce the concept of using economic model predictive control (EMPC) to respond to demand in electric power systems, using a refinery as an example application. Xu and Wang [213] investigate a feedback control approach to address energy consumption in a job-shop scheduling problem. Tong *et al.* [210, 214] incorporates chemical process dynamics in formulations accounting for DR in chemical process scheduling and control. Tong *et al.* account for process dynamics during transition periods and incorporate DR in their objective; however, the process is considered at steady-state during production periods, not allowing the optimization to alter operation during production periods to respond to dynamic constraints and dynamic energy price. Additionally, Tong *et al.* consider parameter tuning for linear controllers rather than utilizing nonlinear model predictive control in the SC formulation [214]. The SC problem is decomposed into a scheduling problem with constant transitions and a control problem (optimal parameter tuning). The authors mention the results are sub-optimal, but are progress toward true optimality.

This work utilizes a case study of a CSTR model with a first-order, irreversible reaction to illustrate the benefits of adjusting operations based on periodic electricity price changes. Moreover, periodic effective maximum cooling is added to the model. During the heat of the day, effective maximum cooling is reduced compared to night-time operation. To simulate these dynamic cooling and price conditions, periodic constraints of both effective maximum cooling and electricity price are utilized in the optimization. The discrete-time optimization is able to adjust manipulated process variables throughout the entire horizon to respond to these dynamic energy price and dynamic cooling constraints.

4.2 Problem Formulation in Discrete-time

Typical scheduling optimization seeks to maximize profit (P) by changing scheduling variables (x_s) such as the order of products, subject to scheduling constraints such as demand, production rate, storage costs, etc. There are two main types of models for scheduling of chemical processes: discrete-time and continuous-time [215]. The majority of previous work on integrated scheduling and control utilizes continuous-time scheduling formulations with integrated process dynamics to enable dynamic optimization of both scheduling and control [21–25, 27, 29, 30, 32–35, 158, 168, 186]. Recent work also demonstrates the possibilities of using discrete-time formulations to integrate scheduling and control [37, 38, 162]. To account for dynamic process constraints and dynamic scheduling parameters (e.g. dynamic market conditions), this work utilizes a fully discrete-time method for the integrated scheduling and control problem. The discrete-time formulation enables dynamic constraints and dynamic market conditions to be considered in integrated optimization of both production sequence and process operation during and between production slots, enabling optimization of effective demand response during production periods, effective optimization of grade transition timing, and effective optimization of process control during both production and transitional periods. A generalized scheduling optimization is shown in Equation 4.1, where time is discrete.

$$\begin{aligned} \underset{x_s}{\text{maximize}} \quad & P(x_s, y, t) \\ \text{subject to} \quad & \text{scheduling constraints} \end{aligned} \tag{4.1}$$

Unlike scheduling which considers economic constraints but not process dynamics, MPC includes process dynamics but inherently contains no economic considerations. MPC drives a process to a setpoint by manipulating process variables (x_c) such as flow rates, subject to the process model (e.g. reaction rates, mass balance), generalized by Equation 4.2, where time is discrete.

$$\begin{aligned} \underset{x_c}{\text{minimize}} \quad & \left\| y_{model} - y_{sp} \right\| \\ \text{subject to} \quad & \text{process model} \end{aligned} \tag{4.2}$$

EMPC adjusts MPC by maximizing profit (P) rather than minimizing error to a setpoint using the same dynamic process model. The economic objective is reminiscent of a scheduler.

$$\begin{aligned} \underset{x_c}{\text{maximize}} \quad & P(x_c, y, t) \\ \text{subject to} \quad & \text{process model} \end{aligned} \tag{4.3}$$

By combining the constraints from EMPC (the process model) and the scheduling optimization (scheduling variables such as prices and demand), with an economic objective function over the same discretized time horizon, a fully unified control and scheduling optimization is achieved.

However, schedulers do not consider the same set of process variables that a controller process model considers. Thus, a link is required between scheduling variables and process variables. Namely, linking a product on/off binary variable ($B_{i,t}$, representing production of product i at time t) from typical scheduling formulations to the associated process variable (y_p), as defined by product specifications. Production (B) is typically a binary variable while process variables are typically continuous. Since most product specifications include a tolerance, this linking can be done through a step function. Figure 4.1 shows this relationship for a sample product whose specification is $x = 1.5$ with a tolerance of 0.5.

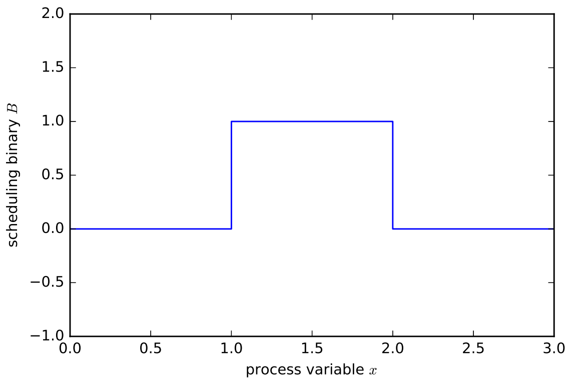


Figure 4.1: A generic linking function between the continuous process specification variable x and the binary scheduling variable for the associated product b.

With the linking function in place, economic optimization based on both scheduling and control economics is possible with both a process model and scheduling constraints, as shown in Equation 4.4. This is the proposed paradigm of combined scheduling and control explored in this work.

$$\begin{aligned} \underset{x_s, x_c}{\text{maximize}} \quad & P(x_s, x_c, y, B) \\ \text{subject to} \quad & \text{scheduling constraints} \\ & \text{process model} \\ & \text{linking function} \quad B(y_p) \end{aligned} \tag{4.4}$$

The formulation in this work consists of a large-scale set of nonlinear differential and algebraic equations (DAE) that describe a MIDO problem. The continuous horizon of the problem is discretized by orthogonal collocation onto finite elements (see Figure 4.2), becoming a large system of algebraic equations containing binary variables $B_{i,t}$, which determine the current on-specification production of each product i at time t.

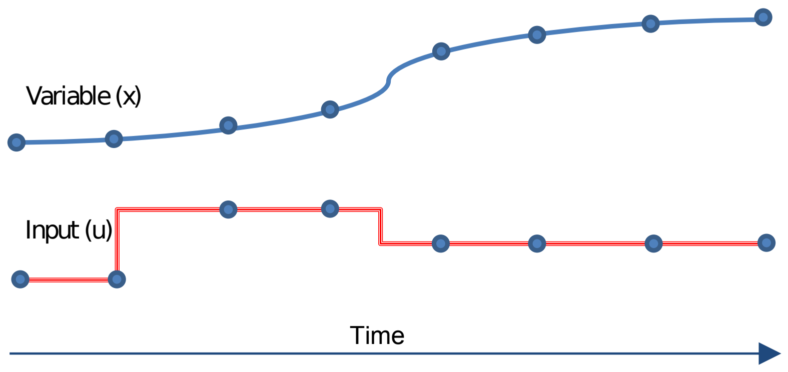


Figure 4.2: Variables are discretized over a horizon onto finite elements by orthogonal collocation.

The objective function is shown in Eq. 4.5, where B is the binary variable that determines if product i is produced at time t, Π is the price of product p, q is the rate of production at time t, n is the number of finite elements, and O is the operational expenditure at time t.

$$\text{maximize} \quad \sum_{t}^{n} \left[\sum_{i}^{n} (q_t \Pi_i B_{i,t}) - O_t \right] \tag{4.5}$$

This discrete-time SC formulation is inherently able to account for dynamic constraints and parameters throughout a prediction horizon by modifying constraint or parameter values at each finite element across the horizon. This improves upon continuous-time, slot-based formulations in which a system is traditionally considered steady-state or constant during production periods, ruling out the possibilities for consideration of dynamic constraints or parameters during production slots [210].

4.2.1 Linking Functions

The full discrete-time integrated scheduling and control problem accounting for full nonlinear process dynamics presented in this work produces a complex and difficult mixed-integer nonlinear (MINLP) problem. Fine-time resolution of the discrete-time SC problem dictates a large number of integer variables $B_{i,t}$. However, on-line or frequent implementation of integrated scheduling and control is beneficial for rejecting process disturbances and responding effectively to market fluctuations [14, 15, 20]. This requires a sufficiently light computational time requirement for the SC problem.

To reduce the computational requirements of the nonlinear discrete-time problem accounting for nonlinear process dynamics, this work recommends continuous relaxations of the linking function (or a "pseudo-binary" approach) to ease the computational requirements of the MINLP problem to enable solution via gradient-based NLP. This section provides various options for linking functions. A few selected methods are tested on the SC problem (Equation 4.4) with sample results of profit and solution time reported.

The first recommended linking function (a "hard constraint") could be formulated as shown in Equation 4.6 or 4.7, where *spec* is the product specification with tolerance tol. In this form, B is zero outside of product specs, but will be driven to one (the upper variable bound) by the economic objective function when on spec —effectively producing a step in B. These constraints are simple and linear or quadratic.

$$B(|spec - x| - tol) \leq 0, \quad B \in [0,1] \tag{4.6}$$

$$B((spec - x)^2 - tol^2) \leq 0, \quad B \in [0,1] \tag{4.7}$$

Similarly, mathematical programming with complementarity constraints (MPCC) can be used to simulate a step function by combining two modified "signum" MPCC formulations, as shown in Equation 4.8 [38]. This formulation provides derivatives for the solver to seek out the product and B is fixed at one when on-spec through constraints rather than objective function encouragement. However, mathematical programming with equilibrium constraints (MPEC) naturally include dependent active inequality constraints which add significant difficulty for the solver (although this can be partially resolved through structured regularization [216]). Further, this method introduces six slack variables (s) per product, per time discretization. All the slack variables must be positive and s_5 and s_6 must additionally be less than or equal to one.

$$x - spec - tol = s_1 - s_2 \tag{4.8a}$$

$$x - spec + tol = s_3 - s_4 \tag{4.8b}$$

$$s_1 * (1 - s_5) + s_2 * s_5 \leq 0 \tag{4.8c}$$

$$s_3 * (1 - s_6) + s_4 * s_6 \leq 0 \tag{4.8d}$$

$$B = s_6 - s_5 \tag{4.8e}$$

In contrast to these binary methods, we also present continuous relaxations to the binary step function (or a "pseudo-binary" approach). For example, Eq. 4.9 provides a continuous gradient with immediate objective function benefit for the solver to recognize the location of products with respect to process state x. In Eq. 4.9a, h represents the max height of the function and must exceed 1. In this format, f exceeds 1 in the range of product specifications and is within [0,1] elsewhere.

$$f(x) = h10^{log(1/h)/tol^2(spec-x)^2} \tag{4.9a}$$

$$B(x) \leq f(x), \quad B \in [0,1] \tag{4.9b}$$

Low values of h present a short, wide hill with clean, far-reaching gradients. To force a square function like a true binary variable, h is increased so f goes to 0 outside the product specifications. Then, through Eq. 4.9b, the function is capped at 1. The economic objective function maximizes B to 1 whenever the concentration is within the associated product specification. Figure

4.3 shows the resulting linking function, demonstrating the wide reach provided by low values of h to the steep binary approximation of large h.

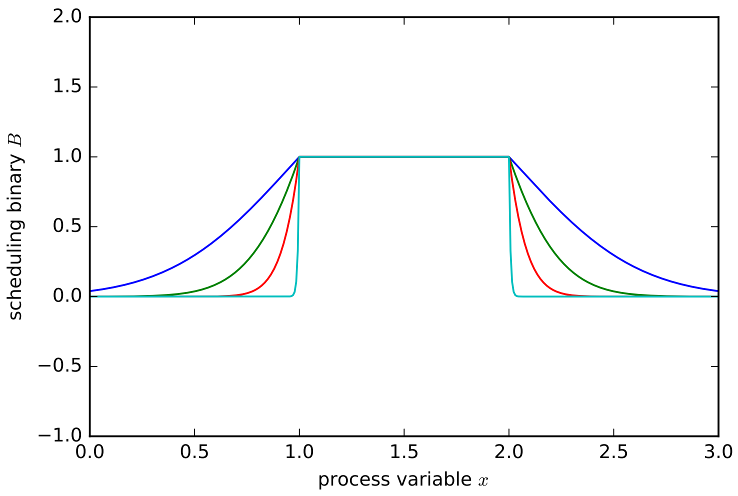


Figure 4.3: A sample plot of Function 4.9 with increasingly large h, beginning with a gentle slope for clean, far-reaching derivatives, progressing towards a strict binary step function.

In this work, h is manually increased and resolved iteratively, with each solution initializing the next. This also helps avoid the numerical difficulties presented by the steep gradients of large h values by being very close to the solution as the numerical challenges increase. This method is related to, but has the opposite effect of, the barrier method used in interior point solvers [217]. It is the authors' opinion that this form would be better implemented within a solver where h could be updated on a per-iteration basis. This is a point of future work.

This function is suitable for this use because product specification variables are typically within a known, relatively small bound. Thus, the function f can provide a gradient through the entire range with initially small h.

Similar to Equation 4.9, Equation 4.10 provides a continuous relaxation by combining two sigmoid functions, scaled to a maximum value of one.

$$B(x) = \frac{\frac{1}{1+exp(k(spec-tol-x))} + \frac{1}{1+exp(k(-spec-tol+x))} - 1}{\frac{1}{1+exp(k(-tol))} + \frac{1}{1+exp(k(-tol))} - 1} \tag{4.10}$$

In Equation 4.10, low values of k provides clean, far-reaching gradients while large k approximates the step function. k in this method is the counterpart to h in Equation 4.9. Figure 4.4 shows this function with increasing values of k.

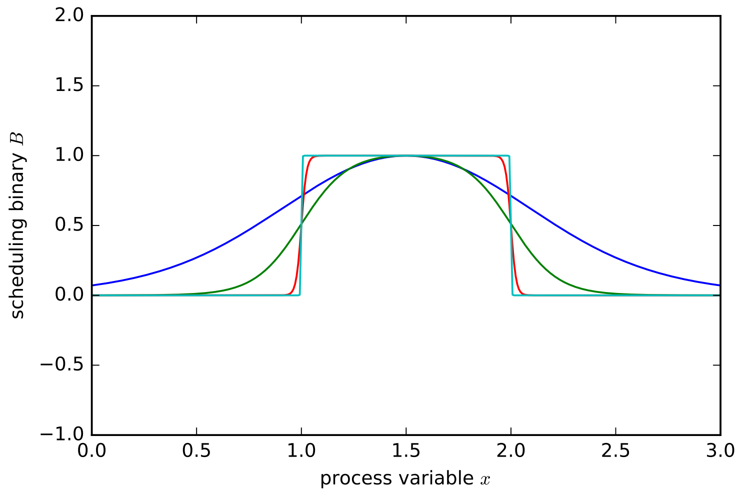


Figure 4.4: A sample plot of the sigmoid function with increasingly large k, beginning with a gentle slope for clean, far-reaching derivatives, progressing towards a strict binary step function.

Sample results

Each of the proposed linking functions is capable of finding a solution. However, since the provided gradients differ, the solver takes a different path to each solution. This can yield slightly different solutions as each problem may fall in different local optima. For the problem provided in Equation 4.4 (further described in Section 4.4), Figure 4.5 shows a set of sample results of both the end profit and the time required to achieve the solution. These results all use continuous variables for B, but each solution is sufficiently close to binary. An attempt at solving these problems with actual binary variables, in an MINLP using the APOPT solver [177], did not solve in under 10,000 seconds.

For this problem, the pseudo-binary method (Equation 4.9) consistently returns the highest profit and is therefore recommended for finding the initial schedule solution. However, Equation 4.7 is the fastest method and is therefore recommended for closed-loop control once a highly detailed solution is obtained and used to initialize the next control move. For other problems with different dynamics or constraints, a different linking function may work better.

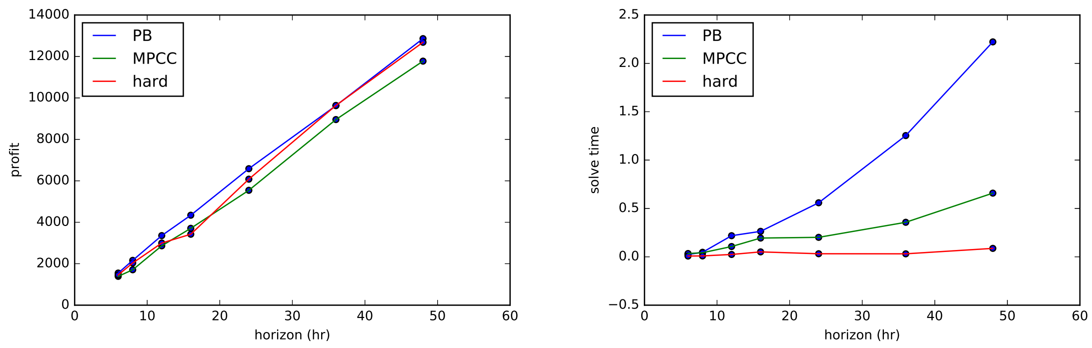


Figure 4.5: A comparison of different linking functions at various horizon lengths. The figures show that Eq. 4.9 (PB or "pseudo-binary") consistently returns the best schedule, while Eq. 4.7 (hard) consistently yields the fastest solution.

