15. Symposium Energieinnovation, 14. bis 16. Februar 2018, Technische Universität Graz, www.EnInnov.TUGraz.at
ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

DYNAMISCHE OPTIMIERUNG VON MODELICA-BASIERTEN MODELLEN

Gerald Schweiger[1, 2], David Blum[3], Michael Wetter[3]

1 AEE Intec, Gleisdorf, Österreich

2 TU Graz, Institut für Softwaretechnologie, Graz, Österreich

3 Lawrence Berkeley National Laboratory, Berkeley, CA, USA

Kurzfassung:
Energiesysteme sind im Wandel. Bestehende Systeme müssen effizienter werden und der Anteil erneuerbarer Energien im System soll ansteigen. Die wissenschaftliche Gemeinschaft präsentiert hierfür unterschiedliche Lösungen, die vielfach eine Adaption des gesamten Energiesystems erfordern. Dies führt dazu, dass Politik, Planer und Energieversorger mit völlig neuen Herausforderungen konfrontiert sind und zunehmend unter Druck geraten. Computergestützte Analysen sind sowohl für die Wissenschaft als auch für Entscheidungsträger von zentraler Bedeutung; in vielen Anwendungen ist dabei die dynamische Optimierung von Systemen oder Prozessen das Ziel der Analysen. In diesem Paper werden die Möglichkeiten der Modellierungssprache Modelica im Bereich der dynamischen Optimierung diskutiert. Des Weiteren wird ein Überblick der Literatur von Modelica-basierter dynamischer Optimierung im Bereich von Energiesystemen gegeben. Interviews mit Experten sind die Grundlage um den State of the Art, aktuelle Barrieren, zukünftige Herausforderungen und fundamentale Einschränkungen zu diskutieren.

Keywords: Modellierung, dynamische Optimierung, Energiesysteme, Modelica

1 Einleitung

Die numerische Simulation ist fester Bestandteil der Industrie und Wissenschaft. In vielen Anwendungen ist dabei die Optimierung das Ziel der Untersuchungen. Die Modelle werden häufig nicht in ein mathematisches Optimierungsproblem transformiert, sondern die Domainexpertise der Anwenderin ist die Grundlage, um „optimale“ Systeme durch simulationsgestützte Parameterstudien und Sensitivitätsanalysen oder durch das Testen verschiedener Regelkonzepte zu identifizieren. Grundsätzlich gilt, dass die Optimierung komplexer ist als die Simulation und die Anforderungen an den Anwender als auch an Tool-Entwickler steigen; die Autoren gehen davon aus, dass der eingeschränkte Einsatz von Optimierung in der Industrie und Wissenschaft nicht ausschließlich auf deren Komplexität zurückzuführen ist. Anwenderfreundliche Tools, domainspezifische Optimierungsbibliotheken oder besseres Ausbildungsangebot würden den Einstieg erleichtern.
Generell wird in der Optimierung zwischen der statischen und dynamischen Optimierung unterschieden. Bei der statischen Optimierung wird eine Funktion unter Berücksichtigung von Nebenbedingungen minimiert, wobei die Optimierungsvariablen Elemente des euklidischen Raumes sind. In der dynamischen Optimierung basieren viele Optimierungsprobleme auf Modellen, die durch differential-algebraische (DAE) Systeme

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

beschrieben werden; dabei gilt es oftmals Funktionen einer unabhängigen Variablen zu bestimmen, die in vielen Anwendungen die Zeit ist. Dynamische Optimierungsprobleme finden sich in verschiedenen Disziplinen wie der Robotik, im automotive Sektor, in unterschiedlichen Bereichen des Energiesystemes oder der Biomechanik; dynamische Optimierungsprobleme treten im Rahmen der Optimalsteuerung, der modelprädiktiven Regelung, der Parameterschätzung oder der Zustandsschätzung auf. Die modellprädiktive Regelung basiert vereinfacht gesagt auf der wiederholten Lösung des Optimalsteurungsproblems (Lösung eines dynamischen Optimierungsproblems auf einem bewegten Horizont); dadurch wird die modellprädiktive Regelung zur „Regelung“. Beispiele für dynamische Optimierungsprobleme im Bereich von Energiesystemen sind die Optimierung von GuD-Kraftwerken, der optimale Start eines Kessels oder die modellprädiktive Regelung von Gebäuden, Fernwärmenetzen und Speichern im elektrischen Netz.

