OBSERVE

Arbeitspaket AP B.4

Lernende prädiktive Regelung, Methoden und Design

Björn Lautenschlager, Kai Kruppa, Gerwald Lichtenberg
bjoern.lautenschlager @ haw-hamburg.de,
kai.kruppa @ haw-hamburg.de,
gerwald.lichtenberg @ haw-hamburg.de

Hochschule für Angewandte Wissenschaften Hamburg
Fakultät Life Sciences
Ulmenliet 20, 21033 Hamburg

Datum: 15. März 2018

Gefördert durch:

Bundesministerium
für Wirtschaft
und Energie

aufgrund eines Beschlusses
des Deutschen Bundestages

Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Inhalt

7 AP B.4 - Lernende prädiktive Regelung, Methoden und Design

7.1 Einleitung

Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.

Für die Gebäudeversorgung in Deutschland werden bereits heute über 40 % der Endenergie eingesetzt, woran sich das hohe energetische Einsparpotential im Gebäudebereich zeigt. Als Ziel für den Gebäudebereich wurde ein nahezu klimaneutraler Gebäudebestand bis zum Jahr 2050 definiert [1]. Die Ziele sollen durch die Kombination unterschiedlicher Maßnahmen erreicht werden, z.B. durch die Optimierung der Gebäudehülle durch Wärme- und Sonnenschutz oder der Integration von erneuerbaren Energien, aber auch durch die Betriebsoptimierung bereits eingebauter Systeme.

Ein Bereich stellt dabei die Optimierung von Gebäudeautomationssystemen dar und als Teil davon die Optimierung von Heizungssystemen. Diese werden in Bestandsanlagen oftmals nicht optimal geregelt, sodass durch eine Überarbeitung der bereits bestehenden Regelung, aber auch durch die Entwicklung und Anwendung neuer modellbasierter Regelungskonzepte, die vorhandenen Systeme in ihrer Effizienz gesteigert werden können und somit einen Beitrag zur Einsparung von Primärenergie leisten.

Dieses Kapitel beschäftigt sich mit der Entwicklung und der Weiterentwicklung neuartiger Regelungskonzepte am Beispiel von Gebäudeautomationssystemen. Dabei steht die modellbasierte Entwicklung eines Entwurfsverfahrens einer iterativ lernenden Regelung im Vordergrund, welches auf Heizungssysteme angewandt und getestet werden soll. Die Frage, ob die auftretenden periodischen Störungen, verursacht durch die Tages-, Wochen- und Jahresläufe der Außentemperatur und des Wärmebedarfs, durch die Anwendung eines iterativ lernenden Reglerkonzeptes berücksichtigt werden können und so zu einer wesentlichen Verbesserung der Regelung des Systems führt, ist von zentraler Bedeutung. Eine weitere Herausforderung besteht dabei in der Berücksichtigung von systematischen Totzeiten wie sie für viele Heizungssystemen typisch sind, z.B. aufgrund der Entfernung zwischen Erzeuger und Verbraucher oder Kesselspülzeiten. Diese Überlegungen legen die Nutzung einer modellprädiktiven Regelung nahe. Daraus ergibt sich die Frage, wie eine modellprädiktive Regelung für die iterativ lernende Regelung genutzt werden kann und soll in diesem Kapitel betrachtet werden. Auch die Frage, ob gespeicherte Messdaten, welche oftmals für Monitoringzwecke genutzt werden, auch direkt für die Regelung genutzt werden können und wie die Menge der zu speichernden Daten für die Nutzung auf Plattformen ohne Zugriff auf großen Speicherkapazitäten, reduzieren werden kann ist Bestandteil dieses Kapitels.

Die multilinearen Systemen habe sich als zweckmäßig für die Modellierung von Heizungssystemen erwiesen [37]. Für lineare Systeme existieren bereits effiziente Algorithmen um das Optimierungsproblem der modellprädiktiven Regelung zu lösen. Für die multilinearen Systeme gibt es diesbezüglich noch keine strukturellen Untersuchungen im Bezug auf das Optimierungsproblem. Das heißt, die Frage wie die Struktur des modellprädiktiven Optimierungsproblems eines multilinearen Systems ist, wird behandelt.

Durch die Anwendung eines EMPC (economic model predictive control) welcher eine lineare Kostenfunktion optimiert, besteht die Möglichkeit anstelle von Wichtungsfaktoren reale Kosten während der Optimierung zu berücksichtigen. Das hätte zur Folge, dass eine Anlage unter wirtschaftlichen Gesichtspunkten kosten optimal laufen wurde, in Bezug auf die in der Kostenfunktion des EMPC festgelegten Parameter. Die Frage, inwieweit sich ein solches Konzept auf die Regelung einer Heizungsanlage anwenden lässt, wird beleuchtet.

Dieses Kapitel ist wie folgt strukturiert. Im Abschnitt 7.2 werden zunächst die unterschiedlichen Modellklassen eingeführt, welche in diesem Kapitel benötigt werden, sowie die wichtigsten Modellkomponenten von Heizungssystemen. Im Abschnitt 7.3 wird eine Konvexitätsanalyse des Optimierungsproblems der modellprädiktiven Regelung für multilineare zeitinvariante Systeme durchgeführt. Anschließend wird im Unterkapitel 7.4 ein datenbasierter iterativ lernender Regler designet und gezeigt wie dieser mit einer modellprädiktiven Regelung kombiniert werden kann. Die Anwendung von Tensorzerlegunsverfahren für die Speicherbedarfsreduktion eines datenbasierten iterativ lernenden Reglers wird in 7.5 gezeigt. Gefolgt

von einer Betrachtung in Abschnitt 7.6 der modellprädiktiven Regelung wenn eine lineare Kostenfunktion genutzt wird. Anschließend wird in 7.7 die verwendete Hardware vorgestellt. Das Kapitel Endet mit dem Unterkapitel 7.8 „Zusammenfassung und Ausblick“.

7.2 Modellbildung für Heizungssysteme

Für die modellprädiktive Regelung nimmt die Modellbildung eine zentrale Rolle ein. In diesem Abschnitt werden die, für dieses Arbeitspaket relevanten Modellklassen eingeführt und anschließend auf die komponentenweise Modellbildung für Heizungssysteme eingegangen. Zunächst werden die unterschiedlichen Modellklassen in Zustandsraumdarstellung eingeführt.