4.3 Strategies for Computational Tractability

To further reduce the time required to solve the MINLP problem, a computationally light continuous-time scheduling optimization is used to initialize the discrete-time problem. Both feedback linearization and nonlinear model-predictive control (NMPC) are used to estimate the transition times in the continuous-time scheduling problem used for initialization.

A pseudo-binary variable strategy is presented to make the discrete-time mixed-integer dynamic optimization (MIDO) problem solvable by NLP. An initialization strategy is presented to further shorten the computational time for the discrete-time problem by using a simpler continuous-time, slot-based scheduling problem. The transition times needed to solve this continuous-time, slot-based scheduling problem are estimated using two alternative techniques: feedback linearization and nonlinear model predictive control. A transposition of the continuous-time scheduling solution to discrete-time is presented to initialize the discrete-time problem.

4.3.1 Continuous-time Scheduling Initialization

This work further lightens the computational burden of the discrete-time SC problem through initialization. As shown in Safdarnejad *et al.* [218], initialization of discrete-time problem variables at each finite element to values close to the optimal solution reduces the computational time required. In this work, initialization of binary variables ($B_{i,t}$) and key process variables at each finite element is employed to reduce the computational burden of each iteration of combined scheduling and control.

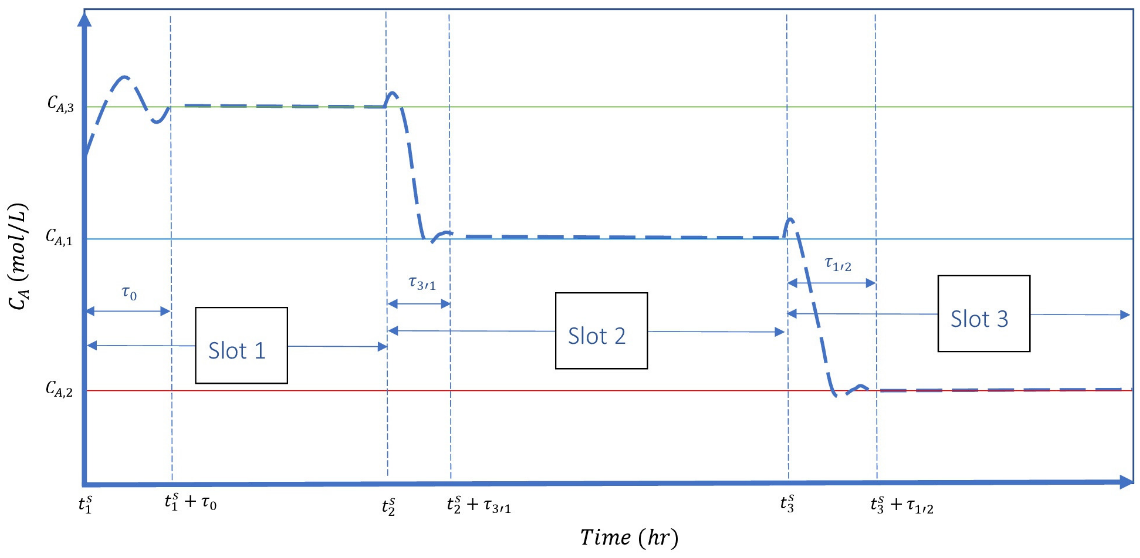


Figure 4.6: Continuous-time scheduling divides the future horizon into time slots that consist of a transition period $\tau_{i'i}$ (where product i' is made in slot s - 1 and product i is made in slot s) followed by the production period for product i.

If implemented closed-loop, initialization can be provided by the solution of a previous iteration. However, continuous-time scheduling optimization is selected as a computationally light way to initialize in the case that a previous solution is unavailable. Continuous-time scheduling divides a future time horizon into time slots composed of a transition period followed by a production period, as shown in Figure 4.6. A continuous-time scheduling optimization is used to initialize the discrete-time problem in this work. This continuous-time optimization seeks to maximize profit while observing scheduling constraints. The objective function is formulated as follows:

$$\begin{aligned} \max_{z_{i,s}, t_i^s, t_i^f \forall i,s} \quad & J = \sum_{i=1}^{n} \Pi_i \omega_i - q c_{rm} T_m \\ \text{s.t.} \quad & \text{Eq.}\, 4.12 - 4.18 \end{aligned} \tag{4.11}$$

where T_m is the makespan, n is the number of slots, $z_{i,s}$ is the binary variable that governs the assignment of product i to a particular slot s, t_s^s is the start time of the slot s where product i is made, t_s^f is the end time of the same slot, Π_i is price per unit of product i, q is production rate, c_{rm} is raw material cost, and ω_i represents the amount of product i manufactured,

$$\omega_i = \sum_{s=1}^{n} \int_{t_s^s + \tau_{i'i}}^{t_s^f} z_{i,s} q \, dt \tag{4.12}$$

where $\tau_{i'i}$ is the transition time between product i' made in slot s - 1 and product i made in slot s. The time points must satisfy the precedence relations

$$t_s^f > t_s^s + \tau_{i'i} \qquad \forall s > 1 \tag{4.13}$$

$$t_s^s = t_{s-1}^f \qquad \forall s \neq 1 \tag{4.14}$$

$$t_n^f \leq T_m \tag{4.15}$$

which require that a time slot be longer than the corresponding transition time, impose the coincidence of the end time of one time slot with the start time of the subsequent time slot and define the relationship between the end time of the last time slot and the total makespan or horizon duration (T_m).

Products are assigned to each slot using a set of binary variables, $z_{i,s} \in \left\{0,1\right\}$ along with constraints of the form

$$\sum_{s=1}^{n} z_{i,s} = 1 \qquad \forall i \tag{4.16}$$

$$\sum_{i=1}^{n} z_{i,s} = 1 \qquad \forall s \tag{4.17}$$

which ensure that only one product is made in each time slot.

The makespan is fixed to the length of the scheduling and control horizon rather than set as a manipulated variable adjustable by the NLP solver. Demand constraints are formulated such that production may not exceed the maximum demand for a given product, as follows:

$$\omega_i \leq \delta_i \qquad \forall i \tag{4.18}$$

The continuous-time scheduling optimization requires transition times between steady-state products ($\tau_{i'i}$) as well as transition times from the current state to each steady-state product if current state is not at steady-state product conditions ($\tau_{0'i}$). The transition times are estimated using two different methods: nonlinear model predictive control (NMPC) and feedback linearization. Transition times between steady-state products can be computed off-line and stored in memory; however, transition times from current state to steady-state products must be calculated on-line at each iteration of integrated scheduling and control optimization if implemented on-line. These two approaches to calculate transition times are discussed in the following subsections.

Nonlinear Model Predictive Control Transitions

NMPC transitions minimize an objective function of the form

$$\begin{aligned} \min_{u,t_f} \quad & J = (x(t_f) - x_{sp})^T W_{sp} (x(t_f) - x_{sp}) + t_f W_{time} \\ \text{s.t.} \quad & \text{nonlinear process model} \\ & x(t_0) = x_0 \end{aligned} \tag{4.19}$$

where $x(t_f)$ is the process state at final time, W_{sp} is the weight on the set point for meeting target product steady-state, t_f is the final time or time required for the transition, W_{time} is the weight on minimizing the final time, x_{sp} is the target product steady-state, and x_0 is the start process state from which the transition time is being estimated. This formulation harnesses knowledge of process dynamics in the system model to find an optimal trajectory and minimum time required to transition from an initial concentration to a desired concentration. The final time chosen by NMPC is taken as the estimate of transition times to use in the continuous-time scheduling optimization. This transition time is expected to be similar to the time taken by the discrete-time combined scheduling and control algorithm to transition between product steady-state conditions as both approaches harness a nonlinear system model to find optimal control profiles between products.

Feedback-Linearized Transitions

Although NMPC harnesses full knowledge of process dynamics as available in the system model and is expected to effectively imitate the transition durations of the combined scheduling and control algorithm presented in this work, NMPC is expected to scale poorly (in terms of computational burden) to larger systems with complex models and large numbers of products. The large computational requirement of NMPC may be unsuitable for initialization purposes in on-line implementations of integrated scheduling and control. Consequently, feedback linearization is presented as an alternative approach for estimating transition times in the continuous-time scheduling optimization.

A linear system $y = f(x,u)$ has the property that $f(x,u_0+u_1) = f(x,u_0)+f(x,u_1)$. Thus the response of the system to the initial input u_0 can be decoupled from that of the step size u_1. Additionally, a closed-form solution for the transition time given a step size can be estimated, thus avoiding preprocessing time and space.

Feedback linearization can be applied to dynamic systems of the form

$$\begin{aligned} \dot{x} &= f(x)+g(x)u \\ y &= h(x) \end{aligned} \tag{4.20}$$

Following the procedure outlined by Khalil *et al.* [219], the control signal u can be designed to make the input-output behavior of the system linear. For instance, in the SISO (Single Input Single Output) case

$$u = \frac{1}{L_g L_f^{\rho-1} h(x)} \left(-L_f^{\rho} h(x) + v \right) \tag{4.21}$$

where L_f represents the Lie derivative along f and ρ is the relative degree of the system. In this case, the input-output behavior of the closed-loop system is

$$v = y^{(\rho)} \tag{4.22}$$

which is a chain of ρ integrators.

An issue with feedback linearization is that the resulting closed-loop dynamics may not be stable, as seen in the SISO case above. This work proposes to use LQR control for the stabilizing controller, since full state feedback of $\hat{x}$ is available. This well-known approach finds a gain K that minimizes

$$J = \int_0^\infty x^T Q x + v^T R v \, dt \tag{4.23}$$

the control law then becomes $v = -K\hat{x}$ [220].

One drawback to using feedback linearization is that it can produce arbitrarily large input signals u in order to maintain linearity. In most cases, this leads to unreasonable state values x that are allowed by the mathematics of the model but are not realistic. For this reason, Q and R should be tuned in order to keep u within reasonable bounds for the relevant step sizes in the system.

Once the system has been tuned appropriately, then the transition times can be calculated independent of the starting point. This approach is similar to the scale-bridging model (SBM) as presented by Baldea *et al.* (2016) [22]. However, whereas Baldea *et al.* use linearization as a method for feeding information on process dynamics to a scheduler, this work uses feedback linearization strictly to estimate transition times for a continuous-time scheduling initialization for an overall nonlinear problem.

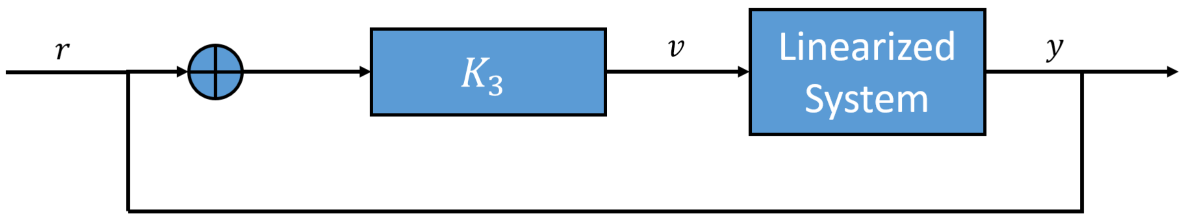


Figure 4.7: Stabilized linearized system block diagram.

The relevant difference between initial and target states is fed in as the reference signal s_{step} to calculate the transition durations. A function of the form

$$t_{trans} = \alpha \cdot log(s_{step}) + \beta \tag{4.24}$$

is fit to simulated transition times from a range of s_{step} to give a closed form and computationally light solution for finding the transition time in the estimated CSTR system. This feedback linearization method is expected to scale to larger systems with negligible computational requirements.

Continuous-Time to Discrete-Time Transpose

The solution to the continuous-time optimization provides a schedule that includes slot start times t_s^s and slot end times t_s^f as well as product assignment to each slot as determined by $z_{i,s}$. This continuous-time schedule determines the initialization of both the binary variable $B_{i,t}$ and the state variables x at each finite element. The initialization reduces the computational time required for the NLP solver at each iteration of combined scheduling and control. The initialization occurs according to the following algorithm:

for each finite element (fe),

$$\text{iff} \quad z_{i,s} = 1 \quad and \quad t_s^s < t_{fe} < t_s^f : \tag{4.25a}$$

$$\text{then} \quad B_{i,fe} = 1 \quad and \quad x_{fe} = x_i; \tag{4.25b}$$

$$\text{else} \quad B_{i,fe} = 0 \tag{4.25c}$$

For each time slot in the continuous-time schedule $[t_s^s, t_s^f]$, $B_{i,fe}$ transposes the product assignment $(z_{i,s})$ to finite elements within that time segment. x is initialized to the appropriate steady-state operating value (x_i) corresponding with the product manufactured during a given slot as given by $z_{i,s}$.

4.4 Case Study

This section presents the CSTR problem used to highlight the value of the formulation introduced in this work. While this system does not directly represent a specific industrial problem, the generic CSTR model is applicable in various industries from food/beverage to oil and gas and chemicals. This work details a general approach to combined scheduling and control and this model demonstrates the benefits on a generic system. Notable assumptions of a CSTR include: constant volume, well mixed and constant density.

The model shown in Eqs. 4.26 to 4.29 is an example of an exothermic, first-order reaction of $A \Rightarrow B$ where the reaction rate is defined by an Arrhenius expression and the reactor temperature is controlled by a cooling jacket. The fluid in the cooling jacket undergoes an external, arbitrary cooling process where ΔH_{cool} is the effective cooling rate.

$$\frac{dC_A}{dt} = \frac{q}{V}(C_{A0} - C_A) - k_0 e^{-E_A/RT} C_A \tag{4.26}$$

$$\frac{dT}{dt} = \frac{q}{V}(T_f - T) - \frac{1}{\rho C_p} k_0 e^{\frac{-E_A}{RT}} C_A \Delta H_r - \frac{UA}{V\rho C_p}(T - T_c) \tag{4.27}$$

$$\frac{dT_c}{dt} = \frac{q_{cool}}{V_j}(T_{cin} - T_c) + \frac{UA}{V_j \rho C_p (T - T_c)} \tag{4.28}$$

$$\Delta H_{cool} = \rho C_{p.cool} q_{cool} (T_c - T_{cin}) \tag{4.29}$$

In these equations, C_A is the concentration of reactant A, C_{A0} is the feed concentration, q is the inlet and outlet volumetric flowrate, V is the tank volume (q/V signifies the residence time), E_A is the reaction activation energy, R is the universal gas constant, UA is an overall heat transfer coefficient times the tank surface area, ρ is the fluid density, C_p is the fluid heat capacity, k_0 is the rate constant, T_f is the temperature of the feed stream, C_{A0} is the inlet concentration of reactant A, ΔH_r is the heat of reaction, q_{cool} is the flowrate of coolant, V_j is the volume of the cooling jacket, T is the temperature of reactor, T_c is the temperature of cooling jacket, T_{cin} is the temperature of cooling return line and $C_{p.cool}$ is the cooling fluid heat capacity. Table 4.1 lists the CSTR parameters used.

In this example, one reactor can make multiple products by varying the concentrations of A and B in the outlet stream. The manipulated variables in this optimization are ΔH_{cool} and

Table 4.1: Reactor Parameter Values

Parameter	Value
V	$400m^3$
q_{cool}/V_{jacket}	$5hr^{-1}$
E_A/R	$8750K$
$\frac{UA}{V\rho C_p}$	$0.523hr^{-1}$
k_0	$1.8e10hr^{-1}$
T_f	$350K$
C_{A0}	$1mol/L$
$\frac{\Delta H_r}{\rho C_p}$	$-209\frac{Km^3}{mol}$

q. q is bounded by $100m^3/hr \leq q \leq 120m^3/hr$ and ΔH_{cool} is either bounded by $4MW$ or a diurnal maximum cooling curve. The sample problem uses three products over a 48-hour horizon. The product descriptions are shown in Table 4.2, where the product specification tolerance is $\pm 0.005\,mol/L$.

Table 4.2: Product Specifications

Product	C_A	Max Demand	Price
	(mol/L)	(m^3)	(\$/10 m^3)
1	0.35	1920	24
2	0.12	2880	27
3	0.25	2880	21

The only scheduling constraint used in this case study is demand, as shown in Equation 4.30. While these results use maximum demand (useful for situations like filling storage tanks rather than filling orders), it can easily switch to minimum demand by flipping the inequality.

$$\int_0^t B_{i,t} \leq maxdemand_i, \quad \forall i \tag{4.30}$$

The pseudo-binary variable approach is implemented via the following equations, with C_A being the process state variable relating to each product i:

$$f_i(C_A) = h10^{log(1/h)/tol^2(C_{A,i}-C_A)^2} \quad \forall i \tag{4.31}$$

$$B_i(C_A) \leq f_i(C_A), \quad B_i \in [0,1] \quad \forall i \tag{4.32}$$

For continuous-time scheduling initialization, NMPC estimations of transition times are calculated using the following objective:

$$\begin{aligned} \min_{\Delta H, q, t_f} \quad & J = (C_A(t_f) - C_{A,sp})^T W_{sp} (C_A(t_f) - C_{A,sp}) + t_f W_{time} \\ \text{s.t.} \quad & \text{Eq.}\, 4.26 - 4.29 \end{aligned} \tag{4.33}$$

while feedback linearization estimates transition times with the following closed-form equation fit to feedback linearized process simulations:

$$t_{trans} = 0.9853 \cdot log(s_{step}) + 5.332 \tag{4.34}$$

For a detailed derivation of Equation 4.34, the reader is directed to Appendix A.