2 Dynamische Optimierung

Die generelle Form eines dynamischen Optimierungsproblems lässt sich wie folgt darstellen:

$$\underset{x,y,u,p}{\text{minimize}} \quad \Phi(t_f, x(t_f) + \int_{t_0}^{t_f} L(x(t), y(t), u(t))dt \tag{1a}$$

$$s.t.\ F(\dot{x}(t),\ x(t),\ y(t),\ u(t)) = 0, \quad x(t_0) = x_0 \tag{1b}$$

$$h(\dot{x}(t),\ x(t), y(t),\ u(t), p) \leq 0 \tag{1c}$$

$$g(\dot{x}(t),\ x(t),\ y(t), u(t), p) = 0 \tag{1d}$$

$$H(x(t_f), y(t_f),\ p) \leq 0 \tag{1e}$$

$$\forall t \in [t_0, t_f]$$

Dabei bezeichnet x die Zustandsvariablen, y algebraische Variablen, u Kontrollvariablen, p Parameter und $t \in [t_0, t_f]$ die Zeit. Die Zielfunktion ist in 1a dargestellt; die Systemdynamik wird durch das implizite DAE in 1b beschrieben. 1c-1e sind die Nebenbedingungen. Die Lösung des dynamischen Optimierungsproblems liegt darin, eine Trajektorie $u(t)$ zu finden, welche die Zielfunktion minimiert und die Modellgleichungen sowie Nebenbedingungen erfüllt. Es gibt mehrere Methoden, um dynamische Optimierungsprobleme zu lösen; im Bereich von komplexen Problemen sind numerische Methoden basierend auf notwendigen Optimalitätskriterien erster Ordnung für lokale Optima verbreitet [1]. Methoden können weiteres in direkte und indirekte Lösungsverfahren unterteilt werden; indirekte Methoden basieren auf der Auswertung der Optimalitätsbedingungen und führen meist zu Mehrpunkt-Randwertproblemen, die numerisch gelöst werden. Beispiele für indirekte Verfahren sind indirekte Kollokationsverfahren oder indirekte Mehrfachschießverfahren. Bei indirekten Verfahren werden die Steuervariablen (oder die Steuervariablen und die Zustandsvariablen) diskretisiert und das unendlich dimensionale Problem in ein endlich dimensionales nichtlineares Optimierungsproblem transformiert. Beispiele für direkte Verfahren sind direkte Kollokationsverfahren oder Mehrfachschießsverfahren. Optimierungsprobleme können manuell diskretisiert und in algebraischen Modellierungssprachen (AMPL, GAMS, Pyomo) modelliert werden. Für den Anwender ist es jedoch einfacher, speziell für dynamische

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

Optimierung entwickelte Tools zu verwenden. Das Problem inklusive Systemmodel kann dann in der ursprünglichen Form formuliert werden und das Tool übernimmt die Diskretisierung. Diese Tools können entweder auf General Purpose Languages (C++, Matlab) oder auf Komponenten-basierter Modellierung (APMonitor, gPROMS, Modelica-basierte Tools) aufbauen [2]. Der große Vorteil komponentenbasierter Modellierung liegt in der einfacheren Bedienung für Anwender.

3 Modelica

Modelica ist eine freie, objektorientierte, gleichungsbasierte Sprache für die komponentenbasierte Modellierung physikalischer Systeme, die von verschiedenen Simulationsumgebungen genutzt wird. Gegenüber kausalen Modellierungssprachen (z.B. Simulink), ergeben sich einige Vorteile. Modelica unterstützt die akausale Modellierung auf Basis von impliziten DAE Gleichungen. Die akausale Modellierung erhöht die Wiederverwendbarkeit von Modellen im Vergleich zu Modellen, die in kausalen Sprachen modelliert sind, in denen die Input-Output Kausalität fixiert ist. In Modelica können sowohl kausale als auch akausale Modelle erstellt werden. Weiteres ermöglicht Modelica die Modellierung von Multi-Domain Systemen; dies ist im Bereich von Energiesystemen von Bedeutung, da viele konzeptionelle Überlegungen für zukünftige Energiesysteme auf der sektoralen Kopplung (Storm-Wärme-Gas) basieren. Die Modelica Standard Library ist eine freie Bibliothek, die von der Modelica Association entwickelt wird und Komponenten für verschiedene Domains zur Verfügung stellt. Im Bereich von Energiesystemen gibt es spezielle Bibliotheken für Gebäude, Stromnetze, thermische Anwendungen, etc. [3].

3.1 Modelica basierte dynamische Optimierung

Ursprünglich wurde Modelica für simulations-basierte Analysen ohne eine direkte Unterstützung für Optimierung entwickelt. Um Modelica-basierte Optimierung zu ermöglichen, wurde die Spracherweiterung Optimica eingeführt [1], welche unter anderem die Formulierung von Zielfunktionen und Nebenbedingungen erlaubt. Aufbauend auf Optimica wurde die Spracherweiterung Dynamic Optimization Modeling Language (DOML) eingeführt [4]. Die meisten Methoden für Modelica basierte Optimierung erfordern, dass die Gleichungen 1a-1e zweifach kontinuierlich differenzierbar sind; Diskontinuitäten können in manchen Fällen durch differenzierbare Annaeherungen ersetzt werden. Abbildung **1** illustriert die Modelica-basierte Optimierung mit Optimica.