7.2.1 Zustandsraummodelle

Das dynamische Verhalten eines physikalischen Systems kann mathematisch beschrieben werden. Das bedeutet, dass die Beziehung zwischen dem Eingangssignal $\mathbf{u}(t)$ und dem Ausgangssignal $\mathbf{y}(t)$ berechnet wird. Das Blockdiagramm in Abbildung 7.2-1 beschreibt das Eingangs-Ausgangsverhalten eines Systems.

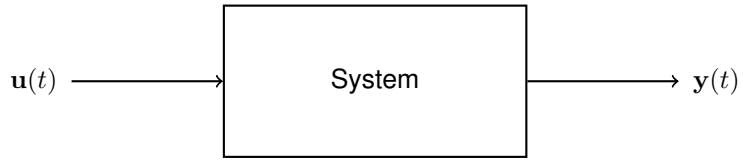


Abbildung 7.2-1: Blockdiagramm des Eingangs-Ausgangsverhalten

Eine Standardform um ein dynamisches Modell zu beschreiben sind Zustandsraummodelle (ZRM), welche in der allgemeinsten Form nichtlinear sind.

7.2.1.1 Nichtlineare Zustandraummodelle

Das dynamische Verhalten eines Systems kann für viele Anwendungen durch Differentialgleichungen (DGL) n-ter Ordnung beschrieben werden. Die Differentialgleichungen beschreiben das Eingangs-Ausgangsverhalten eines Systems in kontinuierlichen Zeitbereich. Eine DGL n-ter Ordnung kann auch als System von Differentialgleichungen erster Ordnung geschrieben werden. Im allgemeinen sind die rechten Seiten der DGLs nichtlinear. Ein System von Differentialgleichungen erster Ordnung führt zu der Beschreibung des dynamischen Systems als ein nichtlineares zeitkontinuierliches Zustandsraummodell

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \tag{7.2-1}$$

$$\mathbf{y}(t) = \mathbf{g}(\mathbf{x}(t), \mathbf{u}(t)), \tag{7.2-2}$$

$$\mathbf{x}(0) = \mathbf{x}_0, \tag{7.2-3}$$

wobei $\mathbf{f}$ die Zustandsübergangsfunktion ist, $\mathbf{g}$ die Ausgangsfunktion, $\mathbf{x}(t) \in \mathbb{R}^n$ ist der Zustandsvektor, $\mathbf{u}(t) \in \mathbb{R}^m$ ist der Eingangsvektor, $\mathbf{y}(t) \in \mathbb{R}^p$ ist der Ausgangsvektor, $\mathbf{x}_0$ ist der Vektor welcher die Startwerte enthält und t beschreibt die Zeit. Das System hat n Zustände, m Eingänge and p Ausgänge. Die Änderung der Zustände des Systems wird durch die Zustandsgleichung (7.2-1) beschrieben und die Ausgänge des Systems durch die Ausgangsgleichung (7.2-2).

Es wurde angenommen, dass alle Signale zu jeder Zeit t bekannt sind, was zu einer zeitkontinuierlichen Beschreibung des Systems führt. Für die zeitdiskrete Beschreibung sind die Zustände, Eingänge und Ausgänge des Systems nur noch zu festen Abtastzeitpunkten bekannt, zum Beispiel für einen Eingang $[u(0 \cdot t_s), u(1 \cdot t_s), ...]$ mit der Abtastzeit t_s. Für eine einfachere Schreibweise wird die Abtastzeit weggelassen $u(k \cdot t_s) = u(k)$, wobei $k \in \mathbb{N}_0$ der Index der Zeitschritte ist. Das Systemverhalten im Zeitdiskreten wird durch eine Differenzengleichung beschrieben und nicht mehr durch eine Differentialgleichung, sodass zum Zeitpunkt k die Zustände auf des Systems $\mathbf{x}(k+1)$ von den aktuellen und den vergangenen Zuständen und Eingägnen abhängen. Damit ergibt sich für das nichtlineare zeitdiskrete

Zustandsraummodell

$$\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k)), \tag{7.2-4}$$
$$\mathbf{y}(k) = \mathbf{g}(\mathbf{x}(k), \mathbf{u}(k)), \tag{7.2-5}$$
$$\mathbf{x}(0) = \mathbf{x}_0. \tag{7.2-6}$$

Die nichtlinearen Zustandsraummodelle sind die allgemeinste Modellklasse. Die Modellklasse der multilinearen Zustandsraummodelle, sowie die linearen Zustandsraummodelle stellen Unterklassen der nichtlinearen Zustandsraummodelle dar.

7.2.1.2 Multilineare Zustandsraummodelle

Die multilinearen zeitinvarianten (MTI - multilinear time-invariant) Zustandsraummodelle wurden in [28] eingeführt. Die Systembeschreibung durch nichtlineare Zustandsraummodelle erlaubt belibige Funktionen als rechte Seiten, wohingegen die rechten Seite der Differentialgleichungen eines multilinearen Zustandsraummodels multilineare Funktionen sein müssen.

Als erstes wird der Monomvektor für die Definition der multilinearen Zustandsraummodelle eingeführt.

Definition 7.2.1 *Der Monomvektor ist definiert als*

$$\mathbf{m}(\mathbf{x}(t), \mathbf{u}(t)) = \begin{pmatrix} 1 \\ u_m \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} 1 \\ u_1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ x_n \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} 1 \\ x_1 \end{pmatrix}$$

$$= \left(\bigotimes_{i=m}^{1} \begin{pmatrix} 1 \\ u_i \end{pmatrix} \right) \otimes \left(\bigotimes_{i=m}^{1} \begin{pmatrix} 1 \\ x_i \end{pmatrix} \right), \tag{7.2-7}$$

wobei $\mathbf{x} \in \mathbb{R}^n$ mit den Elementen x_i, $i = 1, \ldots, n$ der Zustandsvektor ist und $\mathbf{u} \in \mathbb{R}^m$ mit den Elementen u_j, $j = 1, \ldots, m$ der Eingangsvektor ist und $\otimes$ ist das Kronecker-Produkt. Für eine Folge von Kronecker-Produkten verändertt sich der Index um eins in jedem Schritt vom unteren zum oberen Index, [28].