Four test cases were considered to develop the integration of time-based parameters:

1. Static pricing and cooling constraints
2. Static pricing, diurnal cooling constraint function
3. Static cooling constraint, diurnal pricing function
4. Diurnal pricing and cooling constraint functions

Case 1 is the standard case with time-independent constraints that should largely replicate the results of a continuous-time, slot-based scheduling formulation. Results from cases 2 and 3 are summarized since their combined effects reappear in case 4.

The dynamic diurnal cycles of energy price and effective cooling constraints are generalized by simple sinusoidal curves, as shown in Figure 4.8. The energy price varies from \$10 per MWh during the day to \$90 per MWh during the night, with the static price representing the average of \$50 per MWh. The effective cooling constraint represents the amount of cooling done that affects the system; in other words, the cooling done minus losses to the environment, etc. Therefore, the higher ambient temperature during the day reduces effective cooling to the reactor because of heat loss to the environment, while more cooling is possible during the colder night. This consideration allows the system to account for demand response in the optimization, leveraging the abilities to account for dynamic constraints and parameters in the discrete-time dynamic optimization problem.

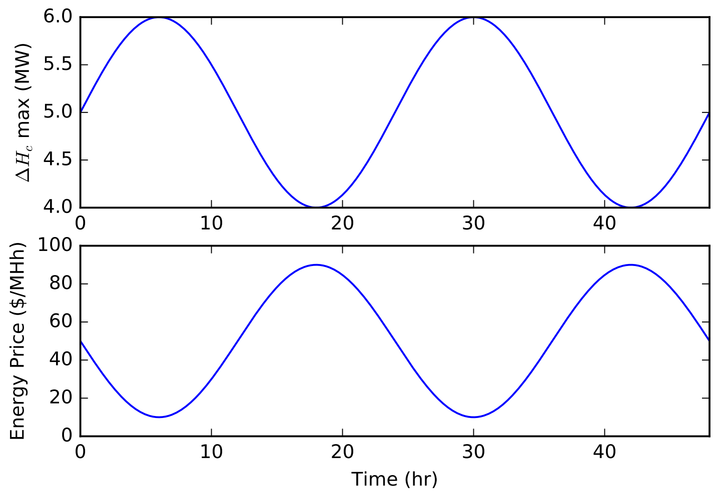


Figure 4.8: Plots of maximum effective cooling constraint and time-of-day pricing over 48 hours.

The objective function is formulated as follows:

$$\begin{aligned} \text{maximize} \quad & \sum_{t}^{n}\sum_{i}^{n}(q_t\Pi_i B_{i,t}) - E_t \\ \text{s.t.} \quad & \text{Process Model} \quad (\text{Eqs. 4.26 - 4.29}) \\ & \text{Scheduling constraints} \quad (\text{Eq 4.30}) \\ & \text{Pseudo-binary Eqs.} \quad (\text{Eqs. 4.31 - 4.32}) \end{aligned} \tag{4.35}$$

4.4.1 Closed-loop Control

Since this method uses the full process model of MPC with sufficiently fine time discretization, it can be used in closed-loop control. Once an adequate solution is reached using an initialization method above, the previous solution horizon provides the initialization for the next control move calculation. On rare occasions of sufficiently large disturbances, the previous solution may not be adequate for initialization and one of the above initialization strategies may be used once again.

This work solves a simple closed-loop sample case over 24 of the 48 hour horizon. Further analysis of this method's closed-loop strengths and challenges, including responses to more drastic disturbances of both control (e.g. sudden change in concentration) or scheduling (e.g. change in demand) parameters is a subject of future work.

The closed-loop implementation uses the same problem formulation as Equation 4.4. The selected linking function is Equation 4.7, which is generally the fastest method. The individual product demands are updated on each iteration based on real-time production numbers.

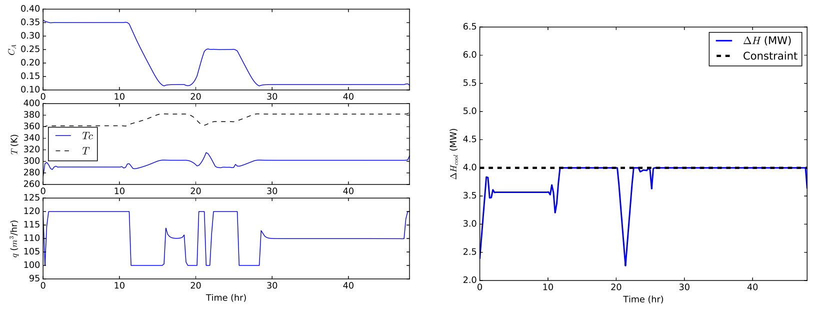
4.5 Results

The results of each of the four test cases are described below. The description for each case includes comparisons of the effects of strategies employed to reduce the computational requirements of the problem. Each case has 2 plots for each initialization scheme employed. The first plot shows the system state variables, and the second plot shows the maximum cooling constraint with the actual system cooling. In cases 2 and 4, the energy price curve is overlaid onto the plots, with the right axis showing price units. The description of each case also contains tables detailing the computational requirements and economic results for convergent cases.

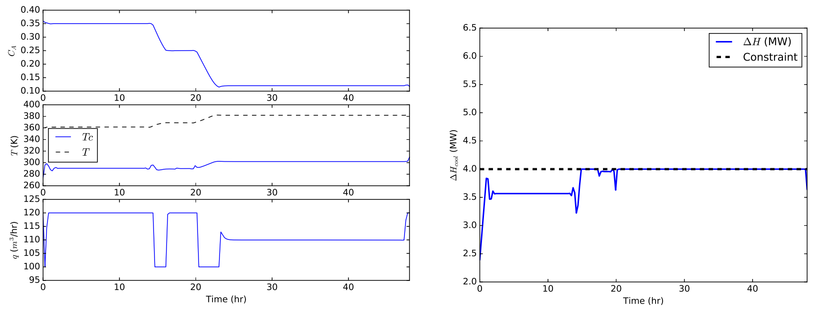
Although the relaxed psuedo-binary variables are capable of yielding non-integer values, they almost always come very close to integer values. There are occasional non-integer values, especially during transitions, but their overall effect is minor compared to the magnitude of the problem, the uncertainty of a 48 hour schedule and plant-model mismatch. To remove the effects of intermediate values, the results are post-processed and the reported profits include only on-spec production.

Each problem is solved on an Intel i7 CPU-6700 at 3.40 GHz. The continuous-time scheduling problems are solved using the COUENNE branch-and-bound MINLP solver [178], the NMPC transition time estimations are solved with the APOPT MINLP solver [177], and the discrete-time integrated problems are solved with the IPOPT NLP solver [217]. The discrete-time problems are discretized over 200 finite elements with one collocation point within each finite element. Each problem is formulated using the Pyomo modeling language [58, 175]. Pyomo is designed for ultimate flexibility rather than solution speed [59]. The flexibility is useful for exploring these initialization strategies, but Pyomo is not recommended for time-sensitive solutions. Thus, the case study is replicated in GEKKO to compare solution speeds.

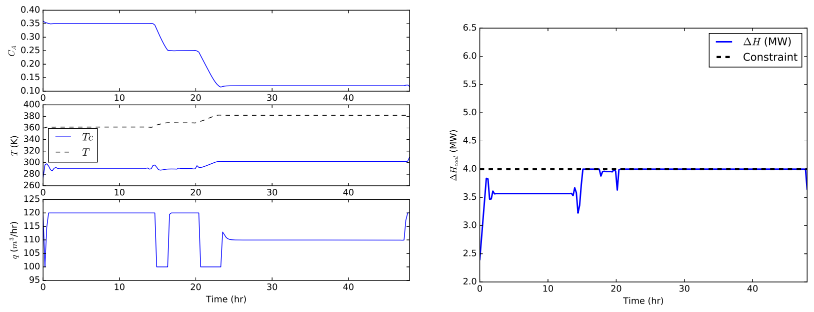
4.5.1 Case 1: Static Pricing and Cooling Constraints



(a) No initialization employed, reached max iterations.



(b) Initialized by continuous-time scheduling with linear-estimated transition times.



(c) Initialized by continuous-time scheduling with NMPC-estimated transition times.

Figure 4.9: Case 1 results.

As shown in Figure 4.9, case 1, the standard case with time-independent constraints of static price and static cooling, maximizes the production of product 2 for all initialization schemes employed because of its high price. Production of product 3 is minimized due to its low price.

Table 4.3: Computational Requirements: Case 1

Initialization Scheme	Initialization CPU time (s)	Discrete Problem CPU time (s)	Total CPU time (s)
None	0	$> 10,000$	$> 10,000$
Continuous-time (linear)	0.18	619	619
Continuous-time (NMPC)	0.60	921	922

Table 4.4: Economic Summary: Case 1

Initialization	Product Production (m^3)			Profit
Scheme	1	2	3	(\$)
Continuous-time (linear)	1752	2866	559	3538
Continuous-time (NMPC)	1780	2839	559	3541

Product 1 sells at an intermediate price and is therefore produced during the remainder of the fixed horizon duration. The production order is selected to minimize transition times by stepping down to products with incrementally lower concentrations.

The benefits of initialization are demonstrated by the orders of magnitude in computational time reduced by applying continuous-time initialization to the problem. Effective continuous-time scheduling initialization guides the discrete-time problem to find the optimal solution, whereas the non-initialized problem fails to converge within reasonable time. The non-optimal result of the non-initialized problem after maximum iterations is shown in Figure 4.9(a) for comparison. The problem reaches a local minimum, producing a large amount of each product but creating a sub-optimal schedule with more transitions than necessary.

As expected, the feedback linearization estimations of transition times provide a reduction in initialization CPU time by roughly 60%, compared to the NMPC method. However, the CPU time required for initialization is negligible with respect to the overall problem. The economic results of the variations in continuous-time scheduling initializations vary only negligibly; however, continuous-time scheduling initialization with feedback linearization requires the least computational time.

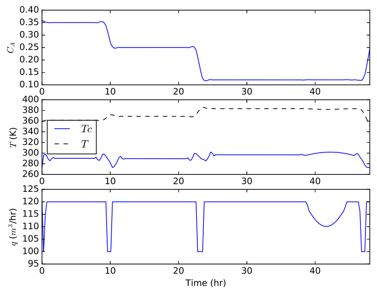
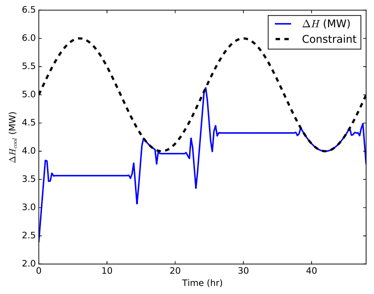



Figure 4.10: Case 2: Initialized by continuous-time scheduling with NMPC-estimated transition times.

Table 4.5: Economic Summary: Case 2

Initialization	Product Production (m^3)			Profit
Scheme	1	2	3	(\$)
Continuous-time (NMPC)	1777	2851	909	4010

4.5.2 Case 2: Static Pricing, Diurnal Cooling Constraint Function

The diurnal cooling constraint curve applied in case 2 allows product 2 to be produced at a higher rate compared to case 1. The production rate is decreased during the hottest part of the day, but reaches the lower production rate of case 1 for only a brief period. Further, the transitions between products occur more quickly when the max cooling constraint is higher because of the extra cooling (especially the transition between products 3, C_A=0.25, and 2, C_A=0.12).

The overall profit for Case 2 increases ˜13% over case 1 for convergent continuous-time initialization. This shows the value of considering time-dependent constraints in combined scheduling and control and further justifies a discrete-time formulation because of the ease with which it can apply these constraints. Continuous-time formulations require steady-state conditions during production slots, which eliminates the possibility of considering time-dependent parameters as demonstrated in this work.

CPU time requirements for convergent continuous-time initialized problems decrease by over 50% compared to case 1. This demonstrates the extra effort required by the optimization algorithms to find an optimal solution while meeting the restrictive, fixed cooling constraint in case 1.

4.5.3 Case 3: Static Cooling Constraint, Diurnal Pricing Function

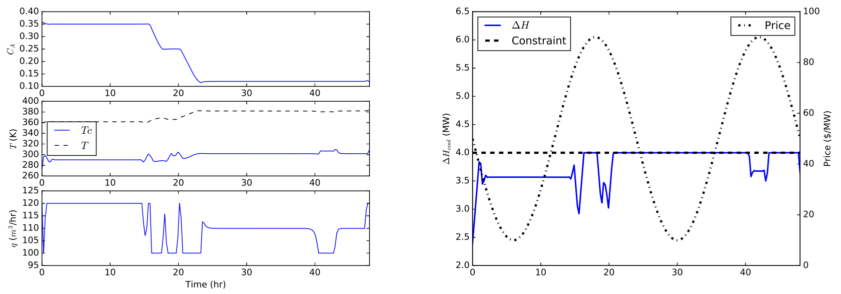


Figure 4.11: Case 3: Initialized by continuous-time scheduling with NMPC-estimated transition times.

Table 4.6: Economic Summary: Case 3

Initialization Scheme	Product Production (m^3) 1	2	3	Profit (\$)
Continuous-time (linear)	1754	2836	489	3400
Continuous-time (NMPC)	1919	2810	370	3482

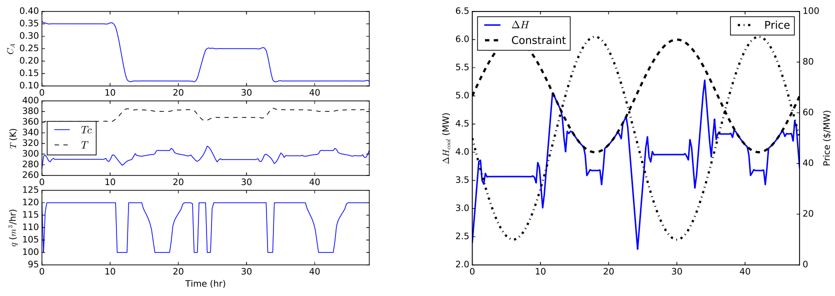
Case 3 largely follows case 1, except that production rates decrease when energy prices peak. Energy costs too much during these times, so the optimization minimizes production rate (q) to the lower bound of 100 m^3/hr. Also, transitions between products occur at slightly different times to compensate for different production rates and to transition during times of cheaper energy.

The profit in this case for the continuous-time initialized problems decreased slightly (˜3%) from case 1 due to high energy prices, since this case considers realistic dynamic pricing. Again, time-dependent parameters are shown to be worth considering.

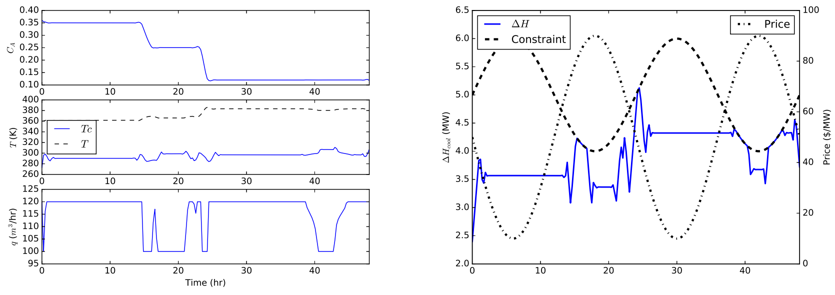
Effective continuous-time scheduling initialization once again guides the discrete-time problem to find the optimal solution, whereas the non-initialized problem ends at a local minimum, producing a large amount of each product but creating a sub-optimal schedule with more transitions than necessary. CPU time requirements for initialized problems increase by approximately

40% compared to Case 1, demonstrating the extra effort exerted by the optimization algorithm to find the optimal solution with dynamic pricing parameters.

4.5.4 Case 4: Diurnal Pricing and Cooling Constraint Functions



(a) No initialization employed.



(b) Initialized by continuous-time scheduling with NMPC-estimated transition times.

Figure 4.12: Case 4 results.

Table 4.7: Computational Requirements: Case 4

Initialization Scheme	Initialization CPU time (s)	Discrete Problem CPU time (s)	Total CPU time (s)
None	0	$> 10,000$	$> 10,000$
Continuous-time (linear)	0.18	$> 10,000$	$> 10,000$
Continuous-time (NMPC)	0.60	571	572

Case 4 implements the positive effects of case 2 as well as the peak energy prices of case 3 — the transitions occur at different places, production rate of product 2 is maximized and production at peak energy prices is decreased (Figure 4.12). Transitions occur more quickly during

Table 4.8: Economic Summary: Case 4

Initialization Scheme	Product Production (m^3) 1	2	3	Profit ($)
Continuous-time (NMPC)	1778	2820	816	3863

periods of higher maximum cooling. The overall profit increases by approximately 10% over the base case (Case 1) for the convergent initialized problem, but is still lower than Case 2 due to the effects of peak energy prices (Table 4.8).