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

$$\underset{x,y,u,p}{\text{minimize}} \quad \Phi(t_f, x(t_f)) + \int_{t_0}^{t_f} L(x(t), y(t), u(t))dt$$

Allgemeine Form der Zielfunktion

$$s.t.\, F(\dot{x}(t), x(t), y(t), u(t)) = 0,$$

"Systemdynamik " beschrieben durch ein implizites System von DAEs.

$$h(\dot{x}(t), x(t), y(t), u(t), p) \leq 0$$
$$g(\dot{x}(t), x(t), y(t), u(t), p) = 0$$
$$H(x(t_f), y(t_f), p) \leq 0$$

Nebenbedingungen

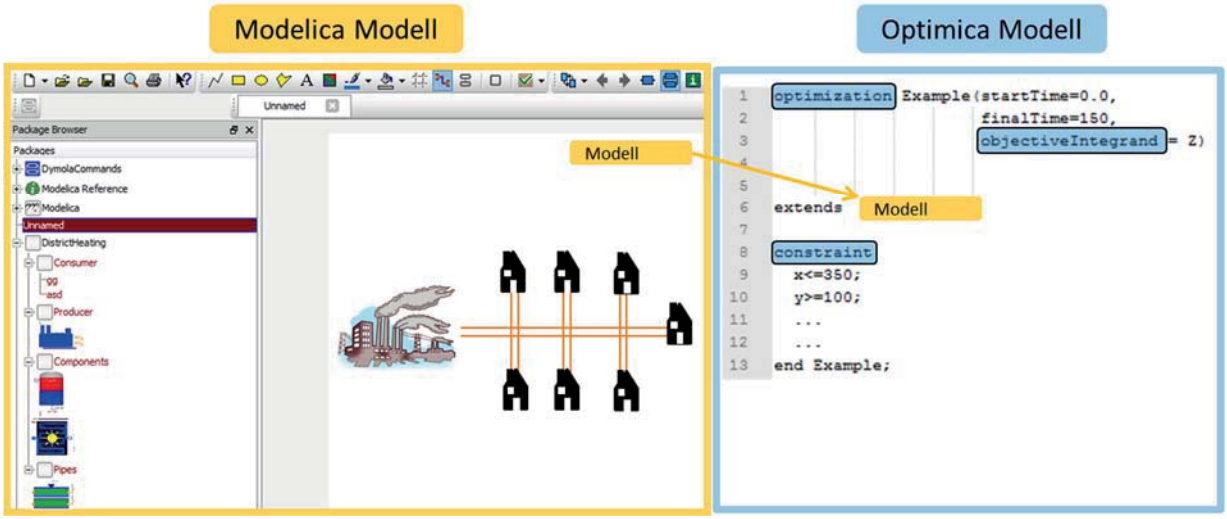

Abbildung 1: Dynamische Optimierung in Modelica und Optimica. In Gelb: das Modelica Model. In Blau: Formulierung des Optimierungsproblems in Optimica. Das Modelica Modell kann direkt in die Optimierung (in das Optimica Modell) übernommen werden (dargestellt durch den gelben Pfeil).

Der große Vorteil von Modelica-basierter Optimierung liegt darin, dass Modelle in einer graphischen, aukausalen komponentenbasierten Modellierungssprache modelliert und direkt in die Optimierung (in das Optimica Modell) übernommen werden können (gelb markiert in Abbildung 1). Zusätzlich zu dem Modell müssen die Zielfunktion und optional Nebenbedingungen im Optimica Modell definiert werden (blau markiert in Abbildung 1).
Optimica wird aktuell von JModelica und OpenModelica unterstützt. Das Open Source Tool JModelica verwendet Python für Skripting und Prototyping; dem Anwender wird keine graphische Modellierungsumgebung zur Verfügung gestellt [5]. Optimica wird vom Optimica-Compiler unterstützt, der Teil von JModelica ist. JModelica hat eine Schnittstelle zu dem Tool CasADi [6], welches (i) eine symbolische Repräsentation des Optimierungsproblems und (ii) die benötigten ersten und zweiten Ableitungen liefert. Diese Repräsentation wird dann verwendet, um das unendlich-dimensionale Problem mittels Kollokation zu diskreditieren; das Ergebnis ist ein nichtlineares Optimierungsproblem (NLP). Das NLP wird dann mittels eines Interior-Point Algorithmus (IPOPT [7]) gelöst; da viele Probleme im Bereich der Energiesystemoptimierung nicht konvex sind, kann kein globales Optimum garantiert werden. Optimica Compiler Toolkit ist ein kommerzielles Tool, das auf JModelica basiert und einige zusätzliche Funktionen aufweist. Durch die Unterstützung von verschlüsselten Bibliotheken können Modelle basierend auf kommerziellen Bibliotheken optimiert werden [8]. Das Open Source Tool OpenModelica hat eine graphische Modellierungsumgebung sowie