Beispiel 7.2.1 *Monomvektor mit zwei Zuständen, x_1 und x_2 und einem Eingang u_1*

$$\mathbf{m}(\mathbf{x}, \mathbf{u}) = \begin{pmatrix} 1 \\ u_1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ x_2 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ x_1 \end{pmatrix} = \begin{pmatrix} 1 \\ x_1 \\ x_2 \\ x_1 x_2 \\ u_1 \\ u_1 x_1 \\ u_1 x_2 \\ u_1 x_1 x_2 \end{pmatrix}.$$

Das Zustandsraummodell eines kontinuierlichen MTI Systems mit n Zuständen, m inputs und p Ausgängen in Matrixdarstellung ist gegeben durch

$$\dot{\mathbf{x}}(t) = \mathbf{F}\mathbf{m}(\mathbf{x}(t), \mathbf{u}(t)) \tag{7.2-8}$$
$$\mathbf{y}(t) = \mathbf{G}\mathbf{m}(\mathbf{x}(t), \mathbf{u}(t)) \tag{7.2-9}$$

mit der Übergangsmatrix $\mathbf{F} \in \mathbb{R}^{n \times 2^{n+m}}$ und der Ausgangsmatrix $\mathbf{G} \in \mathbb{R}^{p \times 2^{n+m}}$.

Das zeitdiskrete Zustandsraummodell eines MTI Systems ist gegeben durch

$$\mathbf{x}(k+1) = \mathbf{F}\mathbf{m}(\mathbf{x}(k), \mathbf{u}(k)) \,, \tag{7.2-10}$$
$$\mathbf{y}(k) = \mathbf{G}\mathbf{m}(\mathbf{x}(k), \mathbf{u}(k)) \,, \tag{7.2-11}$$

dabei ist $\mathbf{x}$ der Zustandsvektor, $\mathbf{u}$ der Eingangsvektor, $\mathbf{m}$ der Monomvektor, $\mathbf{y}$ der Ausgangsvektor, $\mathbf{F} \in \mathbb{R}^{n \times 2^{n+m}}$ die Übergangsmatrix und $\mathbf{G} \in \mathbb{R}^{p \times 2^{n+m}}$ die Ausgangsmatrix.

Beispiel 7.2.2 *Die Zustandsübergangsgleichung eines Modells zweiter Ordnung ist gegeben durch*

$$\begin{pmatrix} x_1(k+1) \\ x_2(k+1) \end{pmatrix} = \begin{pmatrix} f_{11} & f_{12} & f_{13} & f_{14} & f_{15} & f_{16} & f_{17} & f_{18} \\ f_{21} & f_{22} & f_{23} & f_{24} & f_{25} & f_{26} & f_{27} & f_{28} \end{pmatrix} \begin{pmatrix} 1 \\ x_1(k) \\ x_2(k) \\ x_1(k)x_2(k) \\ u(k) \\ u(k)x_1(k) \\ u(k)x_2(k) \\ u(k)x_1(k)x_2(k) \end{pmatrix}.$$

Für eine ausführliche Beschreibung der multilinearen zeitinvarianten Systeme, siehe auch Kapitel 6 „AP A.4: Dezentrale Netzwerkregelung, Methoden und Design“ im Abschnitt 6.1 „Modellierung von Heizungsanlagen mit multilineare Modelle“.

7.2.1.3 Lineare Zustandsraummodelle

Das dynamische Verhalten eines linearen Systems wird durch eine lineare Differentialgleichung beschrieben, was bedeutet, dass die rechten Seiten lineare Funktionen sind. Damit folgt für das lineare Zustandsraummodell

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), \tag{7.2-12}$$
$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \tag{7.2-13}$$

mit der Systemmatrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, der Eingangsmatrix $\mathbf{B} \in \mathbb{R}^{n \times m}$, der Ausgangsmatrix $\mathbf{C} \in \mathbb{R}^{p \times n}$ und der Durchgangsmatrix $\mathbf{D} \in \mathbb{R}^{p \times m}$.

Für das zeitdiskrete lineare Zustandsraummodell folgt

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k), \tag{7.2-14}$$
$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k) \tag{7.2-15}$$

Beispiel 7.2.3 *Die Zustandsgleichung eines linearen Zustandsraummodells mit zwei Zuständen, einem Eingang und zwei Ausgängen sieht wie folgt aus*

$$\begin{pmatrix} x_1(k+1) \\ x_2(k+1) \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} x_1(k) \\ x_2(k) \end{pmatrix} + \begin{pmatrix} b_{11} \\ b_{21} \end{pmatrix} u(k).$$

Bei manchen Anwendungen sind die Eingangsgrößen diskret und nicht kontinuierlich. Zum Beispiel kann ein Eingangssignal nur die Werte null und eins annehmen, also an und aus geschaltet werden. Aus diesem Grund werden im folgenden die sogenannten hybriden Zustandsraummodelle, mit diskreten und kontinuierlichen Eingangssignalen eingeführt.

7.2.1.4 Hybride Zustandsraummodelle

Das hybride Modell, welches in diesem Kapitel verwendet wird, ist ein lineares Zustandsraummodell mit kontinuierlichen und diskreten Eingangssignalen. Die Beschreibung eines linearen Zustandsraummodells wie es in Abschnitt 7.2.1.3 eingeführt wurde, wird um einen zusätzlichen Eingangsekttor $\mathbf{u}_{dis}$ erweitert, welcher die diskreten Eingangssignale enthält. Die Werte des Vektors $\mathbf{u}_{dis}$ müssen in der Menge der natürlichen Zahlen $\mathbb{N}_0$ liegen. Das führt auf das zeitkontinuierliche hybride lineare Zustandsraummodell

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{B}_{dis}\mathbf{u}_{\mathbf{dis}}(t), \tag{7.2-16}$$
$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) + \mathbf{D}_{dis}\mathbf{u}_{\mathbf{dis}}(t) \tag{7.2-17}$$

mit der Eingangsmatrix $\mathbf{B} \in \mathbb{R}^{n \times m_{dis}}$ und dem Eingangsvektor $\mathbf{u}_{dis} \in \mathbb{N}_0^{m_{dis}}$ der diskreten Signale und der Anzahl der diskreten Eingänge m_{dis}.

Das entsprechende zeitdiskrete Zustandsraummodell ist gegeben durch

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) + \mathbf{B}_{dis}\mathbf{u}_{\mathbf{dis}}(k), \tag{7.2-18}$$

$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k) + \mathbf{D}_{dis}\mathbf{u}_{\mathbf{dis}}(k). \tag{7.2-19}$$

Diese Modellklassen können für die Modellierung von Heizungssystemen genutzt werden.