The CPU requirement of the convergent initialized problem are similar to that of Case 2, but lower than those of Cases 1 and 3 (Table 4.7). This demonstrates the additional effort required by the optimization algorithm to find an optimal solution with a restrictive, fixed constraint versus a dynamic constraint. The benefits of continuous-time scheduling initialization are again reiterated by the orders of magnitude reduced in computational time required and by the guidance to the optimal solution rather than a local minimum. As in Case 2, the continuous-time scheduling initialization with NMPC estimations converges whereas the linearized initialization fails to converge.

4.5.5 GEKKO Solutions

As previously mentioned, Pyomo is designed for flexibility rather than speed. This section reimplements the SC problem in the GEKKO modeling language [75], which specializes in robust, quick solutions to dynamic optimization problems. Pyomo and GEKKO have some structural differences, especially in the way each handles orthogonal collocation. To replicate the same degrees of freedom used in Pyomo, GEKKO solutions use 400 finite elements with no internal nodes.

All 12 GEKKO solutions converged in under 10,000 seconds and all profit results are within 4% difference of the converged Pyomo results reported. The time results from GEKKO are shown in Table 4.9 and the profit results are shown in Table 4.10.

For the six cases in which Pyomo did not reach a solution in under 10,000 seconds, GEKKO reached a solution in an average time of 214 seconds. In the 6 cases that both Pyomo and GEKKO

Table 4.9: Computational Requirements: GEKKO

Initialization Scheme	Case 1 CPU time (s)	Case 2 CPU time (s)	Case 3 CPU time (s)	Case 4 CPU time (s)
None	193	187	278	362
Continuous-time (linear)	137	111	191	152
Continuous-time (NMPC)	129	103	187	155

reached a solution, Pyomo took an average of 793 seconds while GEKKO took an average of 150 seconds – about 5 times faster.

Using GEKKO, the uninitialized problems reached local minima that were worse than the solutions from initialized cases, confirming the value of the initialization techniques. Unlike in Pyomo, linear and nonlinear initializations achieved the same solution but the nonlinear initialization proved slightly faster on average.

Table 4.10: Profit Results: GEKKO

Initialization Scheme	Case 1 Profit	Case 2 Profit	Case 3 Profit	Case 4 Profit
None	3598	3835	2942	3575
Continuous-time	3618	3975	3488	3995

4.5.6 Closed-loop Simulation

The previous cases demonstrate this method's ability to provide a detailed schedule, including considering time-dependent parameters such as energy cost and effective cooling constraints. However, this method is also capable of closed-loop control without modification because it utilizes a full dynamic process model and can begin with any initial conditions. In this capacity, this formulation can overcome process disturbances of short-time scales with economic consideration of multiple products.

The previous results used Equation 4.9 for flexibility and the best solution. This section uses Equation 4.7 and GEKKO for speed in online control. Under these new conditions, and with the highly detailed solution of the previous time-step as initialization, each closed-loop solution at every 5 minute interval solved fast enough for real-time. A summary of solution times is provided in Table 4.11.

Table 4.11: Closed-Loop Time Summary

Horizon (hr)	Control Move	Total Solver Time (hr)	Solutions	Mean Solution Time (s)
24	5 min	8.39	288	105

4.5.7 Summarized Results

Table 4.12: Initialization Comparison

Initialization Scheme	Profit ($) Pyomo	Profit ($) GEKKO	CPU time (s) Pyomo	CPU time (s) GEKKO
None	NA	3487	$> 10,000$	255
Continuous-time (linear)	3469	3769	586 (2 converged)	148
Continuous-time (NMPC)	3724	3769	897 (4 converged)	144

Table 4.13: Case Comparison

Case	Profit ($)	Energy Pricing	Cooling Constraint
1	3541	Static	Static
2	4010	Static	Dynamic
3	3482	Dynamic	Static
4	3863	Dynamic	Dynamic

Table 4.12 shows profit and CPU requirements of each initialization scheme, averaged across all convergent cases, to make an overall comparison between initialization schemes. Table 4.13 displays the profit for continuous-time scheduling initialization with NMPC estimated transition times (the only initialization scheme convergent for all cases) to demonstrate the effects of diurnal constraints and parameters on combined scheduling and control optimization.

In summary, time-dependent constraints affect the profit, optimal schedule, and optimal control or operation of a chemical process. These considerations can have a significant economic impact, with diurnal constraints increasing profits ˜13% from the base case in this example. It is anticipated that, under the right circumstances, the scheduler may go so far as to switch products in response to these diurnal cycles, forcing extra transitions that would not be possible in current implementations of slot-based combined scheduling and control formulations, where the number of slots frequently equals the number of products. In other cases, the scheduler may order products differently with time-based constraints in consideration. Further, this method is easily applied to other time-dependent parameters beyond diurnal cycles, such as feed stock price predictions. These explorations are the subject of future work.

The discrete-time formulation is shown to be a feasible and effective method to account for time-dependent parameters and constraints in combined scheduling and control. The positive effects of continuous-time scheduling initialization have been demonstrated. Convergent continuous-time scheduling initialization decreases computational requirements on average by approximately 15 times. The method for estimating transition times in the continuous-time scheduling initialization is found to be significant in determining the convergence of the discrete-time SC problem. NMPC estimations are found to be more consistent for initializing the nonlinear discrete-time formulation and are found to guide the solution to more optimal solutions. Even with the effective NMPC initialization, the full MINLP did not solve successfully in under 10,000 seconds using the APOPT or Bonmin solvers.

4.6 Conclusion

This work applied a nonlinear discrete-time formulation for combined scheduling and control. This method provided a schedule of sequential products using the full model dynamics through the entire horizon. The discrete-time formulation easily allowed the implementation of time-dependent parameters and constraints. This work applied time-dependent parameters of diurnal cycles of energy price and maximum effective cooling of a CSTR. This optimization improved open-loop scheduling profit prediction by 13% over the base scenario. This work implemented a pseudo-binary approach to assist gradient-descent solvers in finding the optimal solution to an inherently mixed-integer problem. This work also leveraged continuous-time scheduling with different methods to estimate transition times to calculate an optimal schedule order and schedule timing to initialize the discrete-time problem. Continuous-time scheduling with nonlinear estimations of transition times consistently decreased the computational requirements of the nonlinear discrete-time problem by many orders of magnitude.

CHAPTER 5. CONCLUSION

This dissertation explored novel approaches in the field on large-scale non-linear dynamic optimization, especially applied to scheduling and control. Optimal scheduling and control applications have historically been separate problems with little coordination. The current scheme presents inefficiencies in manufacturing of continuous chemical processes. After understanding the approaches, benefits, challenges and pitfalls of current scheduling and control practices, this dissertation investigated the combination of these applications to a single problem to achieve more optimal production than the two applications could achieve separately.

5.1 Economic Benefits of Combining Scheduling and Control

Chapter 2 demonstrated the economic justification for combining scheduling and control. It explored four levels of successively tighter integration: open-loop segregated scheduling and control, closed-loop segregated scheduling and control, open-loop scheduling with consideration of process dynamics, and closed-loop integrated scheduling and control responsive to process disturbances and market fluctuations. Each level was applied in three scenarios of unexpected disturbances. It was demonstrated on a CSTR benchmark application in closed-loop simulation over 24 hours that tighter integrations of the two approaches allowed for increased capacity to overcome disturbances in both process manufacturing and the product markets. This work validates the field of combining scheduling and control. Further, a simple, novel approach for combining scheduling and control was implemented as the highest level of integration. A fixed horizon integrated scheduling and control formulation for multi-product, continuous chemical processes is utilized, in which nonlinear model predictive control (NMPC) and continuous-time scheduling are combined. This approach decomposed the MINLP problem to coordinated NLP and MILP optimizations.

5.2 Novel Scheduling and Control Paradigm

Chapters 3 and 4 presented approaches to fully unify scheduling and control for continuous, multi-product manufacturing.

In Chapter 3, linear model predictive control is extended to both provide closed-loop control and optimize a product grade schedule. The proposed methods are time scaling of the linear dynamics based on throughput rates and grade-based objectives for product scheduling based on a mathematical program with complementarity constraints. The linear model is adjusted with a residence time approximation to time-scale the dynamics based on throughput. Although nonlinear models directly account for changing dynamics, the model form is restricted to linear differential equations to enable fast online cycle times for large-scale and real-time systems. This method of extending a linear time-invariant model for scheduling is designed for many advanced control applications that currently use linear models. Simultaneous product switching and grade target management is demonstrated on a reactor benchmark application. The objective is a continuous form of discrete ranges for product targets and economic terms that maximize overall profitability.

Chapter 4 investigates combining scheduling and control using a nonlinear discrete-time formulation, utilizing the full nonlinear process model throughout the entire horizon. This discrete-time form lends itself to optimization with time-dependent constraints and costs. The discrete-time approach to combined scheduling and control is explained, along with sample pseudo-binary variable functions to ease the computational burden of this approach. An initialization strategy using feedback linearization, nonlinear model predictive control, and continuous-time scheduling optimization is presented. The formulation is applied with a generic CSTR system in open-loop simulations over a 48-hour horizon and a sample closed-loop implementation. The value of time-based parameters is demonstrated by applying cooling constraints and dynamic energy costs of a sample diurnal cycle, enabling demand response via combined scheduling and control.

5.2.1 Novel Contributions

In context of other work in the field, the work presented in this dissertation stands out in the following ways:

- The economic benefit of combining scheduling and control is demonstrated quantitatively compared to current practice
- Time is represented with the “discrete-time” paradigm
- Solutions present both a long-term schedule featuring multiple products and process dynamics, and short-term control moves for closed-loop implementation
- The paradigm can be easily simplified for a reliable replacement of linear MPC, or expanded to include any constraints or objectives
- The discrete-time approach enables consideration of time-based parameters and constraints, which are shown to have significant impact in economic optimality

5.3 Future Work

While this and other dissertations have addressed the value of combining scheduling and control, and have presented various approaches, much work is still required before such paradigms become common in industrial applications.

Chapter 2 demonstrated the benefit of ISC on a CSTR case study with three scenarios. The development of additional benchmark problems applicable to a wider variety of industrial scenarios is proposed as an important potential subject of future work. With increasing research in ISC, benchmark problems for formulation performance comparison of integrated scheduling and control formulations, as well as for comparison against a baseline segregated scheduling and control formulation, are increasingly important. Benchmark applications and scenarios applicable to batch processes, multi-product continuous processes, and other processes with scenarios representative of probable industrial occurrences should be developed.

The work presented in this dissertation is applicable to continuous processes considering a single process unit. Progressive integrations proving economic benefit of scheduling and control integration should also be applied to batch processes and continuous processes considering multiple process units.

This work motivates continued investigation into discrete-time formulations and time-dependent parameters in considering both transitions and product manufacturing. In particular, the pseudo-binary approach presented in Chapter 4 should be implemented as part of an interior point solver. Additionally, accounting for product inventory, the specifics of closed-loop implementation, and accounting for process and market uncertainties should be addressed in future work as this method matures.

5.3.1 Solver Development

Advances to algorithms, such as the large-scale non-linear programming solvers presented in the Introduction and used throughout this work, enabled the possibility of combining scheduling and control. However, further improvements in these solvers would greatly facilitate more reliable solutions to more complex applications. This section presents some ideas for solver development.

Combining Interior Point and Active Set Approaches

The benefits of combining interior-point (IP) and active set (AS) have been theorized [221]. Figure 5.1 shows benchmark performance of various solvers, including the theoretical performance of a combined active set (APOPT) and interior point (BPOPT) solver, where the faster solution of each method is accepted. Some solvers tap these benefits by switching between methods at predetermined conditions [107]. Further combining interior-point and active-set methods will unify the benefits of each method: robust early steps of poorly initialized problems with an accelerated convergence near the solution. Figure 5.1 shows the combination of AS and IP having potential to solve approximately 30% of problems unable to be solved through either method alone.

Interior-point and active-set methods are the two leading approaches to NLP with inequality constraints [222]. Each method has its strengths and weaknesses. Interior point methods tend to do better with larger, poorly initialized problems while active set methods are faster when excellent initialization is available [95, 223].

Active set methods have the benefit of smaller matrix factorizations when inactive constraints are ignored. Further, they encourage the solver to meet the active inequality bounds directly, when optimal. The biggest weakness of active set methods is the accurate selection of the active set. Each iteration of an AS solver can include multiple minor iterations as it seeks the active set. Interior point methods provide a barrier term on inequality constraints that maintain feasibility.

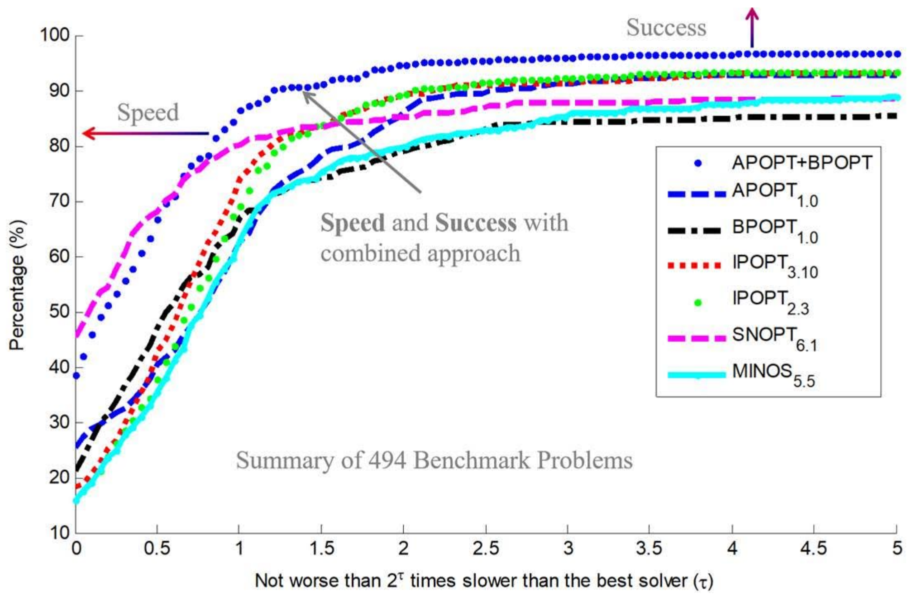


Figure 5.1: Solver comparison on benchmark set showing benefit of combining IP (BPOPT) and AS (APOPT) methods [1].

However, these barriers can slow the approach to active inequality bounds. Further, all constraints are considered on each iteration, leading to larger matrices in the Newton step.

Future work should seek to combine the benefits of AS (smaller matrices, faster approach to inequality bounds) with the benefits of IP (better initialization, no time wasted in switching the active set).

One approach to combining the methods includes dividing the inequality constraints into "active", "inactive" and "semi-active". Active and inactive constraints will be treated in the standard AS approach, while "semi-active" constraints will include a barrier term, like inequality constraints in the IP method. These semi-active constraints would traditionally be inactive in the AS method and would therefore be ignored. By ignoring them, the Newton step is liable to skip over them – resulting in a change in the active set and an additional minor iteration. By providing the barrier term, the solver is aware of the constraint, and encouraged not to violate it, but is not forced to treat it as an equality constraints (like typical "active" constraints).

This combined method requires a new objective function:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \Phi(x) := f(x) - \mu \sum_i D_i ln(x_i) \\ \text{subject to} \quad & c(x) = 0 \\ & x \geq 0 \end{aligned} \tag{5.1}$$

where D is the set of semi-active constraints ($D_i = 1$ if constraint i is semi-active, $D_i = 0$ otherwise).

The Newton step of the combined method incorporates barriers on semi-active constraints, ignores inactive constraints and calculates only the semi-active bound multipliers implicitly, as follows:

$$\begin{bmatrix} H_k + \Sigma_{D_k} & A_k & -E \\ A_k^T & 0 & 0 \\ -E^T & 0 & 0 \end{bmatrix} \begin{pmatrix} d_k^x \\ d_k^\lambda \\ d_k^{z_A} \end{pmatrix} = - \begin{pmatrix} \nabla \Phi_{\mu_j}(x_k) + A_k \lambda_k \\ c_{AD}(x_k) \\ E^T(-s_k) \end{pmatrix} \tag{5.2}$$

$$d^{z_{D_k}} = \mu_j X_k^{-1} e - z_{D_k} - \Sigma_{D_k} d^{z_{D_k}} \tag{5.3}$$

Where c_{AD} is the set of active and semi-active constraints, $A_k := \nabla c_{AD}(x_k)$, z_A and z_D are the variable bounds for active and semi-active bounds (respectively), and $\Sigma_D := X_k^{-1} Z_{D_k}$.

The set of semi-active constraints are always in the region of x_k that are not already active. Determination of the semi-active set will likely be a function of d_{k-1}^x and/or Ψ. The goal is to predict all the inequality constraints that have significant affect on the next Newton step, without including too many. Over-inclusion of semi-active constraints makes the matrix factorization too large, thus losing some the the AS benefit. However, initializing with $d_0^x = \infty$ basically yields IP iterations at the beginning, where the IP method is stronger.