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

ein Python-Interface. Der OpenModelica Compiler wurde erweitert, um Optimica zu unterstützen. Für die dynamische Optimierung steht dem Anwender einerseits die Toolchain OpenModelica – CasADi – IPOPT zur Verfügung, die der Toolchain in JModelica ähnlich ist [9] und andererseits die Toolchain OpenModelica - ADOL-C – IPOPT [10]. Dabei werden die notwendigen Ableitungen mit Hilfe von ADOL-C [11] ermittelt; ADOL-C wurde in OpenModelica integriert. IDOS ist eine online-verfügbare Umgebung, die den Service bietet, optimale Steuerungsprobleme zu lösen. Der IDOS-Server unterstützt die Formulierung von Optimalsteuerungsproblemen basierend auf DOML [12]. Die DOML-Compilerumgebung basiert sehr stark auf dem Optimica-Compiler von JModelica und wurde um zusätzliche Funktionalitäten erweitert. Der IDOS-Server hat mehrere Algorithmen zur Lösung von Optimalsteuerungsproblemen integriert: Mehrfachschießsverfahren, Methoden die auf einer apriori Diskretisierung der Systemgleichungen basieren und das Problem anschließend in ein nichtlineares Programmierproblem transformieren oder Methoden, die adjungierten Gleichungen verwenden, um reduzierte Gradienten auszuwerten.

3.1.1 Direkte Kollokation

Die numerische Lösung des dynamischen Optimierungsproblems erfolgt in den meisten Modelica Tools mit dem Kollokationsverfahren, welches im Folgenden dargestellt wird.
In einem ersten Schritt wird der Zeithorizont in n_e Elemente unterteilt. Die Zeit in element i ist normalisiert:

$$\tilde{t_i}(\tau) = t_{i-1} + h_i\tau, \quad \forall\tau \in [0,1]$$

$\tilde{t_i}(\tau)$ ist die nicht-normalisierte Zeit, τ ist die normalisierte Zeit und t_i ist die right boundary des Elements i. Die variablen $\dot{x}, x, y$ and u werden mittels sogenannter Kollokations-Polynome in der lokalen Zeit τ unter Verwendung von n_c Kollokationspunkten als Interpolationspunkte approximiert. Diese Polynome basieren auf den Lagrange Grundpolynomen, wobei die Kollokationspunkte als Interpolationspunkte verwendet werden; die Kollokationspolynome sind demnach:

$$x_i(\tau) = \sum_{k=0}^{n_c} x_i\, \tilde{l_k}(\tau), \quad y_i(\tau) = \sum_{k=1}^{n_c} y_i\, l_k(\tau), \quad u_i(\tau) = \sum_{k=1}^{n_c} u_i\, l_k(\tau)$$

$\tilde{l_k}$ ist das Lagrange Grundpolynom inclusive dem ersten Punkt, um kontinuierliche Zustandsvariablen zu garantieren, $l_k$ist das Lagrange Grundpolynom ohne ersten Punkt. Das Grundpolynom ist wie folgt definiert:

$$\tilde{l_k}(\tau) = \prod_{\substack{l=0,\dots,n_c \\ l\neq k}} \frac{\tau - \tau_l}{\tau_k - \tau_l}, \quad l_k(\tau) = \prod_{\substack{l=1,\dots,n_c \\ l\neq k}} \frac{\tau - \tau_l}{\tau_k - \tau_l}$$

τ_k sind die Kollokationspunkte; es gibt verschiedene Methoden um die Kollokationspunkte zu wählen (Radau, Gauss, Lobatto, etc.), jeweils mit verschiedenen numerischen Eigenschaften. Die Polynom-Approximierung von $\dot{x}$ ist durch die Ableitung von x defniert:

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

$$\dot{x}_\iota(\tau) = \frac{dx_\iota}{d\widetilde{t_\iota}}(\tau) = \frac{d\tau}{d\widetilde{t_\iota}}\frac{dx_\iota}{d\tau}(\tau) = \frac{1}{h_\iota}\sum_{k=0}^{n_c} x_{\iota,k}\frac{d\widetilde{l_k}}{d\tau}(\tau)$$

Das ursprüngliche DAE wird dadurch in NLP Nebenbedingungen transformiert:

$$F\left(t_{i,k}, \dot{x}_{i,k}, x_{i,k}, y_{i,k}, u_{i,k}, p\right) = 0, \quad \forall i \in [1..n_e], \ \forall k \in [1..n_c]$$

Das ursprüngliche unendlich-dimensionale Optimierungsproblem ist somit zu einem endlich-dimensionalen Problem reduziert worden und die Lösung des NLP ist eine approximierte Lösung des ursprünglichen dynamischen Optimierungsproblems.