7.2.2 Modellbildung von Heizungssystemen

Für die Modellbildung von Heizungssystemen hat sich in dem Projekt ModQS eine komponentenweise Modellbildung als zielführend herausgestellt, da Heizungssysteme für jedes Gebäude individuell geplant werden und sich somit in den meisten Fällen von einander unterscheiden. Je nach Anforderung hat jedes Gebäude seine eigenen Kombination von verbauten Komponenten, aber die einzelnen Komponenten tauchen immer wieder auf, z.B. der Kessel welcher als Wärmeerzeuger in vielen Gebäuden zu finden ist. Das heißt, wenn einzelne Komponenten modelliert werden, können diese in unterschiedlichen Zusammenhängen wieder genutzt werden. Einige Komponenten wurden in dem Projekt ModQS entwickelt und für MATLAB/Simulink in einer „Heizungsbibliothek“ der HeatLib zusammengefasst, [23]. Im folgend wird die Modellbildung der Hauptkomponenten, wie ein Wärmeerzeuger in Form eines Kessels, der Verbraucher mit den Heizkörpern und dem Gebäude, sowie einer Pumpe vorgestellt. Dabei basiert die Modellbildung auf Wärmeleistungsbilanzen und führt auf Differentialgleichungen welche das dynamische Verhalten der jeweiligen Komponente beschreiben.

7.2.2.1 Kessel

Der Kessel deckt der Wärmebedarf $\dot{Q}_{ab}$ des Verbrauchers. Die Änderung der Vorlauftemperatur des Kessels $T_{vl,k}$ wird durch die Differentialgleichung

$$\dot{T}_{vl,k}(t) = \frac{\dot{V}(t)\left(T_{rl,k}(t) - T_{vl,k}(t)\right)}{V_{kessel}} + \frac{P_{in}(t) - k_{kessel}\left(T_{vl,k}(t) - T_u(t)\right)}{c\rho V_{kessel}} \tag{7.2-20}$$

beschrieben. Dabei ist V_{kessel} das Kesselvolumen, $T_{rl,k}$ die Rücklauftemperatur des Kessels, $\dot{V}$ der Volumenstrom, T_u die Umgebungstemperatur des Kessels, k_{kessel} der Wärmeübergangskoeffizient vom Kessel zur Umgebung, P_{in} ist die Wärmeleistung die das Wasser erwärmt und c und ρ sind die spezifische Wärmekapazität und die Dichte von Wasser. Die zugeführte Wärmeleistung $P_{in} = \alpha P_{max}$ wird durch das Modulationssignal $\alpha \in [0, 1]$ geregelt, wobei P_{max} die maximale Leistung des Kessels ist.

7.2.2.2 Verbraucher

Der Verbraucher besteht aus dem Gebäude mit der Raumtemperatur T_{raum} und den Heizkörpern mit der Rücklauftemperatur $T_{rl,h}$. Es wird angenommen, dass der Wärmeübergang von den Heizkörpern zum Gebäude bzw. zum Raum proportional zur Temperaturdifferenz zwischen dem Heizkörper mit der Temperatur $T_{rl,h}$ und dem Raum mit der Temperatur T_{raum} ist. Außerdem wird angenommen, dass die Wärmeverluste des Gebäudes proportional zur Differenz der Raumtemperatur T_{raum} und der Außentemperatur T_a sind.

Der Kessel erwärmt das Wasser welches mit der Vorlauftemperatur $T_{vl,k}$ und dem einem konstanten Volumenstrom $\dot{V}$ die Heizkörper versorgt, welcher die Wärme an den Raum abgibt. Unter der Annahme, dass das Wasser im Heizkörper instantan vollständig durchmischt ist, verlässt das kalte Wasser den Heizkörper wieder mit der Rücklauftemperatur $T_{rl,h}$. Beschrieben durch die Differentialgleichung

$$\dot{T}_{rl,h}(t) = \frac{\dot{V}(t)(T_{vl,k}(t) - T_{rl,h}(t))}{V_v} - \frac{k_{h,r}(T_{rl,h}(t) - T_{raum}(t))}{c\rho V_v}, \tag{7.2-21}$$

mit dem gesamten Verbrauchervolumen V_v und dem Wärmeübergangskoeffizienten $k_{h,r}$ von den Heizkörpern zum Gebäude. Die Raumtemperatur ist gegeben durch die Differentialgleichung

$$\dot{T}_{raum}(t) = \frac{k_{h,r}(T_{rl,h}(t) - T_{raum}(t)) - k_{r,a}(T_{raum}(t) - T_a(t))}{C_r}, \tag{7.2-22}$$

mit der Wärmekapazität des Gebäudes C_r und dem Wärmeübergangskoeffizienten vom Gebäude nach außen $k_{r,a}$.

7.2.2.3 Pumpe

Die Pumpe legt den Volumenstrom $\dot{V}$ in Abhängigkeit der Differenz zwischen der Gebäudetemperatur T_{raum} und gewünschten Gebäudetemperatur $T_{raum,ref}$ fest. Dieses Verhalten tritt auf wenn Thermostatventile genutzt werden. Für große Unterschiede zwischen T_{raum} und $T_{raum,ref}$ ändert sich der Volumenstrom $\dot{V}$ nicht mehr und der Volumenstrom ist in Sättigung. Es wird angenommen, dass die Differenz nicht zu groß ist, sodass die Abhängigkeit linear ist

$$\dot{V} = \dot{V}_{mean} + \beta(T_{raum,ref} - T_{raum}), \tag{7.2-23}$$

mit dem mittleren Volumenstrom $\dot{V}_{mean}$ und der Steigung β.

7.2.2.4 Drei-Wege-Ventil

Ein Drei-Wege-Ventil mischt zwei Volumenströme $\dot{V}_1$ und $\dot{V}_2$ mit den Temperaturen T_1 und T_2, z.B. um die Vorlauftemperatur eines Heizkreises zu reduzieren in dem Wasser aus dem Rücklauf dem Vorlauf beigemischt wird. Wenn das Mischungsverhältnis

$$\phi = \frac{\dot{V}_2}{\dot{V}_1} \tag{7.2-24}$$

bekannt ist, kann der resultierende Volumenstrom $\dot{V}_m$ eines Drei-Wege-Ventils berechnet werden

$$\dot{V}_m = \dot{V}_1 + \dot{V}_2 = (1 + \phi)\,\dot{V}_1 \tag{7.2-25}$$

und die Temperatur ergibt sich zu

$$T_m = \frac{T_1\dot{V}_1 + T_2\dot{V}_2}{\dot{V}_1 + \dot{V}_2} = \frac{1}{1+\phi}\,(T_1 + \phi T_2) \tag{7.2-26}$$

Ein Mischungsverhältnis von null bedeutet dass $\dot{V}_1 = \dot{V}_m$ und nichts wird dem Volumenstrom beigemischt $\dot{V}_1$.