In summary, at poor initialization the solver will approximate the IP method for robustness, but as the solver iterates it will approximate the AS method for convergence efficiency. In the middle, the barrier term on the constraints will reduce the number of minor iterations required to update the active set and the size of the matrix factorization is reduced.

Exploration of this method includes accurately determining the active, semi-active and inactive constraint split, how typical IP aspects (eg barrier reduction) are addressed, compatibility with a filter method [104], and testing on applicable problems.

Higher Order Methods

The core of every nonlinear optimization iteration is determining the step size and direction. To find optimal step size and direction, the Newton step (or a quasi-newton method) is the most commonly used method for gradient-based optimization [222]. While good (providing quadratic convergence near the optimal [127]), it only quadratically approximates the objective function and linearly approximates the constraints [222]. With exponentials and trigonometric functions in the constraints, linear approximations are insufficient.

For example, in root finding of the Equation 5.4 , the root is at $x = 0$. At initial conditions of $x_0 = -5$, Newton's method requires almost 30 times the iterations of Halley's method, as shown in Table 5.1. This is because Newton's method ignores the strong nonlinearity and far overshoots the root, then steps slowly back down. Halley's method detects the nonlinearity and steps more conservatively without overshooting.

$$f(x) = e^x - 1 \tag{5.4}$$

Table 5.1: 30x Improvement in Iteration Count with Halley's Method

	Newton's Method	Halley's Method
Iterations	148	5

The Halley step (a higher-order Householder method), gives a cubic approximation of the objective function and quadratic approximation of the constraints. While a second-order correction (e.g. the Shamanskii method [224]) approximates non-linear constraints and objective functions more accurately than the Newton step, the Halley step can also be expanded with the Shamanskii method to achieve quintic convergence as well as an even better approximation of non-linear constraints and non-linear objective functions than the independent Halley step [125]. The further away from the solution, the worse Newton's method approximates non-linear constraints. In addition, if globalization strategies are utilized, the Maratos effect can occur in areas close to the solution.

Literature for unconstrained optimization has shown faster convergence of higher order methods over Newton's method in areas close to the solution with a globalization strategy implemented. [127]. A newton-step correction similar to a second order correction was extended from smooth convex applications to non-convex nonlinear optimization in [128]. The extension of the newton-step achieved optimal convergence for convex problems and improved the best rate of convergence for non-convex problems. Another iterative method called the predictor-corrector method was applied to Halley's method in [129] to improve convergence rate.

Many NLP algorithms use a Second Order Correction (SOC) to improve the proposed step. In a common SOC, the Shamanskii method, a_k is the proposed Newton Step calculated by Equation 5.5 and s_k is the SOC correction from Equation 5.6, so the new step is $a_k + s_k$.

$$f'(x_k)a_k = -f(x_k) \tag{5.5}$$

$$f'(x_k)s_k = -f(x_k + a_k) \tag{5.6}$$

This method is commonly used because it uses the same matrix factorization for $f'(x)$. This is beneficial since the matrix factorization is the most expensive part of the linear algebra.

Cuyt and Rall [225] present a similarly efficient numerical implementation for the multivariate Halley step. Rather than a SOC s_k, this method calculates a Halley correction b_k in Equation 5.7 and the proposed step becomes $x_k + c_k$, where c_k is calculated in Equation 5.8.

$$f'(x_k)b_k = f''(x_k)a_k a_k \tag{5.7}$$

$$c_k = \frac{a_k^2}{a_k + \frac{1}{2}b_k} \tag{5.8}$$

In Equations 5.7-5.8, multiplication and division of the vectors is component-wise and the tensor multiplication is

$$Bx = \sum_{k=1}^{n} b_{ijk} x_k$$

for a vector $x \in R^n$ and $B \in R^{n \times n \times n}$.

This implementation also uses the same matrix factorization for $f'(x)$. Similar to SOC criteria, this implementation does not necessitate calculation of the Halley correction if the Newton step is sufficient (ie QP).

One drawback to higher order methods is the expensive of calculating higher-order derivatives. A high order convergent iterative scheme not requiring second derivative information by using numerical methods has been proposed in [130, 131]. Alternatively, advancements to AMLs could facilitate higher-order derivatives.

Bibliography

[1] Hedengren, J., Mojica, J., Cole, W., and Edgar, T., 2012. "APOPT: MINLP solver for differential and algebraic systems with benchmark testing." In *INFORMS National Meeting*. viii, 115

[2] Biegler, L. T., and Zavala, V. M., 2009. "Large-scale nonlinear programming using IPOPT: An integrating framework for enterprise-wide dynamic optimization." *Computers & Chemical Engineering,* **33**(3), pp. 575–582. 1, 14, 78

[3] Pontes, K. V., Wolf, I. J., Embiruçu, M., and Marquardt, W., 2015. "Dynamic Real-Time Optimization of Industrial Polymerization Processes with Fast Dynamics." *Industrial & Engineering Chemistry Research,* **54**(47), pp. 11881–11893. 1, 56, 78

[4] Feng, J. Y., Brown, A., O'Brien, D., and Chmielewski, D. J., 2015. "Smart grid coordination of a chemical processing plant." *Chemical Engineering Science,* **136**, pp. 168–176. 1, 57, 80

[5] Tedford, N. P., and Martins, J. R. R. A., 2010. "Benchmarking multidisciplinary design optimization algorithms." *Optimization and Engineering,* **11**(1), Feb, pp. 159–183. 1

[6] Zhang, Q., and Grossmann, I. E., 2016. "Enterprise-wide optimization for industrial demand side management: Fundamentals, advances, and perspectives." *Chemical Engineering Research and Design,* **116**(Supplement C), pp. 114 – 131 Process Systems Engineering - A Celebration in Professor Roger Sargent's 90th Year. 1

[7] Washington, I., and Swartz, C., 2015. "Multi-Period Dynamic Optimization for Large-Scale Differential-Algebraic Process Models under Uncertainty." *Processes,* **3**(3), pp. 541–567. 1

[8] Nyström, R. H., Franke, R., Harjunkoski, I., and Kroll, A., 2005. "Production campaign planning including grade transition sequencing and dynamic optimization." *Computers & Chemical Engineering,* **29**(10), pp. 2163–2179. 1, 33, 41

[9] Touretzky, C. R., and Baldea, M., 2014. "Integrating scheduling and control for economic

MPC of buildings with energy storage." *Journal of Process Control,* **24**(8), pp. 1292–1300. 1, 36

[10] Powell, K. M., Cole, W. J., Ekarika, U. F., and Edgar, T. F., 2013. "Dynamic optimization of a campus cooling system with thermal storage." In *2013 European Control Conference (ECC)*, pp. 4077–4082. 1

[11] Huang, R., Zavala, V. M., and Biegler, L. T., 2009. "Advanced step nonlinear model predictive control for air separation units." *Journal of Process Control,* **19**(4), pp. 678–685. 1

[12] Martin, R. A., Gates, N. S., Ning, A., and Hedengren, J. D. "Dynamic optimization of high-altitude solar aircraft trajectories under station-keeping constraints." *Journal of Guidance, Control, and Dynamics* Accepted. 1

[13] Zavala, V. M., and Biegler, L. T., 2009. "Optimization-based strategies for the operation of low-density polyethylene tubular reactors: Moving horizon estimation." *Computers & Chemical Engineering,* **33**(1), pp. 379–390. 1

[14] Baldea, M., and Harjunkoski, I., 2014. "Integrated production scheduling and process control: A systematic review." *Computers & Chemical Engineering,* **71**, pp. 377–390. 3, 5, 31, 32, 55, 57, 69, 77, 80, 85

[15] Engell, S., and Harjunkoski, I., 2012. "Optimal operation: Scheduling, advanced control and their integration." *Computers & Chemical Engineering,* **47**, pp. 121–133. 3, 5, 31, 32, 79, 80, 85

[16] Harjunkoski, I., Nyström, R., and Horch, A., 2009. "Integration of scheduling and control-Theory or practice?." *Computers & Chemical Engineering,* **33**, pp. 1909–1918. 3, 31, 56, 77, 79, 80

[17] Shobrys, D. E., and White, D. C., 2002. "Planning, scheduling and control systems: why cannot they work together." *Computers & Chemical Engineering,* **26**(2), pp. 149–160. 3, 31, 79

[18] Beal, L. D. R., Petersen, D., Pila, G., Davis, B., Warnick, S., and Hedengren, J. D., 2017. "Economic benefit from progressive integration of scheduling and control for continuous

chemical processes.” *Processes,* **5**(4). 3, 4, 5, 31

[19] Capón-García, E., Guillén-Gosálbez, G., and Espuña, A., 2013. “Integrating process dynamics within batch process scheduling via mixed-integer dynamic optimization.” *Chemical Engineering Science,* **102**, pp. 139–150. 3, 31, 33, 34, 56, 57, 77, 79

[20] Zhuge, J., and Ierapetritou, M. G., 2012. “Integration of Scheduling and Control with Closed Loop Implementation.” *Industrial & Engineering Chemistry Research,* **51**(25), pp. 8550–8565. 4, 5, 33, 35, 36, 79, 85

[21] Baldea, M., Du, J., Park, J., and Harjunkoski, I., 2015. “Integrated production scheduling and model predictive control of continuous processes.” *AIChE Journal,* **61**(12), pp. 4179–4190. 4, 5, 33, 34, 57, 79, 82

[22] Baldea, M., Touretzky, C. R., Park, J., Pattison, R. C., and Harjunkoski, I., 2016. “Handling input dynamics in integrated scheduling and control.” In *Automation, Quality and Testing, Robotics (AQTR), 2016 IEEE International Conference on*, IEEE, pp. 1–6. 4, 5, 33, 34, 57, 79, 82, 94

[23] Chu, Y., and You, F., 2013. “Integration of production scheduling and dynamic optimization for multi-product CSTRs: Generalized Benders decomposition coupled with global mixed-integer fractional programming.” *Computers & Chemical Engineering,* **58**, pp. 315–333. 4, 5, 33, 41, 57, 79, 82

[24] Chu, Y., and You, F., 2012. “Integration of scheduling and control with online closed-loop implementation: Fast computational strategy and large-scale global optimization algorithm.” *Computers & Chemical Engineering,* **47**, pp. 248–268. 4, 5, 33, 35, 36, 41, 57, 79, 82

[25] Chu, Y., and You, F., 2014. “Moving Horizon Approach of Integrating Scheduling and Control for Sequential Batch Processes.” *AIChE Journal,* **60**(5), pp. 1654–1671. 4, 33, 35, 79, 80, 82

[26] Pattison, R. C., Touretzky, C. R., Johansson, T., Harjunkoski, I., and Baldea, M., 2016. “Optimal Process Operations in Fast-Changing Electricity Markets: Framework for Scheduling with Low-Order Dynamic Models and an Air Separation Application.” *Industrial & Engi-*

neering Chemistry Research, **55**(16), pp. 4562–4584. 5, 33, 34, 36

[27] Pattison, R. C., Touretzky, C. R., Harjunkoski, I., and Baldea, M., 2017. "Moving Horizon Closed-Loop Production Scheduling Using Dynamic Process Models." *AIChE Journal,* **63**(2), pp. 639–651. 5, 31, 33, 34, 36, 41, 82

[28] Terrazas-Moreno, S., Flores-Tlacuahuac, A., and Grossmann, I. E., 2007. "Simultaneous cyclic scheduling and optimal control of polymerization reactors." *AIChE Journal.* 5, 33, 34

[29] Flores-Tlacuahuac, A., and Grossmann, I. E., 2006. "Simultaneous Cyclic Scheduling and Control of a Multiproduct CSTR." *Industrial & Engineering Chemistry Research,* **45**(20), pp. 6698–6712. 5, 33, 57, 78, 79, 82

[30] Flores-Tlacuahuac, A., and Grossmann, I. E., 2010. "Simultaneous scheduling and control of multiproduct continuous parallel lines." *Industrial and Engineering Chemistry Research,* **49**(17), pp. 7909–7921. 5, 31, 33, 36, 57, 79, 82

[31] Terrazas-Moreno, S., Flores-Tlacuahuac, A., and Grossmann, I. E., 2008. "Lagrangean heuristic for the scheduling and control of polymerization reactors." *AIChE Journal,* **54**(1), 1, pp. 163–182. 5, 31, 33, 79

[32] Zhuge, J., and Ierapetritou, M. G., 2014. "Integration of Scheduling and Control for Batch Processes Using Multi-Parametric Model Predictive Control." *AIChE Journal,* **60**(9), pp. 3169–3183. 5, 33, 35, 41, 79, 82

[33] Zhuge, J., and Ierapetritou, M. G., 2015. "An Integrated Framework for Scheduling and Control Using Fast Model Predictive Control." *AIChE Journal,* **61**(10), pp. 3304–3319. 5, 33, 35, 36, 41, 79, 82

[34] Zhuge, J., and Ierapetritou, M. G., 2016. "A Decomposition Approach for the Solution of Scheduling Including Process Dynamics of Continuous Processes." *Industrial and Engineering Chemistry Research.* 5, 33, 35, 41, 79, 82

[35] Du, J., Park, J., Harjunkoski, I., and Baldea, M., 2015. "A time scale-bridging approach for integrating production scheduling and process control." *Computers & Chemical Engineering,* **79**, pp. 59–69. 5, 33, 34, 36, 57, 79, 82

[36] Pattison, R. C., Touretzky, C. R., Harjunkoski, I., and Baldea, M., 2016. "Moving horizon closed-loop production scheduling using dynamic process models." *AIChE Journal,* **61**(3), 7, pp. 1–15. 5

[37] Beal, L. D., Park, J., Petersen, D., Warnick, S., and Hedengren, J. D., 2017. "Combined model predictive control and scheduling with dominant time constant compensation." *Computers & Chemical Engineering,* **104**, pp. 271–282. 5, 33, 82

[38] Beal, L. D. R., Clark, J. D., Anderson, M. K., Warnick, S., and Hedengren, J. D., 2017. "Combined Scheduling and Control with Diurnal Constraints and Costs Using a Discrete Time Formulation." In *FOCAPO/CPC*. 5, 82, 86

[39] Beal, L. D., Petersen, D., Grimsman, D., Warnick, S., and Hedengren, J. D., 2018. "Integrated scheduling and control in discrete-time with dynamic parameters and constraints." *Computers & Chemical Engineering,* **115**, pp. 361 – 376. 5, 33, 77

[40] Nishi, T., and Matsumoto, I., 2015. "Petri net decomposition approach to deadlock-free and non-cyclic scheduling of dual-armed cluster tools." *IEEE Transactions on Automation Science and Engineering,* **12**(1), pp. 281–294. 5

[41] Kim, H. J., Lee, J. H., and Lee, T. E., 2014. "Non-cyclic scheduling of a wet station." *IEEE Transactions on Automation Science and Engineering,* **11**(4), pp. 1262–1274. 5

[42] Wikborg, U., and Lee, T. E., 2013. "Noncyclic scheduling for timed discrete-event systems with application to single-armed cluster tools using pareto-optimal optimization." *IEEE Transactions on Automation Science and Engineering,* **10**(3), pp. 699–710. 5

[43] Sakai, M., and Nishi, T., 2017. "Noncyclic scheduling of dual-armed cluster tools for minimization of wafer residency time and makespan." *Advances in Mechanical Engineering,* **9**(4), p. 168781401769321. 5

[44] Kim, H. J., Lee, J. H., and Lee, T. E., 2015. "Time-Feasible Reachability Tree for Noncyclic Scheduling of Timed Petri Nets." *IEEE Transactions on Automation Science and Engineering,* **12**(3), pp. 1007–1016. 5

[45] Kim, H. J., Lee, J. H., and Lee, T. E., 2015. "Noncyclic Scheduling of Cluster Tools With a Branch and Bound Algorithm." *IEEE Transactions on Automation Science and Engineering,*

12(2), pp. 690–700. 5

[46] Rall, L. B., 1981. *Automatic Differentiation: Techniques and Applications.*, Vol. 120 of *Lecture Notes in Computer Science* Springer, Berlin. 6

[47] Cervantes, A., and Biegler, L. T., 1999. “Optimization strategies for dynamic systems.” In *In C. Floudas, P. Pardalos (Eds), Encyclopedia of Optimization*, Kluwer Academic Publishers. 7

[48] Bock, H., and Plitt, K., 1984. “A multiple shooting algorithm for direct solution of optimal control problems*.” *IFAC Proceedings Volumes,* **17**(2), pp. 1603 – 1608 9th IFAC World Congress: A Bridge Between Control Science and Technology, Budapest, Hungary, 2-6 July 1984. 7

[49] Lorenz T. Biegler, 2010. *Nonlinear Programming: Concepts, Algorithms, and Applications to Chemical Processes.* 7

[50] Ross, I. M., and Karpenko, M., 2012. “A review of pseudospectral optimal control: From theory to flight.” *Annual Reviews in Control,* **36**(2), pp. 182 – 197. 7

[51] Qin, S. J., and Badgwell, T. A., 2003. “A survey of industrial model predictive control technology.” *Control Engineering Practice,* **11**(7), pp. 733–764. 7, 55, 58

[52] Findeisen, R., Allgöwer, F., and Biegler, L., 2007. *Assessment and Future Directions of Nonlinear Model Predictive Control.*, Vol. 358. 7

[53] Ellis, M., Durand, H., and Christofides, P. D., 2014. “A tutorial review of economic model predictive control methods.” *Journal of Process Control,* **24**(8), pp. 1156–1178. 8, 56, 78

[54] Ji, L., and Rawlings, J. B., 2015. “Application of MHE to large-scale nonlinear processes with delayed lab measurements.” *Computers & Chemical Engineering,* **80**, pp. 63 – 72. 8, 60

[55] Würth, L., Rawlings, J. B., and Marquardt, W., 2009. “Economic dynamic real-time optimization and nonlinear model predictive control on infinite hoirzons.” *. . . Symposium on Advanced Control . . .* (2). 8

[56] Biegler, L. T., 2007. “An overview of simultaneous strategies for dynamic optimization.”