3.2 Bisherige Arbeiten im Bereich von Energiesystemen

[13] präsentiert den optimalen Einsatz eines elektrochemischen Speichers im Zusammenspiel mit einer MV Photovoltaik Anlage implementiert in JModelica. Die Autoren weisen darauf hin, dass traditionelle Ansätze dieses Problem mit einer Kombination aus linearer (gemischt-ganzzahliger) Optimierung und RES/load Prognose-Tools lösen. Diese Ansätze sind jedoch nicht in der Lage, die Nichtlinearitäten des Speichermodells zu berücksichtigen; die detaillierte Modellierung des Speichers ist jedoch enorm wichtig, um die Lebensdauer von Speichern abzuschätzen. Input sind stündliche Strompreise (Kauf und Verkauf), Prognose der Photovoltaik Produktion und die Prognose des lokalen Verbrauchs sowie die Kapazität und der Ladezustand des Speichers. Basierend auf der Maximierung des täglichen Profits wird der Energieaustausch des Speichers in einem 15min Intervall berechnet. [14] präsentiert den optimalen Start eines GuD-Kraftwerk implementiert in JModelica. Dabei wird das Spannungsniveau der Dampfturbinenwelle als Nebenbedingung berücksichtigt. Das Ziel besteht in der Erhöhung der Verfügbarkeit und Flexibilität solcher Anlagen ohne die Lebensdauer der Dampfturbine zu reduzieren. Des Weiteren wurde die Änderungsrate der Gasturbinenlast beschränkt.

[15] demonstriert den optimalen Start eines Dampfkessels. Das Ziel der in JModleica.org implementierten Optimalsteuerung besteht darin, die Zeit zu miniminieren, bis sich ein Referenzdruck sowie ein Referenz-Dampfstrom eingestellt haben. Nebenbedingung ist u.a. die Wärmebelastung im Boiler. Resultate zeigen, dass sich der Aufwand bei Modelica-basierten Implementierungen im Vergleich zu individuellen, fallspezifischen Lösungen enorm reduziert. [16], [17] präsentieren einen Modelica-basierten Framework zur dynamischen Optimierung und Simulation von zukünftigen Fernwärmenetzen implementiert in Python und Optimica Compiler Toolkit. Es wird (i) eine Modelica Bibliothek für Simulation, (ii) eine Modelica Bibliothek für Optimierung, (iii) eine Methode um sogenannten Mixed-Integer-Optimal-Control Probleme zu lösen sowie (iv) mehrere Fallstudien präsentiert. Die Zielfunktion der dynamischen Optimierung besteht in der Minimierung der Versorgungstemperatur sowie des Differenzdruckes am Erzeuger unter Einhaltung von minimalen und maximalen Versorgungstemperaturen und Differenzdrücken an den Abnehmerstationen.

Die Integration von modellprädiktiven Regelungen in Gebäuden rentiert sich aktuell aufgrund des Zeitaufwands für die Modellerstellung nur für sehr große Gebäude. Um die

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

Implementierung zu erleichtern, präsentiert [18] die Entwicklung eines Open Source Frameworks für modellprädiktive Regelungen in Gebäuden. Der Framework generiert automatisch Code für Energie- und Energiekostenoptimierung sowie für Parameterschätzungen basierend auf gegebenen Modelica Modellen, Problemspezifikationen und Input Daten. Anschließend wird JModelica verwendet, um das daraus resultierende Optimierungsproblem zu lösen. [19] präsentiert die Vorteile von Modelica basierter Lösung von nichtlinearen Optimalsteuerungen eines Gebäudes im Vergleich zu konventionellen Methoden. Dabei wird die Kühlung, die Heizung und der Energieverbrauch des Ventilators minimiert; die Temperatur und der Massenstrom sind durch obere und untere Grenzen limitiert. Das Optimerungsproblem wurde in JModelica implementiert und mit einer ableitungsfreien Simulations-basierten Optimierung (Nelder–Mead Algorithmus) verglichen. Die Lösung basierend auf der Kollokationsmethode war um den Faktor 2200 schneller. In [20] wurde eine Toolbox für die Modellierung und Identifikation von grey-box Gebäudemodell in Modelica und JModelica entwickelt. Das Gebäudemodell wurde in Modelica modelliert, während die Parameterschätzung in JModelica und Python gelöst wurde. In [21] wurde eine modellprädiktive Regelung für ein Bürogebäude in Modelica und JModelica implementiert; dabei wurde die Parameterschätzung aus [22] verwendet. Dabei konnte gezeigt werden, dass eine modellprädiktive Regelung im Vergleich zu einer konventionellen Regelung 30% der Energie einsparen kann. In [23] wird Modelica für die Modellierung und Optimierung einer Campus-Kaltwasseranlage verwendet. Die Solltemperatur des Kondensatorkühlwassers in der Anlage wird täglich optimiert, um die Summe des Energieverbrauchs von Kältemaschine und Kühlturm zu minimieren. Der Betrieb der Anlage wird in Modelica modelliert, wobei die Kühllasten des Campus mit Hilfe eines Regressionsmodells vorhergesagt und die Optimierung mit GenOpt durchgeführt werden.