Die einzelnen Modellkomponenten werden in den folgenden Abschnitten für unterschiedliche Modelle genutzt.

7.3 Konvexitätsanalyse der prädiktiven Regelung für MTI Systeme

Für die modellprädiktive Regelung ist die Abbildung des dynamischen Verhaltens des Systems mit Hilfe eines Modells unumgänglich. Bei Anwendungen, wie bei Heizungsnalagen oder chemischen Prozessen, bei denen die Systeme mit Hilfe von Energie- oder Wärmeleistungsbilanzen modelliert werden, treten oft Nichtlinearitäten auf. Um das dynamische Verhalten solcher Systeme näherungsweise wiederzugeben eignen sich die MTI Systeme und bilden eine zweckmäßige Klasse für die Modellierung von Heizungssystemen [37]. Auch stellen MTI Modelle eine gute Approximation für nichtlineare Modelle dar, wenn eine Näherung mit einem linearen Modell nicht mehr ausreichend genau ist, [24].

Die modellprädiktive Regelung unter Verwendung eines linearen Modells wie sie in Kapitel 6 „AP A.4: Dezentrale Netzwerkregelung, Methoden und Design“ im Abschnitt 6.4.1 „Lineare modellprädiktive Regelung“ eingeführt wurde, ist in vielen Veröffentlichungen untersucht worden, [35]. Auch in Anwendung auf Heizungssysteme wurde die modellprädiktive Regelung untersucht, z.B. in [19, 36, 20, 40]. Die Verwendung linearer Modelle in Verbindung mit einer quadratischen Kostenfunktion führen zu einem konvexen Optimierungsproblem, welches mit standard Algorithmen effizient gelöst werden kann, [30]. Müssen nichtlineare Effekte mit berücksichtigt werden, da eine lineare Beschreibung nicht mehr ausreichend genau ist, führt dies zu einem nichtlinearen Modell, welches für den MPC genutzt wird. Die nichtlineare modellprädiktive Regelung führt im allgemeinen zu sehr hohem Rechenaufwand, auf Grund des nicht konvexen Optimierungsproblemens. Für die Klasse der MTI Systeme fehlen noch strukturelle Untersuchungen des modellprädiktiven Optimierungsproblems. Der folgende Abschnitt beschäftigt sich mit der Konvexitätsanalyse der modellprädiktiven Regelung für MTI Systeme. Ist das Optimierungsproblem konvex, so kann mit bekannten Algorithmen, z.B. mit dem Interior Point Algorithmus, das globale Minimum der Kostenfunktion schnell gefunden werden [9]. Die Ergebnisse die im Folgenden Abschnitt vorgestellt werden sind in [26] veröffentlicht.

7.3.1 Input lineare MTI Systeme

Eine allgemeine Einführung der multilinearen zeitinvarianten (MTI) Systeme erfolgte in Abschnitt 7.2.1.2. Für die Untersuchung der Konvexität des Optimierungsproblems wird angenommen, dass es sich um ein MTI System mit einem Eingang handelt.

Für die weiterführenden Untersuchungen wird eine Modellklasse verwendet mit einem Eingang, welcher linear in die Zustandsübergangsfunktion eingeht. Somit ändert sich die Zustandsübergangsgleichung (7.2-10 zu

$$\mathbf{x}(k+1) = \hat{\mathbf{F}}\mathbf{m}(\mathbf{x}(k)) + \mathbf{B}u(k) \tag{7.3-27}$$

mit der Zustandsmatrix $\hat{\mathbf{F}} \in \mathbb{R}^{n\times 2^n}$ und dem Eingangsvektor $\mathbf{B} \in \mathbb{R}^{n\times 1}$. Ein solches MTI System ist multilinear in den Zuständen und linear in den Eingängen. Es bedeutet, dass Multiplikationen von Zuständen auftreten können, aber keine Multiplikationen des Eingangs mit einem Zustand.

Beispiel 7.3.1 *Die Zustandsübergangsfunktion mit einem Eingang u und zwei Zuständen x_1 und x_2 ist gegeben durch*

$$\begin{pmatrix} x_1(k+1) \\ x_2(k+1) \end{pmatrix} = \begin{pmatrix} \hat{f}_{1,1} & \hat{f}_{1,2} & \hat{f}_{1,3} & \hat{f}_{1,4} \\ \hat{f}_{2,1} & \hat{f}_{2,2} & \hat{f}_{2,3} & \hat{f}_{2,4} \end{pmatrix} \begin{pmatrix} 1 \\ x_1(k) \\ x_2(k) \\ x_1(k)x_2(k) \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} u(k) \; .$$

Im folgenden wird ein solches System als input lineares MTI System bezeichnet.

7.3.2 Optimierungsproblem der modellprädiktiven Regelung

Im allgemeinen ist das Lösen eines nichtlinearen Optimierungsproblems sehr aufwendig und es ist nicht garantiert, dass das globale Optimum gefunden wird. Ist das Optimierungsproblem konvex gibt es sehr effiziente Methoden dieses zu lösen und das globale Optimum zu finden, z.B. die Interior Point

Methode, [9]. Aus diesem Grund ist es sinnvoll das Optimierungsproblem für MTI Systeme auf Konvexität zu untersuchen.

Die qudratische Kostenfunktion der modellprädiktiven Regelung, [30]

$$J(\mathbf{u}) = \underbrace{\sum_{i=1}^{H_p} \|\mathbf{x}(k+i) - \mathbf{r}(k+i)\|^2_{\mathbf{Q}(i)}}_{J_x(\mathbf{u})} + \underbrace{\sum_{i=0}^{H_u-1} \|\Delta\mathbf{u}(k+i)\|^2_{\mathbf{R}(i)}}_{J_{\Delta u}(\mathbf{u})}, \tag{7.3-28}$$

mit der Übergangsfunktion $\mathbf{x}(k+1) = \mathbf{Fm}(\mathbf{x}(k), \mathbf{u}(k))$, dem Referenzsignal $\mathbf{r}(k)$ und den Änderungen der Eingänge $\Delta\mathbf{u}(k)$, wird im Folgenden verwendet. Der Vorhersagehorizont wird mit H_p bezeichnet und der Regelungshorizont mit H_u, für die gilt $H_u \leq H_p$. Für die beiden Wichtungsmatrizen $\mathbf{Q}(i) \geq 0$ und $\mathbf{R}(i) \geq 0$ wird im folgenden angenommen, dass es sich um Diagonalmatrizen handelt mit den Diagonalelementen $q_j(i)$ und $r_k(i)$, sodass gilt

$$\mathbf{Q}(i) = \underset{j=1,\ldots,n}{\operatorname{diag}} \left(q_j(i)\right), \; \mathbf{R}(i) = \underset{k=1,\ldots,m}{\operatorname{diag}} \left(r_k(i)\right).$$