Chemical Engineering and Processing: Process Intensification, **46**(11), pp. 1043 – 1053 Special Issue on Process Optimization and Control in Chemical Engineering and Processing. 9

[57] Hedengren, J. D., Shishavan, R. A., Powell, K. M., and Edgar, T. F., 2014. "Nonlinear modeling, estimation and predictive control in apmonitor." *Computers & Chemical Engineering,* **70**, pp. 133 – 148 Manfred Morari Special Issue. 9, 56, 65, 68

[58] Hart, W. E., Laird, C., Watson, J.-P., and Woodruff, D. L., 2012. *Pyomo–optimization modeling in python.*, Vol. 67 Springer Science & Business Media. 10, 48, 100

[59] Dunning, I., Huchette, J., and Lubin, M., 2017. "Jump: A modeling language for mathematical optimization." *SIAM Review,* **59**(2), pp. 295–320. 10, 100

[60] Andersson, J., Åkesson, J., and Diehl, M., 2012. "Casadi: A symbolic package for automatic differentiation and optimal control." In *Recent advances in algorithmic differentiation*. Springer, pp. 297–307. 11

[61] Bisschop, J., and Meeraus, A., 1982. "On the development of a general algebraic modeling system in a strategic planning environment." In *Applications*. Springer, pp. 1–29. 11

[62] Fourer, R., Gay, D., and Kernighan, B., 1993. "Ampl." *Danvers, MA: Boyd & Fraser,* **117**. 11

[63] Barton, P. I., and Pantelides, C., 1993. "gproms-a combined discrete/continuous modelling environment for chemical processing systems." *Simulation Series,* **25**, pp. 25–25. 11

[64] Åkesson, J., Årzén, K.-E., Gäfvert, M., Bergdahl, T., and Tummescheit, H., 2010. "Modeling and optimization with optimica and jmodelica. org—languages and tools for solving large-scale dynamic optimization problems." *Computers & Chemical Engineering,* **34**(11), pp. 1737–1749. 11

[65] Houska, B., Ferreau, H. J., and Diehl, M., 2011. "Acado toolkit—an open-source framework for automatic control and dynamic optimization." *Optimal Control Applications and Methods,* **32**(3), pp. 298–312. 11

[66] Ross, I. M., 2004. "User's manual for dido: A matlab application package for solving

optimal control problems." *Tomlab Optimization, Sweden*, p. 65. 11

[67] Patterson, M. A., and Rao, A. V., 2014. "Gpops-ii: A matlab software for solving multiple-phase optimal control problems using hp-adaptive gaussian quadrature collocation methods and sparse nonlinear programming." *ACM Transactions on Mathematical Software (TOMS),* **41**(1), p. 1. 12

[68] Rutquist, P. E., and Edvall, M. M., 2010. "Propt-matlab optimal control software." *Tomlab Optimization Inc,* **260**(1). 12

[69] Becerra, V. M., 2010. "Solving complex optimal control problems at no cost with psopt." In *Computer-Aided Control System Design (CACSD), 2010 IEEE International Symposium on*, IEEE, pp. 1391–1396. 12

[70] Bisschop, J., 2006. *AIMMS - Optimization Modeling*. Lulu.com. 12

[71] Grant, M., and Boyd, S., 2008. "Graph implementations for nonsmooth convex programs." In *Recent Advances in Learning and Control*, V. Blondel, S. Boyd, and H. Kimura, eds., Lecture Notes in Control and Information Sciences. Springer-Verlag Limited, pp. 95–110 `http://stanford.edu/~boyd/graph_dcp.html`. 12

[72] Andersen, M., Dahl, J., Liu, Z., and Vandenberghe, L., 2011. "Interior-point methods for large-scale cone programming." *Optimization for machine learning*, pp. 55–83. 12

[73] Löfberg, J., 2004. "Yalmip : A toolbox for modeling and optimization in matlab." In *In Proceedings of the CACSD Conference*. 12

[74] Mitchell, S., Consulting, S. M., and Dunning, I., 2011. Pulp: A linear programming toolkit for python The University of Auckland, Auckland, New Zealand. 12

[75] Beal, L. D. R., Hill, D. C., Martin, R. A., and Hedengren, J. D., 2018. "Gekko optimization suite." *Processes,* **6**(8). 12, 30, 106

[76] Luis Morales, J., Nocedal, J., hard Waltz, R. A., Liu, G., and Goux, J.-P. "Assessing the Potential of Interior Methods for Nonlinear Optimization.". 13

[77] Forsgren, A., Gill, P. E., and Wright, M. H., 2002. "Interior methods for nonlinear optimization." *SIAM Review,* **44**(4), pp. 525–597. 13, 14, 15

[78] Han, Z., 2015. "Primal-Dual Active-Set Methods for Convex Quadratic Optimization with Applications." PhD thesis, Lehigh. 14

[79] Gill, P. E., Murray, W., and Saunders, M. A., 2002. "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization." *SIAM Journal on Optimization,* **12**(4), pp. 979–1006. 14

[80] Biegler, L. T., 2014. "Nonlinear programming strategies for dynamic chemical process optimization." *Theoretical Foundations of Chemical Engineering,* **48**(5), pp. 541–554. 14

[81] Wong, E. L. S., 2011. "Active-set methods for quadratic programming." *Thesis*, p. 137. 14

[82] Arief Syaichu-Rohman, R. H. M., 2004. Convergence study of some simple gradient projection based qp solvers for mpc. 14

[83] David J. Ternet, L. T. B., 1999. "Interior-point methods for reduced hessian successive quadratic programming.". 14

[84] Bartlett, R. A., Wachter, A., and Biegler, L. T. "Active set vs. interior point strategies for model predictive control." In *Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No.00CH36334)*, Vol. 6, pp. 4229–4233 vol.6. 14

[85] Mishra, S., Topcu, U., and Tomizuka, M., 2011. "Optimization-based constrained iterative learning control." *IEEE Transactions on Control Systems Technology,* **19**(6), pp. 1613–1621. 14

[86] Wilson, R. B., 1963. "A Simplical Algorithm for Concave Programming." PhD thesis, Harvard University. 14

[87] Oberlin, C., and Wright, S. J., 2006. "Active constraint identification in nonlinear programming." {*SIAM*} *Journal on Optimization,* **17**(2), pp. 577–605. 14

[88] Gill, P. E., Murray, W., and Saunders, M. A., 2005. "Users Guide for SQOPT Version 7: A Fortran Package for Large-Scale Linear and Quadratic Programming." *Numerical Analysis Report NA 97-4*, pp. 1–64. 14

[89] Fletcher, R., and Leyffer, S., 2002. *Nonlinear programming without a penalty function.*, Vol. 91. 14, 24

[90] Fletcher, R., and Leyffer, S., 1998. "User manual for filterSQP." *Numerical Analysis Report NA/181*(June 1998). 14

[91] Wächter, A., and Biegler, L. T., 2005. "Line Search Filter Methods for Nonlinear Programming: Motivation and Global Convergence." *SIAM Journal on Optimization,* **16**(1), pp. 1–31. 14

[92] Benson, H. Y., Vanderbei, R. J., and Shanno, D. F., 2002. "Interior-point methods for nonconvex nonlinear programming: Filter methods and merit functions." *Computational Optimization and Applications,* **23**(2), pp. 257–272. 14

[93] Fletcher, R., Gould, N. I. M., Leyffer, S., Toint, P. L., and Wächter, A., 2002. "Global convergence of a trust-region sqp-filter algorithm for general nonlinear programming." *SIAM Journal on Optimization,* **13**(3), pp. 635–659. 14, 19

[94] Ulbrich, M., Ulbrich, S., and Vicente, L. N., 2004. "A globally convergent primal-dual interior-point filter method for nonlinear programming." *Mathematical Programming,* **100**(2), pp. 379–410. 14, 15

[95] Byrd, R. H., Gilbert, J. C., and Nocedal, J., 2000. "A trust region method based on interior point techniques for nonlinear programming." *Mathematical Programming,* **89**(1), pp. 149–185. 15, 114

[96] Byrd, R. H., Hribar, M. E., and Nocedal, J., 1999. "An interior point algorithm for large-scale nonlinear programming." *SIAM Journal on Optimization,* **9**(4), pp. 877–900. 15

[97] Gould, N. I. M., Orban, D., Sartenaer, A., and Toint, P. L., 2001. "Superlinear convergence of primal-dual interior point algorithms for nonlinear programming." *SIAM Journal on Optimization,* **11**(4), pp. 974–1002. 15

[98] Tits, A. L., Wächter, A., Bakhtiari, S., Urban, T. J., and Lawrence, C. T., 2003. "A primal-dual interior-point method for nonlinear programming with strong global and local convergence properties." *SIAM Journal on Optimization,* **14**(1), pp. 173–199. 15

[99] Vanderbei, R. J., and Shanno, D. F., 1999. "An interior-point algorithm for nonconvex nonlinear programming." *Computational Optimization and Applications,* **13**(1/3), pp. 231–252. 15

[100] Yamashita, H., 1998. "A globally convergent primal-dual interior point method for constrained optimization." *Optimization Methods and Software,* **10**(2), pp. 443–469. 15

[101] Janick V. Frasch, Milan Vukov, H. J. F. M. D. "A new quadratic programming strategy for efficient sparsity exploitation in sqp-based nonlinear mpc and mhe.". 19

[102] Murray, W., 1997. "Sequential quadratic programming methods for large-scale problems." *Computational Optimization and Applications,* **7**(1), pp. 127–142. 19

[103] Spellucci, P., 1998. "New technique for inconsistent qp problems in the sqp method." *Mathematical Methods of Operations Research,* **47**(3), pp. 355–400. 19

[104] Leyffer, S., López-Calva, G., and Nocedal, J., 2006. "Interior Methods for Mathematical Programs with Complementarity Constraints." *SIAM Journal on Optimization,* **17**(1), pp. 52–77. 23, 117

[105] Schittkowski, K., 2009. "An active set strategy for solving optimization problems with up to 200,000,000 nonlinear constraints." *Applied Numerical Mathematics,* **59**, pp. 2999–3007. 24

[106] Gratton, S., Toint, P. L., and Tröltzsch, A. "An active-set trust-region method for derivative-free nonlinear bound-constrained optimization." *Optimization Methods & Software,* **26**, pp. 873–894. 24

[107] Byrd, R. H., Nocedal, J., and Waltz, R. A. "KNITRO: An Integrated Package for Nonlinear Optimization.". 24, 114

[108] Wächter, A., and Biegler, L. T., 2004. "On the Implementation of an Interior-Point Filter Line-Search Algorithm for Large-Scale Nonlinear Programming.". 24

[109] Wan, W., 2017. "Advances in Newton-based Barrier Methods for Nonlinear Programming." PhD thesis, Carnegie Mellon. 24

[110] Benson, H. Y., and Shanno, D. F., 2014. "Interior-point methods for nonconvex nonlinear programming: cubic regularization." *Comput Optim Appl,* **58**, pp. 323–346. 24

[111] Wan, W., and Biegler, L. T., 2016. "Structured regularization for barrier NLP solvers." *Computational Optimization and Applications*(January). 24

[112] Shanno, D. F., and Vanderbei, R. J., 2000. "Interior-point methods for nonconvex nonlinear programming: orderings and higher-order methods." *Ser. B,* **87**, pp. 303–316. 24

[113] Mehrotrat, S., 1992. "On the Implementation of a Primal-Dual Interior Point Method." *SIAM J. OPTIMIZATION,* **2**(4), pp. 575–601. 24

[114] Mayne, D. Q., and Polak, E., 1982. "A superlinearly convergent algorithm for constrained optimization problems." Springer, Berlin, Heidelberg, pp. 45–61. 24

[115] Wächter, A., and Biegler, L. T., 2005. "Line Search Filter Methods for Nonlinear Programming: Local Convergence." pp. 32–48. 24

[116] Zhu, Z., and Jian, J., 2009. "An efficient feasible SQP algorithm for inequality constrained optimization." *Nonlinear Analysis: Real World Applications,* **10**, pp. 1220–1228. 24, 26

[117] Byrds, R. H., Schnabelf, R. B., and Shultzt, G. A., 1987. "A Trust Region Algorithm for Nonlinearly Constrained Optimization." *SIAM J. NUMER. ANAL,* **24**(5). 24

[118] Chamberlain, R. M., Powell, M. J. D., Lemarechal, C., and Pedersen, H. C., 1982. "The watchdog technique for forcing convergence in algorithms for constrained optimization." Springer, Berlin, Heidelberg, pp. 1–17. 24

[119] Grippof, L., Lampariellof, F., and Lucidi, S., 1986. "A Nonmonotone Line Search Technique for Newton's Method." *SIAM J. NUMER. ANAL,* **23**(4). 24

[120] Panier, E. R., and Tits, A. L., 1991. "Avoiding the Maratos Effect by Means of a Nonmonotone Line Search I. General Constrained Problems." *SIAM J. NUMER. ANAL,* **28**(4), pp. 1183–1195. 24

[121] Ahookhosh, M., Amini, K., and Peyghami, M. R., 2012. "A nonmonotone trust-region line search method for large-scale unconstrained optimization." *Applied Mathematical Modelling,* **36**, pp. 478–487. 24

[122] Pei, Y., and Zhu, D., 2015. "Local convergence of a trust-region algorithm with line search filter technique for nonlinear constrained optimization." *Applied Mathematics and Computation,* **273**, pp. 797–808. 25

[123] Huang, S., and Wan, Z., 2017. "A new nonmonotone spectral residual method for non-

smooth nonlinear equations." *Journal of Computational and Applied Mathematics,* **313**, pp. 82–101. 25

[124] Yamamoto, T., 2000. "Historical developments in convergence analysis for Newton's and Newton-like methods." *Journal of Computational and Applied Mathematics,* **124**, pp. 1–23. 25

[125] Kchouk, B., and Dussault, J.-P., 2014. "A new family of high-order directions for unconstrained optimization inspired by Chebyshev and Shamanskii methods." *Optimization Methods & Software,* **29**(1), pp. 68–87. 25, 117

[126] Kchouk, B., and Dussault, J.-P., 2013. "The chebyshev-shamanskii method for solving systems of nonlinear equations." *J Optim Theory Appl,* **157**, pp. 148–167. 25

[127] Steihaug, T., Suleiman, S., Sara Suleiman, B., and Steihaug TrondSteihaug, T., 2016. "On the final steps of Newton and higher order methods." *Optim Lett,* **10**, pp. 401–416. 25, 117, 118

[128] Ghadimi, S., Lan, G., Ghadimi, S., and Lan, G., 2016. "Accelerated gradient methods for nonconvex nonlinear and stochastic programming." *Math. Program., Ser. A,* **156**, pp. 59–99. 25, 118

[129] Noor, K. I., and Noor, M. A. "Predictor–corrector Halley method for nonlinear equations.". 25, 118

[130] Hueso, J. L., Martínez, E., and Torregrosa, J. R. "Third order iterative methods free from second derivative for nonlinear systems.". 25, 119

[131] Li, D., Liu, P., and Kou, J., 2014. "An improvement of Chebyshev-Halley methods free from second derivative.". 25, 119

[132] Gundersen, G., and Steihaug, T., 2010. "On large-scale unconstrained optimization problems and higher order methods." *Optimization Methods & Software,* **25**(3), pp. 337–358. 25

[133] Janka, D., Kirches, C., Sager, S., and Wächter, A., 2016. "An SR1/BFGS SQP algorithm for nonconvex nonlinear programs with block-diagonal Hessian matrix." *Mathematical Pro-*

gramming Computation, **8**(4), 12, pp. 435–459. 25

[134] Curtis, F. E., Mitchell, T., and Overton, M. L., 2017. "A BFGS-SQP method for nonsmooth, nonconvex, constrained optimization and its evaluation using relative minimization profiles A BFGS-SQP method for nonsmooth, nonconvex, constrained optimization and its evaluation using relative minimization profiles." *Optimization Methods & Software,* **32**(1), pp. 148–181. 25

[135] Li, l., and Yang, Z., 2017. "A QP-free algorithm without a penalty function or a filter for nonlinear general-constrained optimization." *Applied Mathematics and Computation,* **316**, pp. 52–72. 26

[136] Armand, P., and Omheni, R., 2017. "A Mixed Logarithmic Barrier-Augmented Lagrangian Method for Nonlinear Optimization." *J Optim Theory Appl,* **173**(173). 26