4 Empirische Untersuchung

Im Zuge der laufenden empirischen Studie zum Thema „Modelica-based optimization: state of the art and future challenges“ [24] werden Möglichkeiten, Vorteile, Schwächen, aktuelle Barrieren analysiert sowie notwendige Forschungs- und Entwicklungsarbeiten identifiziert. Die zugrundeliegende Methode für die empirische Untersuchung ist eine adaptierte Delphi Methode. Die Delphi Methode ist ein mehrstufiges Befragungsverfahren von Experten und sie dient dazu, zukünftige Entwicklungen, Trends, etc. bestmöglich vorherzusagen [25], [26]. Für diese vorliegende empirische Studie wurden die Delphi Studie in zwei Runden unterteilt. Die erste Runde besteht aus einem Mix aus offenen und geschlossenen Fragen. Die zweite Runde beinhaltet ausschließlich geschossene Fragen, die basierend auf den Resultaten der ersten Runde definiert werden.

4.1 Vorteile und Stärken von Modelica-basierter dynamischer Optimierung

Studien haben gezeigt, dass die Optimalsteuerung oder modellprädiktive Regelung im Vergleich zu konventionellen Regelstrategien Verbesserungen erzielt, welche jedoch im Vergleich zum Aufwand für die Entwicklung dieser bewertet werden müssen. Bei der Entwicklung von komplexen Regelungen beträgt die benötigte Zeit für die Modellerstellung bis zu 80% [27], [28]. Damit sich komplexe Regelungen wirtschaftlich rentieren, werden Ansätze benötigt, die eine schnelle Iteration zwischen der Modellierung, der Definition des

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

Optimierungsproblems sowie dessen Lösung und Evaluierung ermöglichen. Hier kommen die Stärken von gleichungsbasierten, akausalen Modellierungssprachen wie Modelica zur Geltung. Modelica ermöglicht Rapid-Prototyping von physikalischen, domainübergreifende Systemen, Modelle können wiederverwendet und einfach adaptiert werden.

Gleichungsbasierte Sprachen unterstützen eine automatische Konvertierung von Simulationsmodellen in Optimierungsprobleme; diese Sprachen ermöglichen eine symbolische Manipulation und Code Erzeugung um effizient Gradienten und anderen Informationen zu berechnen, welche von NLP Solver verwendet werden können. Des Weiteren kann das unendlich dimensionale Problem automatisch durch Diskretisierungsmethoden in ein endlich dimensionales Problem überführt werden. Hochentwickelte numerische Optimierungsalgorithmen haben oftmals APIs, die nicht den Anforderungen vieler Anwender nach high-level Programmiersprachen genügen; viele State of the ART Tools basieren auf Programmiersprachen wie C++ oder Fortran. Modelica-basierte Tools ermöglichen eine high-level Formulierung des dynamischen Optimierungsproblems, welches automatisch in ein endlich-dimensionales NLP Problem überführt und mittels State of the Art NLP Solver gelöst wird.

4.2 Aktuelle Barrieren für Modelica-basierte dynamische Optimierung

Modelica Tools stellen nur wenige Solver für die Lösung dynamischer Optimierungsprobleme zur Verfügung und dadurch ist deren Anwendung auf bestimmte Klassen von Problemen beschränkt. Die Solver, die in JModelica und OpenModelica implementiert sind, können nicht für Optimierungsprobleme basierend auf hybriden Modellen angewandt werden. [29] präsentiert eine Prototyp-Lösung für Optimalsteuerprobleme, welche durch hybride Systeme definiert und durch higher-index DAEs beschrieben werden. Die Definition des Problems erfolgt mit DOML. Aktuell gibt es wenige Modelica Bibliotheken, die speziell für dynamische Optimierung entwickelt wurden. Modelle, die auf Basis von normalen Bibliotheken entwickelt werden, können völlig ungeeignet für die Optimierung sein, da die Modellgleichungen oftmals nicht zweifach kontinuierlich differenzierbar sind. Viele Anwendungen in unterschiedlichen Disziplinen können durch sogenannte Gemischt-ganzzahlige nichtlineare Optimierungen beschrieben werden; aktuell können solche Probleme in Modelica Tools nicht definiert und gelöst werden.
Im Bereich der Skalierbarkeit von (Modelica-basierten) Optimierungsproblemen mangelt es an wissenschaftlichen Arbeiten.