Das Optimierungsproblem

$$\min_{\mathbf{u}\in\mathcal{U}} J(\mathbf{u}) \tag{7.3-29}$$

ist ein Minimierungsproblem mit dem Vektor $\mathbf{u} = (u(k), u(k+1), ..., u(k+H_p)) \in \mathcal{U}$, welcher die Optimierungsvariablen darstellt. Die Menge $\mathcal{U}$ bezeichnet die Menge der Optimierungsvariablen. Die quadratische Kostenfunktion (7.3-28) wird minimiert in dem die bestmögliche Eingangsfolge $\mathbf{u}$ gesucht wird. Das Optimierungsproblem (7.3-29) ohne Randbedingungen ist konvex, wenn die Kostenfunktion (7.3-28) und die Menge der Optimierungsvariablen $\mathcal{U}$ konvex ist. Für die folgende Betrachtung wird angenommen, dass die Menge $\mathcal{U}$ konvex ist.

Im folgende wird die Konvexität der Kostenfunktion untersucht. Die Kostenfunktion $J(\mathbf{u})$ ist konvex wenn die Hesse-Matrix $\mathbf{H}$ bzw. die zweite Ableitung von $J(\mathbf{u})$ positiv semi-definit ist

$$\mathbf{H} = \nabla^2 J(\mathbf{u}) \geq 0. \tag{7.3-30}$$

Um zu prüfen, ob die Bedingung $\mathbf{H} \geq 0$ erfüllt ist müssen die Eigenwerte $\lambda_i,\ i = 1, \ldots, H_p$ der Hesse-Matrix $\mathbf{H}$ größer oder gleich null sein. Eine ausführlichee Beschreibung des konvexen Optimierungsproblems können in [9] gefunden werden, genauso wie die Methoden um die Konvexität zu beweisen.

Die Summe von konvexen Funktionen ist wieder eine konvexe Funktion. Das bedeutet, dass die beiden Summanden $J_{\Delta u}(\mathbf{u})$ und $J_x(\mathbf{u})$ der Funktion 7.3-31 unabhängig von einander auf Konvexität untersucht werden können. Der Term $J_{\Delta u}(\mathbf{u})$ ist offensichtlich Konvex, da es sich um eine Summe von quadratischen Termen handelt, unabhängig von der Wahl von H_u. Damit reduziert sich die Analyse der Konvexität von $J(\mathbf{u})$ auf den ersten Term $J_x(\mathbf{u})$ der Kostenfunktion. Die Ergebnisse der Untersuchung der reduzierten Kostenfunktion

$$\begin{aligned} J_x(\mathbf{u}) &= \sum_{i=1}^{H_p} \|\mathbf{x}(k+i) - \mathbf{r}(k+i)\|^2_{\mathbf{Q}(i)} \\ &= \sum_{i=1}^{H_p} (\mathbf{x}(k+i) - \mathbf{r}(k+i))^T \mathbf{Q}(i) (\mathbf{x}(k+i) - \mathbf{r}(k+i)) \end{aligned} \tag{7.3-31}$$

werden in dem folgenden Abschnitt für verschiedene Vorhersagehorizonte zusammengefasst.

7.3.3 Konvexitätsuntersuchungen für verschiedene Vorhersagehorizonte

Als erstes wird ein Vorhersagehorizont von $H_p = 1$ betrachtet. Mit der allgemeinen Definition der MTI Systeme (7.2-10) wie sie in Abschnitt 7.2.1.2 eingeführt wurde, mit der Einschränkung auf einen Eingang,

kann die Abhängigkeit von $\mathbf{x}(k+1)$ von u wie folgt dargestellt werden

$$\mathbf{x}(k+1) = \mathbf{F}_{:,1:2^n}\mathbf{m}(\mathbf{x}) + \mathbf{F}_{:,2^n+1:2^{n+1}}\mathbf{m}(\mathbf{x})u(k) \tag{7.3-32}$$

mit den konstanten Vektoren $\mathbf{F}_{:,1:2^n}\mathbf{m}(\mathbf{x}) \in \mathbb{R}^n$ und $\mathbf{F}_{:,2^n+1:2^{n+1}}\mathbf{m}(\mathbf{x}) \in \mathbb{R}^n$. Die zwei Vektoren $\mathbf{F}_{:,1:2^n}\mathbf{m}(\mathbf{x})$ und $\mathbf{F}_{:,2^n+1:2^{n+1}}\mathbf{m}(\mathbf{x})$ sind konstant, da sie nur von den Zuständen $\mathbf{x}$ zur Zeit k abhängen, welche sich während der Optimierung nicht verändern.

Durch einsetzen von (7.3-32) in (7.3-31) erhält man die Kostenfunktion eines Zeitschrittes

$$\begin{aligned} J(k+1)_x(u) &= \sum_{i=1}^{n} q_i(1)(x_i' - r_i')^2 \\ &= \sum_{i=1}^{n} q_i(1)(\mathbf{F}_{i,1:2^n}\mathbf{m}(\mathbf{x}) + \mathbf{F}_{i,2^n+1:2^{n+1}}\mathbf{m}(\mathbf{x})u - r(k+1)_i)^2. \end{aligned} \tag{7.3-33}$$

Die Untersuchungen des Optimierungsproblems mit der Kostenfunktion (7.3-33) eines Zeitschrittes führt zu folgendem Lemma.

Lemma 7.3.1 *Das Optimierungsproblem (7.3-29) eines MTI Systems (7.2-10) mit einem Vorhersagehorizont von eins ist konvex.*

Der Beweis des Lemmas ist in [26] zu finden.