[137] Jian, J. B., 2006. "New Sequential Quadratically Constrained Quadratic Programming Method of Feasible Directions and Its Convergence Rate 1,2." *Journal of Optimization Theory and Applications,* **129**(1), pp. 109–130. 26

[138] Byrd, R. H., Curtis, F. E., and Nocedal, J. "Infeasibility Detection and SQP Methods for Nonlinear Optimization." pp. 2281–2299. 26

[139] Facchinei, F., Kungurtsev, V., Lampariello, L., and Scutari, G., 2017. "Ghost penalties in nonconvex constrained optimization: Diminishing stepsizes and iteration complexity.". 26

[140] Harjunkoski, I., Maravelias, C. T., Bongers, P., Castro, P. M., Engell, S., Grossmann, I. E., Hooker, J., Méndez, C., Sand, G., and Wassick, J., 2014. "Scope for industrial applications of production scheduling models and solution methods." *Computers & Chemical Engineering,* **62**, pp. 161–193. 27, 31, 32, 56, 78

[141] Méndez, C. a., Cerdá, J., Grossmann, I. E., Harjunkoski, I., and Fahl, M., 2006. "State-of-the-art review of optimization methods for short-term scheduling of batch processes." *Computers & Chemical Engineering,* **30**, pp. 913–946. 27

[142] Petersen, D., Beal, L. D. R., Prestwich, D., Warnick, S., and Hedengren, J. D., 2017. "Combined noncyclic scheduling and advanced control for continuous chemical processes." *Processes,* **5**(4). 30, 31, 40, 41

[143] Backx, T., Bosgra, O., and Marquardt, W. "Integration of model predictive control and optimization of processes." In *IFAC Symposium: Advanced Control of Chemical Processes*, pp. 249–260. 31, 56

[144] Soderstrom, T.A., Z. Y., and Hedengren, J., 2010. "Advanced Process Control in Exxon-Mobil Chemical Company: Successes and Challenges." In *CAST Division, AIChE National Meeting*. 31, 55, 56, 77, 78

[145] Nyström, R. H., Harjunkoski, I., and Kroll, A., 2006. "Production optimization for continuously operated processes with optimal operation and scheduling of multiple units." *Computers and Chemical Engineering,* **30**(3), pp. 392–406. 31, 33, 41

[146] Chatzidoukas, C., Perkins, J. D., Pistikopoulos, E. N., and Kiparissides, C., 2003. "Optimal grade transition and selection of closed-loop controllers in a gas-phase olefin polymerization fluidized bed reactor." *Chemical Engineering Science,* **58**(0009), pp. 3643–3658. 31, 79

[147] Nie, Y., Biegler, L. T., and Wassick, J. M., 2012. "Integrated scheduling and dynamic optimization of batch processes using state equipment networks." *AIChE Journal,* **58**(11), 11, pp. 3416–3432. 31, 33, 34, 57, 79

[148] Baldea, M., 2017. "Advances in Energy Systems Engineering." In *Advances in Energy Systems Engineering*. pp. 247–271. 33, 36, 81

[149] Cai, Y., Kutanoglu, E., Hasenbein, J., and Qin, J., 2012. "Single-machine scheduling with advanced process control constraints." *Journal of Scheduling,* **15**(2), pp. 165–179. 33, 36

[150] Chatzidoukas, C., Kiparissides, C., Perkins, J. D., and Pistikopoulos, E. N., 2003. "Optimal grade transition campaign scheduling in a gas-phase polyolefin FBR using mixed integer dynamic optimization." *Computer Aided Chemical Engineering,* **15**(C), pp. 744–747. 33, 34

[151] Chatzidoukas, C., Pistikopoulos, S., and Kiparissides, C., 2009. "A Hierarchical Optimization Approach to Optimal Production Scheduling in an Industrial Continuous Olefin Polymerization Reactor." *Macromolecular Reaction Engineering,* **3**(1), pp. 36–46. 33, 34, 79

[152] Chu, Y., and You, F., 2013. "Integrated Scheduling and Dynamic Optimization of Sequential

Batch Processes with Online Implementation.” *AIChE Journal,* **59**(7), pp. 2379–2406. 33, 35

[153] Chu, Y., and You, F., 2013. “Integrated Scheduling and Dynamic Optimization of Complex Batch Processes with General Network Structure Using a Generalized Benders Decomposition Approach.” *Industrial & Engineering Chemistry Research,* **52**(23), pp. 7867–7885. 33, 34, 79

[154] Chu, Y., and You, F., 2013. “Integration of scheduling and dynamic optimization of batch processes under uncertainty: Two-stage stochastic programming approach and enhanced generalized benders decomposition algorithm.” *Industrial and Engineering Chemistry Research,* **52**(47), pp. 16851–16869. 33, 34, 35, 79

[155] Chu, Y., and You, F., 2014. “Integrated Planning, Scheduling, and Dynamic Optimization for Batch Processes: MINLP Model Formulation and Efficient Solution Methods via Surrogate Modeling.” *Industrial & Engineering Chemistry Research,* **53**(34), pp. 13391–13411. 33, 41, 80

[156] Chu, Y., and You, F., 2014. “Integrated scheduling and dynamic optimization by stackelberg game: Bilevel model formulation and efficient solution algorithm.” *Industrial and Engineering Chemistry Research,* **53**(13), pp. 5564–5581. 33, 34, 80

[157] Dias, L. S., and Ierapetritou, M. G., 2016. “Integration of scheduling and control under uncertainties: Review and challenges.” *Chemical Engineering Research and Design,* **116**, pp. 98–113. 33, 35, 79, 80

[158] Gutierrez-Limon, M. A., Flores-Tlacuahuac, A., and Grossmann, I. E., 2011. “A Multiobjective Optimization Approach for the Simultaneous Single Line Scheduling and Control of CSTRs.” *Industrial & Engineering Chemistry Research,* **51**, pp. 5881–5890. 33, 79, 82

[159] Gutierrez-Limon, M. A., Flores-Tlacuahuac, A., and Grossmann, I. E., 2016. “A reactive optimization strategy for the simultaneous planning, scheduling and control of short-period continuous reactors.” *Computers and Chemical Engineering,* **84**. 33, 36, 80

[160] Gutiérrez-Limón, M. A., Flores-Tlacuahuac, A., and Grossmann, I. E., 2014. “MINLP formulation for simultaneous planning, scheduling, and control of short-period single-unit

processing systems." *Industrial and Engineering Chemistry Research,* **53**(38), pp. 14679–14694. 33, 36, 80

[161] Koller, R. W., and Ricardez-Sandoval, L. A., 2017. "A Dynamic Optimization Framework for Integration of Design, Control and Scheduling of Multi-product Chemical Processes under Disturbance and Uncertainty." *Computers & Chemical Engineering*. 33, 80

[162] Nie, Y., Biegler, L. T., Villa, C. M., and Wassick, J. M., 2015. "Discrete Time Formulation for the Integration of Scheduling and Dynamic Optimization." *Industrial & Engineering Chemistry Research,* **54**(16), pp. 4303–4315. 33, 34, 82

[163] Patil, B. P., Maia, E., and Ricardez-Sandoval, L. A., 2015. "Integration of Scheduling, Design, and Control of Multiproduct Chemical Processes Under Uncertainty." *AIChE Journal,* **61**(8), pp. 2456–2470. 33, 34, 36, 80

[164] Prata, A., Oldenburg, J., Kroll, A., and Marquardt, W., 2008. "Integrated scheduling and dynamic optimization of grade transitions for a continuous polymerization reactor." *Computers & Chemical Engineering,* **32**(3), pp. 463–476. 33, 57, 79

[165] Terrazas-Moreno, S., Flores-Tlacuahuac, A., and Grossmann, I. E., 2008. "Simultaneous design, scheduling, and optimal control of a methyl-methacrylate continuous polymerization reactor." *AIChE Journal*. 33, 34, 80

[166] You, F., and Grossmann, I. E., 2008. "Design of responsive supply chains under demand uncertainty." *Computers and Chemical Engineering,* **32**(12), pp. 3090–3111. 33, 36

[167] Mahadevan, R., Doyle, F. J., and Allcock, A. C., 2002. "Control-relevant scheduling of polymer grade transitions." *AIChE Journal,* **48**(8), pp. 1754–1764. 34

[168] Mojica, J. L., Petersen, D., Hansen, B., Powell, K. M., and Hedengren, J. D., 2017. "Optimal combined long-term facility design and short-term operational strategy for CHP capacity investments." *Energy,* **118**, pp. 97–115. 34, 82

[169] Gupta, D., Maravelias, C. T., and Wassick, J. M., 2016. "From rescheduling to online scheduling." *Chemical Engineering Research and Design,* **116**, pp. 83 – 97 Process Systems Engineering - A Celebration in Professor Roger Sargent's 90th Year. 35, 36

[170] Gupta, D., and Maravelias, C. T., 2016. “On deterministic online scheduling: Major considerations, paradoxes and remedies.” *Computers and Chemical Engineering,* **94**(Supplement C), pp. 312 – 330. 35, 36

[171] Gupta, D., and Maravelias, C. T., 2017. “A general state-space formulation for online scheduling.” *Processes,* **69**(4). 35, 36

[172] Kopanos, G. M., and Pistikopoulos, E. N., 2014. “Reactive scheduling by a multiparametric programming rolling horizon framework: A case of a network of combined heat and power units.” *Industrial and Engineering Chemistry Research,* **53**(11), pp. 4366–4386. 36

[173] Liu, S., Dong, Y., Liu, W., and Zhao, J., 2012. “Multi-view face detection based on cascade classifier and skin color.” *Ieee Ccis2012*, pp. 3–7. 36

[174] Li, Z., and Ierapetritou, M. G., 2007. “Process Scheduling Under Uncertainty Using Multiparametric Programming.” *AIChE Journal,* **53**(12), pp. 3183–3203. 35

[175] Hart, W. E., Watson, J.-P., and Woodruff, D. L., 2011. “Pyomo: modeling and solving mathematical programs in python.” *Mathematical Programming Computation,* **3**(3), pp. 219–260. 48, 100

[176] Carey, G., and Finlayson, B. A., 1975. “Orthogonal collocation on finite elements for elliptic equations.” *Chemical Engineering Science,* **30**(1), pp. 587–596. 48

[177] Hedengren, J., Mojica, J., Cole, W., and Edgar, T., 2012. “APOPT: MINLP Solver for Differential Algebraic Systems with Benchmark Testing.” In *INFORMS Annual Meeting, Pheonix, AZ.* 48, 88, 100

[178] Couenne, 2006. Couenne (convex over and under envelopes for nonlinear estimation). 48, 100

[179] Beal, L. D., Park, J., Petersen, D., Warnick, S., and Hedengren, J. D., 2017. “Combined model predictive control and scheduling with dominant time constant compensation.” *Computers & Chemical Engineering,* **104**, pp. 271–282. 55

[180] Lima, F. V., Rajamani, M. R., Soderstrom, T. A., and Rawlings, J. B., 2013. “Covariance and State Estimation of Weakly Observable Systems: Application to Polymerization Processes.”

IEEE Transactions on Control Systems Technology, **21**(4), 7, pp. 1249–1257. 56, 78

[181] Ma, W., and Agrawal, G., 2010. “An integer programming framework for optimizing shared memory use on gpus.” In *2010 International Conference on High Performance Computing*, pp. 1–10. 56

[182] Angeli, D., Amrit, R., and Rawlings, J. B., 2012. “On Average Performance and Stability of Economic Model Predictive Control.” *IEEE Transactions on Automatic Control,* **57**(7), 7, pp. 1615–1626. 56

[183] Subramanian, K., Rawlings, J. B., and Maravelias, C. T., 2014. “Economic model predictive control for inventory management in supply chains.” *Computers & Chemical Engineering,* **64**, pp. 71–80. 56, 78

[184] Biegler, L., Yang, X., and Fischer, G., 2015. “Advances in sensitivity-based nonlinear model predictive control and dynamic real-time optimization.” *Journal of Process Control,* **30**, pp. 104–116. 56, 78

[185] Kelly, J., and Hedengren, J., 2013. “A steady-state detection (SSD) algorithm to detect non-stationary drifts in processes.” *Journal of Process Control,* **23**(3), March, pp. 326–331. 56, 69

[186] Zhuge, J., and Ierapetritou, M. G., 2012. “Integration of Scheduling and Control with Closed Loop Implementation.” *Industrial & Engineering Chemistry Research,* **51**(25), pp. 8550–8565. 57, 82

[187] Farhangi, H., 2010. “The Path of the Smart Grid 18.” *IEEE Power & Energy Mag.,*, February, p. 18–28. 57, 80

[188] U S Department of Energy, 2006. Benefits of Demand Response in Electricity Markets and Recommendations for Achieving Them Tech. Rep. February, US DOE. 57, 80

[189] Safdarnejad, S. M., Hedengren, J. D., and Baxter, L. L., 2016. “Dynamic optimization of a hybrid system of energy-storing cryogenic carbon capture and a baseline power generation unit.” *Applied Energy,* **172**, pp. 66 – 79. 57, 80

[190] Safdarnejad, S. M., Hedengren, J. D., and Baxter, L. L., 2015. “Plant-level dynamic op-

timization of cryogenic carbon capture with conventional and renewable power sources.” *Applied Energy,* **149**, pp. 354 – 366. 57, 80

[191] Safdarnejad, S., Kennington, L., Baxter, L., and Hedengren, J., 2015. “Investigating the impact of cryogenic carbon capture on power plant performance.” In *Proceedings of the American Control Conference (ACC)*, pp. 5016–5021. 57

[192] Deng, R., Yang, Z., Chow, M.-Y., and Chen, J., 2015. “A Survey on Demand Response in Smart Grids: Mathematical Models and Approaches.” *IEEE Transactions on Industrial Informatics,* **11**(3), pp. 1–1. 57, 80

[193] Beal, L., Clark, J., Anderson, M., Warnick, S., and Hedengren, J., 2017. “Combined scheduling and control with diurnal constraints and costs using a discrete time formulation.” In *FOCAPO / CPC 2017*, Foundations of Computer Aided Process Operations, Chemical Process Control. 57, 69, 73

[194] Mendoza-Serrano, D. I., and Chmielewski, D. J., 2013. “Demand Response for Chemical Manufacturing using Economic MPC.” *Proceedings of 2013 American Control Conference (ACC)*, pp. 6655–6660. 57, 80, 81

[195] Huang, R., Harinath, E., and Biegler, L. T., 2011. “Lyapunov stability of economically oriented NMPC for cyclic processes.” *Journal of Process Control,* **21**(4), pp. 501–509. 57, 81

[196] Hedengren, J. D., and Eaton, A. N., 2017. “Overview of estimation methods for industrial dynamic systems.” *Optimization and Engineering,* **18**(1), March, pp. 155–178. 58

[197] Zhang, Y., Naidu, D. S., Nguyen, H. M., Cai, C., and Zou, Y., 2014. “Time scale analysis and synthesis for model predictive control under stochastic environments.” In *2014 7th International Symposium on Resilient Control Systems (ISRCS)*, pp. 1–6. 60

[198] Srinivasagupta, D., Schättler, H., and Joseph, B., 2004. “Time-stamped model predictive control: an algorithm for control of processes with random delays.” *Computers & Chemical Engineering,* **28**(8), pp. 1337 – 1346. 60

[199] Seborg, D. E., Mellichamp, D. A., Edgar, T. F., and Doyle III, F. J., 2010. *Process dynamics and control*. John Wiley & Sons. 61

[200] Wang, Y., and Boyd, S., 2010. "Fast model predictive control using online optimization." *IEEE Transactions on Control Systems Technology,* **18**(2), pp. 267–278. 61

[201] Powell, K. M., Eaton, A. N., Hedengren, J. D., and Edgar, T. F., 2016. "A continuous formulation for logical decisions in differential algebraic systems using mathematical programs with complementarity constraints." *Processes,* **4**(1), p. 7. 62

[202] Baumrucker, B., Renfro, J., and Biegler, L. T., 2008. "Mpec problem formulations and solution strategies with chemical engineering applications." *Computers & Chemical Engineering,* **32**(12), pp. 2903–2913. 62

[203] Hedengren, J. D., Shishavan, R. A., Powell, K. M., and Edgar, T. F., 2014. "Nonlinear modeling, estimation and predictive control in APMonitor." *Computers & Chemical Engineering,* **70**, 5, pp. 133–148. 78

[204] Angeli, D., Amrit, R., and Rawlings, J. B., 2012. "On Average Performance and Stability of Economic Model Predictive Control." *IEEE Transactions on Automatic Control,* **57**(7), jul, pp. 1615–1626. 78

[205] Pistikopoulos, E. N., and Diangelakis, N. A., 2015. "Towards the integration of process design, control and scheduling: Are we getting closer?." *Computers & Chemical Engineering*. 79

[206] Shi, L., Jiang, Y., Wang, L., and Huang, D., 2014. "Refinery production scheduling involving operational transitions of mode switching under predictive control system." *Industrial and Engineering Chemistry Research,* **53**(19), pp. 8155–8170. 79