4.3 Zukünftige Herausforderungen

Experten sehen großes Verbesserungspotential im Bereich der Fehlersuche und Solver Feedback der verschiedenen Tools. Ein weiterer zentraler Punkt ist die Integration von verschiedenen Solver, um den Anwendungsbereich von Modelica-basierter Optimierung zu vergrößern. Verschiedene Solver können ohne fundamentale Änderungen des Modelica Standards implementiert werden. Modelica ist grundsätzlich geeignet, um gemischt-ganzzahlige nichtlineare Optimierungsprobleme zu formulieren; Optimica könnten ebenfalls dahingehend erweitert werden. Ob Modelica Tools dahingegen weiterentwickelt werden sollten, dass nichtlineare, gemisch-ganzzahlige Probleme gelöst werden können, wird kontrovers diskutiert. Die Komplexität solcher Probleme könnte aufgrund deren schwieriger

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

Lösung limitiert und der Einsatz im Bereich von Energiesystemen dadurch auf wenige Anwendungen beschränkt sein. Damit der Einstieg für Anwender erleichtert wird, sind speziell auf die jeweilige Domain abgestimmte Bibliotheken für die Optimierung inklusive Anwendungsbeispiele notwendig. Des Weiteren würden mehr Kurse an Universitäten im Bereich dynamischer Optimierung den Einstieg für Benutzer erleichtern.

5 Conclusio

Die vorliegende Arbeit diskutiert Anwendungen, aktuelle Barrieren sowie notwendige Forschungs- und Entwicklungsarbeiten von Modelica-basierter Optimierung. Der große Vorteil im Vergleich zu anderen State of the Art Ansätzen besteht in der vereinfachten Erstellung und Adaption von Modellen sowie deren Wiederverwendbarkeit; dadurch eignet sich Modelica für Rapid Prototyping. Trotz der high-level Sprache sind die Anforderungen an den Anwender im Bereich der Optimierung signifikant höher als in der Simulation. Der Anwender muss bei der Definition des Problems die spezifische Anwendung sowie die Anforderungen und Eigenheiten des Solvers berücksichtigen. Experten gehen davon aus, dass verbesserte Tools die Barriere für Anwender aufgrund der Komplexität von Optimierungsproblemen nur bis zu einem gewissen Grad senken können. Entwicklungsbedarf gibt es vor allem bei Optimierungsbibliotheken sowie der Integration von verschiedenen Solvern, um das Einsatzgebiet von Modelica-basierter Optimierung zu vergrößern. Es werden standardisierte Fallstudien benötigt, die einen fairen Vergleich von verschiedenen Methoden ermöglichen. Grundsätzlich sehen Experten vor allem im Bereich der Optimalsteuerung und der modellprädiktiven Regelung von (hybriden) Energiesystemen für Modelica-basierte Optimierung ein großes Potential.

6 Acknowledgement

Diese Arbeit wurde unterstützt vom Assistant Secretary for Energy Efficiency and Renewable Energy, Office of Building Technologies of the U.S. Department of Energy, unter dem Vertrag Nummer DE-AC02- 05CH11231. Diese Arbeit resultierte aus dem IBPSA Project 1, einem internationalen Projekt das unter der Schirmherrschaft der International Building Performance Simulation Association (IBPSA) ausgeführt wird. Project 1 entwickelt und demonstriert BIM/GIS und Modelica Frameworks für die Planung und den Betrieb von Gebäude und Energiesystemen auf Quartiersebene.

7 Referenzen

[1] F. Magnusson, “Numerical and Symbolic Methods for Dynamic Optimization Numerical and Symbolic Methods for Dynamic Optimization Fredrik Magnusson,” PhD thesis, 2016.

[2] F. Magnusson and J. Åkesson, “Dynamic Optimization in JModelica.org,” *Processes*, 2015.

[3] G. Schweiger, G. Engel, and I. Leusbrock, “Multi-domain, open general tools for modelling urban energy systems,” *Energy (submitted)*, 2018.

[4] T. Tarnawski and R. Pytlak, “DOML - a Compiler Environment for Dynamic Optimization Supporting Multiple Solvers,” in *Proceedings of the 10th International*

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

ModelicaConference, 2014.

[5] J. Åkesson, K. Årzén, M. Gäfvert, T. Bergdahl, and H. Tummescheit, “Modeling and optimization with Optimica and JModelica.org — Languages and tools for solving large-scale dynamic optimization problems,” *Comput. Chem. Eng.*, 2010.