Die input lineare Zustandsübergangsfunktion für zwei Zeitschritte ist gegeben durch

$$\mathbf{x}(k+2) = \hat{\mathbf{F}}\mathbf{m}(\mathbf{x}(k+1)) + \mathbf{B}u(k+1). \tag{7.3-34}$$

Einsetzen von (7.3-27) ergibt die Funktion

$$\mathbf{x}(k+2) = \hat{\mathbf{F}}\left(\bigotimes_{i=n}^{1}\begin{pmatrix}1\\ \hat{\mathbf{F}}_{i:}\mathbf{m}(\mathbf{x}(k)) + \mathbf{B}u(k)\end{pmatrix}\right) + \mathbf{B}u(k+1), \tag{7.3-35}$$

dabei hängen die Zustände $\mathbf{x}(k+2)$ nur noch von den Zuständen $\mathbf{x}(k)$ ab. Das Symbol $\otimes$ bezeichnet das Kronecker-Produkt. Die Konvexitätsuntersuchungen des Optimierunsgproblems für einen Vorhersagehorizont von $H_p = 2$ haben gezeigt, dass es nicht mehr konvex ist für die gesamte Klasse der input linearen MTI Systeme und wurde in folgendem Lemma zusammengefasst.

Lemma 7.3.2 *Das Optimierungsproblem (7.3-29) eines input linearen MTI Systems (7.3-27) ohne zusätzliche strukturelle Einschränkungen ist nicht konvex für einen Vorhersagehorizont von zwei.*

Der Beweis wurde anhand eines Gegenbeispipels geführt und kann in [26] nachgelesen werden. Für die weiteren Untersuchungen wird eine Unterklasse der input linearen MTI Systeme eingeführt.

Definition 7.3.1 *Die Unterklasse der input linearen MTI Systeme ist definiert als*

$$\mathbf{x}(k+1) = \hat{\mathbf{F}}\mathbf{m}(\mathbf{x}(k)) + \mathbf{B}u(k)\ , \tag{7.3-36}$$

mit Bedingungen an die Matrizen $\hat{\mathbf{F}}$ und $\mathbf{B}$, welche von einander abhängen.

Die Matrix $\hat{\mathbf{F}}$ muss die Bedingung

$$P(\hat{\mathbf{F}}) \leq \mathbf{S} \tag{7.3-37}$$

elementweise erfüllen. Dabei wird die geforderte Struktur von $\hat{\mathbf{F}}$ durch die sogenannte Vergleichsmatrix $\mathbf{S}$ festgelegt. Der Operator P wird elementweise auf die Matrix $\hat{\mathbf{F}}$ angewandt, wodurch eine Strukturmatrix von $\hat{\mathbf{F}}$ entsteht, welche wie folgt mit Nullen und Einsen gefüllt ist

$$P(\hat{f}_{ij}) = \begin{cases} 1 & \text{if } \hat{f}_{ij} \neq 0, \\ 0 & \text{if } \hat{f}_{ij} = 0. \end{cases} \tag{7.3-38}$$

Die Anzahl der Elemente von $\mathbf{B}$*, welche ungleich null sind wird mit* l *bezeichnet und es gilt*

$$b_i = \begin{cases} b_i & \textit{for } i = 1, ..., l, \\ 0 & \textit{otherwise}. \end{cases} \tag{7.3-39}$$

Die Vergleichsmatrix $\mathbf{S}$ *wird wie folgt gebildet*

$$\mathbf{S} = \mathbf{e} \cdot \left(\bigotimes_{k=n-l}^{n} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \otimes \boldsymbol{\gamma} \right)^T . \tag{7.3-40}$$

wobei der Vektor $\mathbf{e} \in \mathbb{R}^n$ *mit Einsen gefüllt ist und der Vektor* $\boldsymbol{\gamma} \in \mathbb{R}^{2^l}$ *wie folgt definiert ist*

$$\gamma_i = \begin{cases} 1 & \textit{for } i = 1, \ldots, 2^{j-1} + 1 \qquad j = 1, ..., l, \\ 0 & \textit{otherwise}, \end{cases} \tag{7.3-41}$$

mit $i = 1, ..., 2^l$*.*

Mit diesen Struktureinschränkungen des input liearen MIT Systems führen die Konvexitätsuntersuchungen auf folgendes Theorem, wobei der Beweis wieder in [26] zu finden ist.

Theorem 7.3.1 *Das Optimierungsproblem (7.3-29) eines input linear MTI Systems (7.3-36) ist konvex wenn die Struktureinschränkung (7.3-37) erfüllt ist und der Vorhersagehorizont* H_p *kleiner oder gleich zwei ist.*

Es konnte gezeigt werden, dass das Optimierungsproblem (7.3-29) für die gesamte Klasse der MTI Systeme mit einem Eingang und einem Vorhersagehorizont von eins konvex ist. Für eine Vorhersagehorizont von zwei gilt dies nicht mehr. Es wurde eine spezielle Unterklasse der input linearen MTI Systeme eingeführt für welche die Konvexität für zwei Zeitschritte gezeigt werden konnte. Im folgende Abschnitt wird gezeigt wie die Konvexitätseingenschaft für größere Vorhersagehorizonte genutzt werden kann und anhand eines Heizungssystems ein Anwendungsbeispiel gezeigt.

7.3.4 Nutzen und Anwendungsbeispiel

Eine allgemeine Aussage über die Konvexität des Optimierungsproblems für $H_p > 2$ kann an dieser Stelle nicht gemacht werden. Aber wenn das Optimierungsproblem (7.3-29) nicht konvex ist hängt das Ergebnis von der Wahl der Startwerte der Optimierungsvariablen $\mathbf{u}_0$ ab. Anhand eines Beispiels wird im folgenden gezeigt wie die Konvexität des Optimierungsproblems für einen Vorhersagehorizont von zwei genutzt werden kann um bessere Ergebnisse zu erzielen, bei einer nicht konvexen Optimierung mit einem Vorhersagehorizont von zehn.

Beispiel 7.3.2 *Ein input lineares MTI System mit zwei Zuständen und einem input wird als Beispiel verwendet. Die Übergangsmatrix und der Eingangsvektor sehen wie folgt aus*

$$\hat{\mathbf{F}} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \ \mathbf{B} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Das Optimierungsproblem (7.3-29) soll für dieses System zur Zeit k *gelöst werden, wobei der Vorhersagehorizont zehn ist und die Referenz für die Zustände wie folgt gewählt wird* $\mathbf{r}(k+i) = \begin{bmatrix} 2 & 2 \end{bmatrix}^T$*,* $i = 1, \ldots, 10$*. Die beiden Wichtungsmatrizen* $\mathbf{Q}(i)$ *und* $\mathbf{R}(i)$ *sind Einheitsmatrizen und die Eingänge des Systems werden in dem Intervall* $[-10,\ 10]$ *begrenzt.*