[207] Charitopoulos, V. M., Dua, V., and Papageorgiou, L. G., 2017. "Traveling salesman problem-based integration of planning, scheduling, and optimal control for continuous processes." *Industrial & Engineering Chemistry Research,* **56**(39), pp. 11186–11205. 79

[208] Mishra, B. V., Mayer, E., Raisch, J., and Kienle, A., 2005. "Short-term scheduling of batch processes. A comparative study of different approaches." *Industrial and Engineering Chemistry Research,* **44**(11), p. 4022. 79

[209] Rossi, F., Casas-Orozco, D., Reklaitis, G., Manenti, F., and Buzzi-Ferraris, G., 2017. "A Computational Framework for Integrating Campaign Scheduling, Dynamic Optimization

and Optimal Control in Multi-Unit Batch Processes." *Computers & Chemical Engineering*. 79

[210] Tong, X., Liu, X., Chen, P., Liu, S., Luan, K., Li, L., Liu, S., Liu, X., Xie, H., Jin, Y., and Hong, Z., 2015. "Integration of UAV-Based Photogrammetry and Terrestrial Laser Scanning for the Three-Dimensional Mapping and Monitoring of Open-Pit Mine Areas." *Remote Sensing,* **7**(6), pp. 6635–6662. 80, 81, 85

[211] Zhang, X., and Hug, G., 2015. "Bidding strategy in energy and spinning reserve markets for aluminum smelters' demand response." *2015 IEEE Power and Energy Society Innovative Smart Grid Technologies Conference, ISGT 2015*, pp. 1–5. 81

[212] Zhang, X., Hug, G., Kolter, Z., and Harjunkoski, I., 2015. "Industrial demand response by steel plants with spinning reserve provision." *2015 North American Power Symposium, NAPS 2015*. 81

[213] Xu, J., and Wang, L., 2017. "A Feedback Control Method for Addressing the Production Scheduling Problem by Considering Energy Consumption and Makespan." *Sustainability,* **9**(7), p. 1185. 81

[214] Tong, C., Palazoglu, A., and El-Farra, N. H., 0. "A decomposition scheme for integration of production scheduling and control: Demand response to varying electricity prices." *Industrial & Engineering Chemistry Research,* **0**(ja), p. null. 81

[215] Floudas, C. A., and Lin, X., 2004. "Continuous-time versus discrete-time approaches for scheduling of chemical processes: a review." *Computers & Chemical Engineering,* **28**(11), pp. 2109–2129. 82

[216] Wan, W., and Biegler, L. T., 2016. "Structured regularization for barrier NLP solvers." *Computational Optimization and Applications*(January). 86

[217] A. Waechter, A., and Biegler, L. T., 2006. *On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming*., Vol. 106. 87, 100

[218] Safdarnejad, S. M., Hedengren, J. D., Lewis, N. R., and Haseltine, E. L., 2015. "Initialization strategies for optimization of dynamic systems." *Computers and Chemical Engineering,*

78, pp. 39–50. 90

[219] Khalil, H. K., and Grizzle, J., 1996. *Nonlinear systems.*, 3 ed. Prentice hall New Jersey. 93, 146, 147

[220] Hespanha, J. P., 2009. *Linear systems theory*. Princeton university press. 94

[221] Hedengren, J. D., Mojica, J. L., Cole, W., and Edgar, T. F., 2012. "Apopt: Minlp solver for differential algebraic systems with benchmark testing." In *Proceedings of the INFORMS Annual Meeting (2012)*. 114

[222] Biegler, L., 2010. *Nonlinear Programming: Concepts, Algorithms, and Applications to Chemical Processes*. 114, 117

[223] Bartlett, R. A., and Biegler, L. T., 2006. "QPSchur: A dual, active-set, Schur-complement method for large-scale and structured convex quadratic programming." *Optim Eng,* **7**, pp. 5–32. 114

[224] Chang-Yu, W., Yuan-Yuan, C., and Shou-Qiang, D. "Further insight into the shamanskii modification of newton method.". 117

[225] Cuyt, A., and B. Rall, L., 1985. "Computational implementation of the multivariate halley method for solving nonlinear systems of equations.." pp. 20–36. 118

APPENDIX A. COMBINED SCHEDULING AND CONTROL EXAMPLE

The following APMonitor model and Python script detail the variables, equations, and commands necessary to reproduce the combined scheduling and control presented in Section 3.4.1.

Listing A.1: Constants and Parameters

```
Constants
  n = 3  % products

Parameters
  % transition points for steps
  b[1] = 0
  b[2] = 1
  b[3] = 2
  b[4] = 3
  b[5] = 4
  b[6] = 5
  % step up (+) or down (-)
  sg[1] = 1
  sg[2] = -1
  sg[3] = 1
  sg[4] = -1
  sg[5] = 1
  sg[6] = -1
  % magnitude of step function
  m[1] = 1
  m[2] = 1
  m[3] = 2
  m[4] = 2
  m[5] = 3
  m[6] = 3
  % demand for each product
  d[1] = 2
  d[2] = 5
  d[3] = 3
  % flowrate
  q = 2
  % manipulated variable
  u = 0.0 >= 0.0  <= 8.0
  % zero everywhere except last point
  last = 0
```

Listing A.2: Variables and Equations

```
Variables
  x = 0.0
  % individual step functions
  w[1:2*n] >= 0 , <= 1
  % slack variables
  % should be positive when x - b is negative
  s1[1:2*n] >= 0 , <= 1000
  % should be positive when x - b is positive
  s2[1:2*n] >= 0 , <= 1000
  % profit function
  pfcn = 0  >= 0  <= 3
  % total profit at each time step
  profit
  % which is product is being produced
  prod[1:n] = 0
  % integral of product
  iprod[1:n] = 0

Intermediates
  % sum steps
  z[0] = 0
  z[1:2*n] = z[0:2*n-1] + sg[1:2*n] * m[1:2*n] * w[1:2*n]

Equations
  pfcn = z[2*n]
  profit = pfcn * q
  d(x)/dt = -x + u
  prod[1] = w[1] - w[2]
  prod[2] = w[3] - w[4]
  prod[3] = w[5] - w[6]
  d(iprod[1:n])/dt = prod[1:n] * q
  x - b[1:2*n] =  s2[1:2*n] - s1[1:2*n]
  last * (iprod[1:n] - d[1:n]) >= 0
  % include as alternative to objective version
  %s1[1:2*n]*(w[1:2*n]) <= 0
  %s2[1:2*n]*(1-w[1:2*n]) <= 0
  minimize 10000 * s1[1:2*n]*(w[1:2*n])
  minimize 10000 * s2[1:2*n]*(1-w[1:2*n])
  maximize profit
```

The application uses two elements including the model file (schedule.apm), shown in Listings A.1 and A.2, and a data file (schedule.csv). The data file is a list of times between 0 and 7 with time increment 0.1 and another column labeled *last* that is 0 everywhere except the end point as 1. The parameter *last* is to enforce the constraints that a certain amount of each product should be produced. The model is loaded and other options are set through the python script shown in Listing A.3. All source files are available from https://github.com/APMonitor.

Listing A.3: Python Script for Combined Control and Scheduling

```python
from apm import *

s = 'http://byu.apmonitor.com'
a = 'products'
# set up problem
apm(s,a,'clear all')
apm_load(s,a,'schedule.apm')
csv_load(s,a,'schedule.csv')
# create manipulated variable
apm_info(s,a,'MV','u')
apm_option(s,a,'u.status',1)
apm_option(s,a,'u.dcost',1.0)
apm_option(s,a,'u.dmax',0.16)
apm_option(s,a,'u.upper',8.0)
apm_option(s,a,'u.lower',0.0)
# set options
apm_option(s,a,'nlc.imode',6)
apm_option(s,a,'nlc.max_iter',200)
apm_option(s,a,'nlc.solver',3)
apm_option(s,a,'nlc.nodes',2)
# solve combined scheduling and control
output = apm(s,a,'solve')
print(output)
# retrieve solution
sol = apm_sol(s,a)
```

APPENDIX B. FEEDBACK LINEARIZATION ESTIMATION: DERIVATION

A linear system $y = f(x,u)$ has the property that $f(x,u_0+u_1) = f(x,u_0)+f(x,u_1)$. This means that the response of the system to the initial input u_0 can be decoupled from that of the step size u_1. Using this formulation, a closed-form solution for the transition time given a step size can be estimated to avoid preprocessing time and space.

According previous research [219], any system of the form

$$\begin{aligned} \dot{x} &= f(x)+g(x)u \\ y &= h(x) \end{aligned} \tag{B.1}$$

can be feedback linearized. In the present example, this means (substituting $C_A = x_1$, $T = x_2$, and $T_c = u$ in Eqs. 2.10-2.11)

$$\begin{aligned} \frac{dx_1}{dt} &= 5-5x_1-1.8\cdot 10^{10}x_1e^{-8750/x_2} \\ \frac{dx_2}{dt} &= 1750-5.52x_2+3.77\cdot 10^{12}x_1e^{-8750/x_2}+0.523u \end{aligned} \tag{B.2}$$

$$\begin{aligned} f(x) &= \begin{bmatrix} 5-5x_1-1.8\cdot 10^{10}x_1e^{-8750/x_2} \\ 1750-5.52x_2+3.77\cdot 10^{12}x_1e^{-8750/x_2} \end{bmatrix}, \\ g(x) &= \begin{bmatrix} 0 \\ 0.523 \end{bmatrix}, h(x) = x_1 \end{aligned} \tag{B.3}$$

To linearize the output y in terms of an input v that shapes u, an input-output representation of the system is found by taking the 1st and then 2nd time derivative of y.

$$\begin{aligned}\dot{y} &= \frac{dh}{dt} = \frac{\partial h}{\partial x}\frac{dx}{dt} \\ &= \frac{\partial h}{\partial x}(f(x)+g(x)u) \\ &= \frac{\partial h}{\partial x}f(x)\end{aligned} \tag{B.4}$$

since $\frac{\partial h}{\partial x} = [1\ 0]$ and $g(x) = [0\ 0.523]^T$, and therefore $\frac{\partial h}{\partial x}g(x) = 0$. Leveraging (B.4), $\ddot{y}$ can now be solved for:

$$\begin{aligned}\ddot{y} &= \frac{d}{dt}\dot{y} = \frac{d}{dt}\frac{\partial h}{\partial x}f(x) \\ &= \frac{\partial h}{\partial x}\frac{df}{dt} = \frac{\partial h}{\partial x}\frac{\partial f}{\partial x}\frac{dx}{dt} \\ &= \frac{\partial h}{\partial x}\frac{\partial f}{\partial x}(f(x)+g(x)u) \\ &= L_f^2 h(x) + L_g L_f h(x)\end{aligned} \tag{B.5}$$

where $L_f h(x) = \frac{\partial h}{\partial x}f(x)$ is known as the *Lie derivative* of h with respect to f, and $L_f^2 h(x) = L_f L_f h(x)$. Solving equation (B.5) for u yields:

$$u = \frac{1}{L_g L_f h(x)}\Big(-L_f^2 h(x) + \ddot{y}\Big) \tag{B.6}$$

Therefore, letting $v = \ddot{y}$ the system can be structured as in Figure B.1, where

$$\begin{aligned}K_1 &: w = -L_f^2 h(x) \\ K_2 &: u = \frac{v}{L_g L_f h(x)}\end{aligned} \tag{B.7}$$

Since $v = \ddot{y}$, the input-output behavior of the closed-loop system inside the dotted box is the same as a double integrator, and is thus a linear system. According to Khailil *et al.* (1996) [219], the new state vector z for this system is given by:

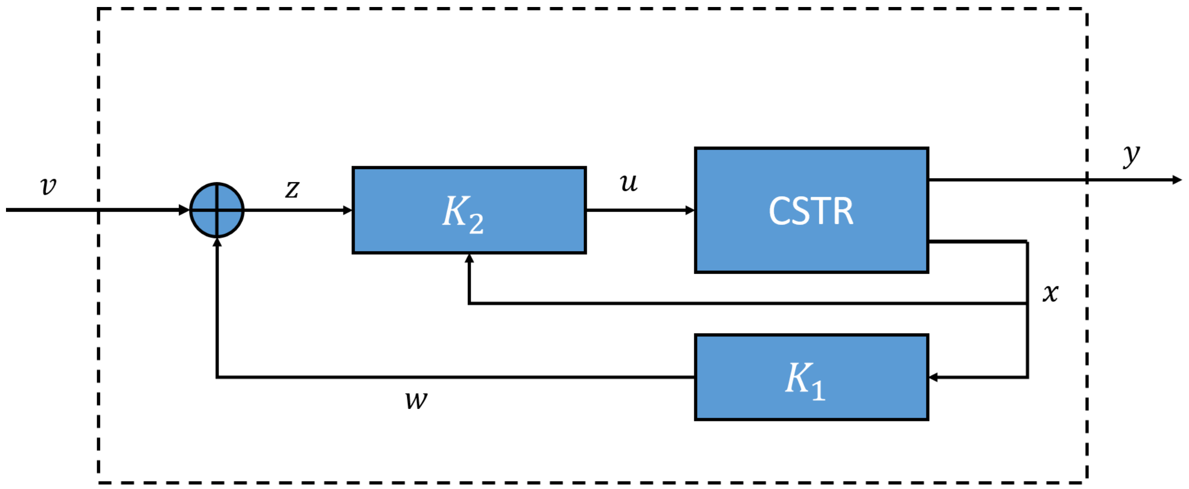


Figure B.1: Feedback linearized system block diagram.

$$z = \begin{bmatrix} h(x) \\ L_f h(x) \end{bmatrix} = \begin{bmatrix} x_1 \\ 5 - 5x_1 - 1.8 \cdot 10^{10} x_1 e^{-8750/x_2} \end{bmatrix} \tag{B.8}$$

With 2 poles on the imaginary axis, the linearized system is marginally stable. In order to compensate for this, a stabilizing controller is needed. One such controller is given by the following transfer function

$$K_3(s) = 27.193 \frac{50s + 100}{s + 50} \tag{B.9}$$

and used in feedback with the standard servo architecture shown in Figure 4.7. Using this configuration, the transition times can be calculated by simply knowing the difference between the starting and end values, or the step size in r. This is a key advantage to linearizing the system in this way. Without a linearized system, a large number of transition times would need to be known *a priori* in order to give the scheduler a comprehensive list. A linearized system only needs to know the increase or decrease of the step response to compensate for the transition time, regardless of the current state.

In order to give the scheduler a complete list of transition times, a function approximation with respect to step size is created. Figure B.2 shows such an approximation. The transition time is determined through simulation for a set of step sizes and then fit to a logarithmic function. The result is the following:

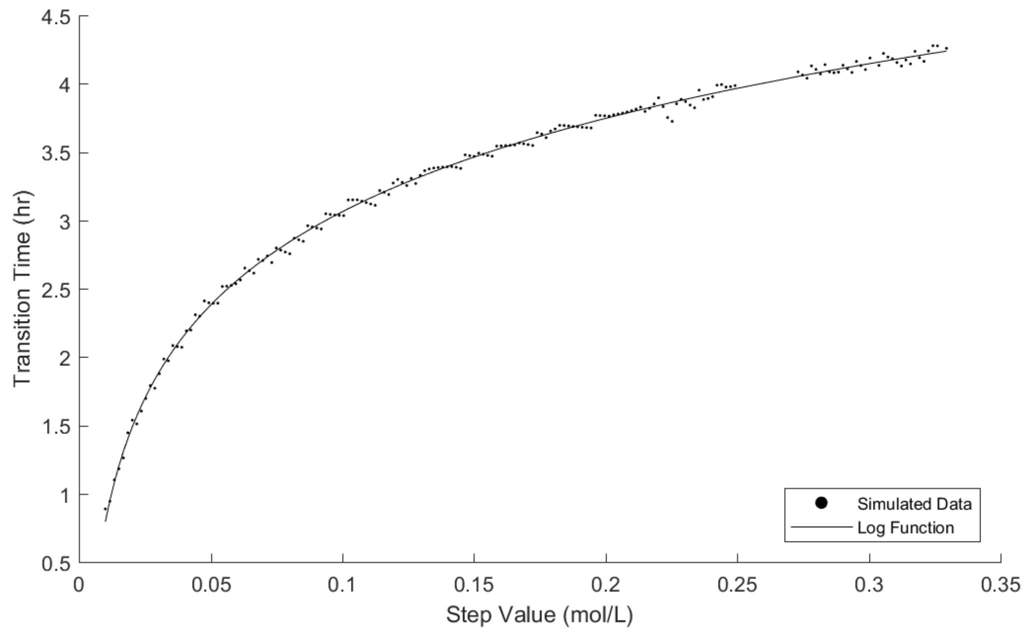


Figure B.2: Actual transition times compared to the log function approximation. The approximation works exceptionally well.

$$t_{trans} = 0.9853 \cdot log(s_{step}) + 5.332 \tag{B.10}$$

where t_{trans} is the transition time and s_{step} is the step size, or difference between starting and ending values. Note here that the transition time is the time for y to settle to within 0.005 of the steady state value.

Thus by linearizing the CSTR system, transition times can be found using Equation (B.10). These transition times can then be used by the continuous-time scheduler to create an optimal schedule for initialization.