[6] J. Andersson, “A General-Purpose SoftwareFramework for DynamicOptimization,” PhD thesis, 2013.

[7] A. Wächter and L. T. Biegler, “On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming,” *Math. Program.*, 2006.

[8] Modelon, “OPTIMICA Compiler Toolkit,” 2017. [Online]. Available: http://www.modelon.com/products/optimica-compiler-toolkit.

[9] A. Shitahun *et al.*, “Model-Based Dynamic Optimization with OpenModelica and CasADi,” in *7th IFAC Symposium on Advances in Automotive Control*, 2013.

[10] V. Ruge, W. Braun, B. Bachmann, A. Walther, and K. Kulshreshtha, “Efficient Implementation of Collocation Methods for Optimization using OpenModelica and ADOL-C,” in *Proceedings of the 10th International ModelicaConference*, 2014.

[11] A. Walther, “Getting Started with ADOL-C,” in *Dagstuhl Seminar Proceedings* , 2012.

[12] R. Pytlak and T. Tarnawski, “IDOS -(also) a Web Based Tool for Calibrating Modelica Models *,” in *Proceedings of the 10th International ModelicaConference*, 2014.

[13] S. Barsali, R. Giglioli, G. Lutzemberger, D. Poli, and G. Valenti, “Optimal operation of storage systems integrated with MV photovoltaic plants, using Jmodelica,” in *2017 IEEE International Conference on Environment and Electrical Engineering*, 2017

[14] F. Casella, F. Donida, and J. Åkesson, “Object-Oriented Modeling and Optimal Control: A Case Study in Power Plant Start-Up,” *IFAC Proc.,* 2011.

[15] F. Belkhir, D. K. Cabo, F. Felgner, and G. Frey, “Optimal Startup Control of a Steam Power Plant Using the JModelica Platform,” *IFAC-PapersOnLine*, 2015.

[16] G. Schweiger, P. Larsson, F. Magnusson, and P. Lauenburg, “District heating and cooling systems - Framework for Modelica-based simulation and dynamic optimization,” Energy, 2017.

[17] G. Schweiger, H. Runvik, F. Magnusson, P.O. Larsson and S. Velut “Framework for dynamic optimization of district heating systems using Optimica Compiler Toolkit,” in *Proceedings of the 12th International ModelicaConference*, 2017.

[18] D. Blum and M. Wetter, “MPCPy: An Open-Source Software Platform for Model Predictive Control in Buildings.,” in *Proceedings of the 15th Conference of International Building Performance Simulation*, 21017.

[19] M. Wetter, M. Bonvini, and T. S. Nouidui, “Equation-based languages – A new paradigm for building energy modeling , simulation and optimization,” *Energy Build.*, 2016.

[20] R. De Coninck, F. Magnusson, J. Åkesson, and L. Helsen, “Toolbox for development and validation of grey-box building models for forecasting and control,” *J. Build. Perform. Simul.*, 2016.

[21] R. De Coninck and L. Helsen, “Practical implementation and evaluation of model predictive control for an office building in Brussels,” *Energy Build.*, 2016.

[22] R. De Coninck, F. Magnusson, J. Åkesson, and L. Helsen, “Toolbox for development and validation of grey-box building models for forecasting and control,” *J. Build. Perform. Simul.*, 2016.

[23] J. Granderson, M. Bonvini, M. A. Piette, J. Page, G. Lin, and L. R. Hu, “Can We

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

Practically Bring Physics-based Modeling Into Operational Analytics Tools?," in *Proceedings of the 2016 ACEE Summer Study on Energy Efficiency in Buildings*, 2016.

[24] G. Schweiger, D. Blum, and M. Wetter, "Modelica-based Optimization: – an empirical survey: State of the Art, Applications and future challenges," work in progress, 2018.

[25] N. Dalkey and O. Helmer, "An Experimental Application of the DELPHI Method to the Use of Experts," *Manage. Sci.*, 1963.

[26] C.-C. Hsu and B. A. Sandford, "The Delphi Technique: Making Sense of Consensus - Practical Assessment, Research & Evaluation," *Pract. Assessment, Res. Eval.*, 2007.

[27] J. Cigler, D. Gyalistras, J. Široký, V.-N. Tiet, and L. Ferkl, "Beyond theory: the challenge of implementing Model Predictive Control in buildings. 11th REHVA World Congress & 8th International Conference on IAQVEC, 2013"

[28] H. Thieblemont, F. Haghighat, R. Ooka, and A. Moreau, "Predictive control strategies based on weather forecast in buildings with energy storage system: A review of the state-of-the art," *Energy Build.*, 2017.

[29] R. Pytlak and D. Suski, "On solving hybrid optimal control problems with higher index DAEs On solving hybrid optimal control problems with higher index DAEs," *Optim. Methods Softw.*, 2017.