Das System

$$\mathbf{x}(k+1) = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} 1 \\ x_1(k) \\ x_2(k) \\ x_1(k)x_2(k) \end{pmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u(k) \tag{7.3-42}$$

gehört zur Unterklasse der input linearen MTI Systeme wie sie nach der Definition 7.3.1 eingeführt wurde. Das bedeutet, dass das Optimierungsproblem für einen Vorhersagehorizont von zwei konvex ist. Für größerer Vorhersagehorizonte ist das Optimierungsproblem nicht konvex, wenn das MTI System (7.3-42) verwendet wird. Aus diesem Grund soll die Tatsache genutzt werden, dass für $H_p = 2$ das Optimierungsproblem konvex ist und die Lösung das globale Optimum ist. Das heißt, dass als erstes die Lösung für einen Vorhersagehorizont von zwei berechnet wird, anstatt direkt die Optimierung mit einem Vorhersagehorizont von zehn auszuführen und zufällig gewählten Startbedingungen der Optimierungsvariablen $\mathbf{u}_0 \in \mathbb{R}^{10}$.

Die optimalen Ergebnisse der Eingänge $u(k)_{opt}$ und $u(k+1)_{opt}$ der konvexen Optimierung mit einem Vorhersagehorizont von zwei, werden nun als die ersten beiden Startwerte für die Optimierung mit einem Vorhersagehorizont von $H_p = 10$ genutzt,

$$\tilde{\mathbf{u}}_0 = \begin{bmatrix} u(k)_{opt} & u(k+1)_{opt} & \mathbf{u}_{0,3:10} \end{bmatrix}^T.$$

Im folgenden werden die Optimierungsergebnisse verglichen, wenn die zwei unterschiedlichen Vektoren $\mathbf{u}_0$ und $\tilde{\mathbf{u}}_0$ verwendet werden. Um die Ergebnisse zu vergleichen wurde die Optimierung 1000 mal, mit unterschiedlichen und zufällig gewählten Startwerten $\mathbf{u}_0$, ausgeführt. Die Ergebnisse, welche mit dem interior point Algorithmus erzielt wurden, werden mit den Ergebnissen eines globalen Optimierers (genetischer Algorithmus) verglichen. Die Ergebnisse zeigten, dass die Optimierung 179 mal fehlschlug wenn die zufälligen Startwerte $\mathbf{u}_0$ genutzt wurden und 48 mal wenn die Startwerte $\tilde{\mathbf{u}}_0$ genutzt wurde. Dabei bedeutet ein Fehler, dass das Ergebnis der Optimierung nicht das globale Minimum der Kostenfunktion darstellt. Die Ergebnisse zeigen den Vorteil, den die Wahl der Startwerte $\tilde{\mathbf{u}}_0$ gegenüber einer Wahl komplett zufälliger Startwerte $\mathbf{u}_0$ bringt. Für dieses Beispiel konnten die Anzahl der Optimierung bei denen das gloabel Minimum nicht gefunden wurde um einen Faktor von 3.7 reduziert werden.

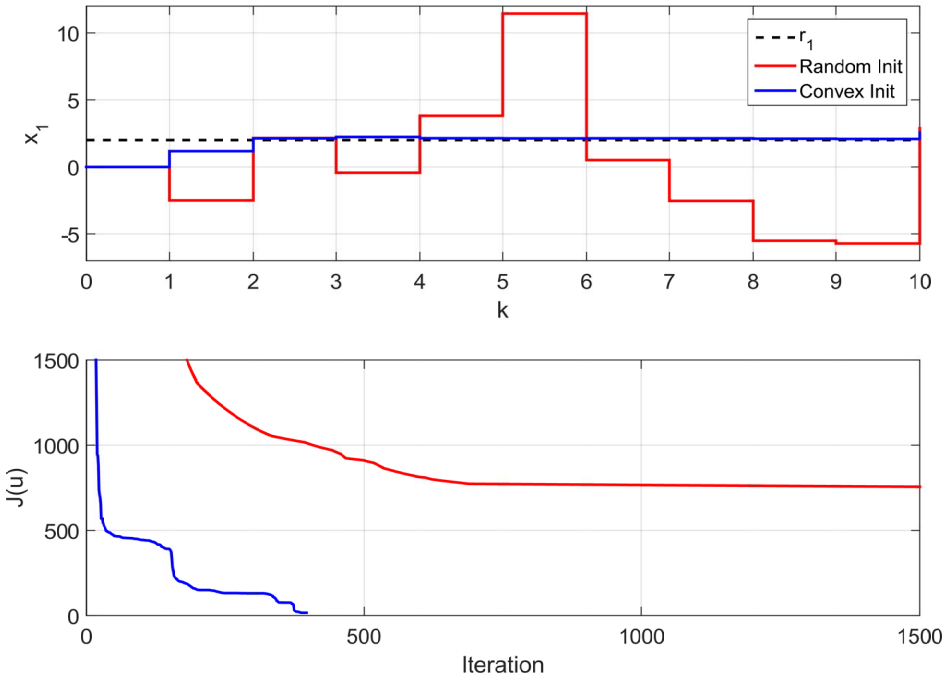


Abbildung 7.3-2: Vergleich der Optimierungsergebnisse mit unterschiedlichen Startwerten $\mathbf{u}_0$ und $\tilde{\mathbf{u}}_0$. , [26]

Abbildung 7.3-2 zeigt das Ergebnis der Optimierung mit den Startwerten $\mathbf{u}_0$ und $\tilde{\mathbf{u}}_0$ und bildet die Trajektorien des Zustandes x_1 des Systems (7.3-42) ab, wobei nur die Optimierung mit den Startwerten $\tilde{\mathbf{u}}_0$ das globale Optimum erreicht. Der Vergleich der Kostenfunktion über die Iterationen der Optimierung zeigt, dass der letzte Wert der Kostenfunktion wesentlich kleiner ist wenn die Startwerte $\tilde{\mathbf{u}}_0$ verwendet werden. Auch stoppt der Algorithmus wesentlich schneller nach weniger Iterationen, wenn die Ergebnisse der konvexen Optimierung als Startwerte verwendet werden.

Im folgenden wird die modellprädiktive Optimierung beispielhaft auf ein Heizungssystem angewandt, welches zur Klasse der input linearen MTI Systeme gehört. Das Modell des Heizungssystems beinhaltet einen Kessel, eine Pumpe und einen Verbraucher mit den Heizkörpern und den Räumen wie sie in Abschnitt 7.2.2 eingeführt wurden. Abbildung 7.3-3 zeigt ein Schema des Heizungssystems. Die

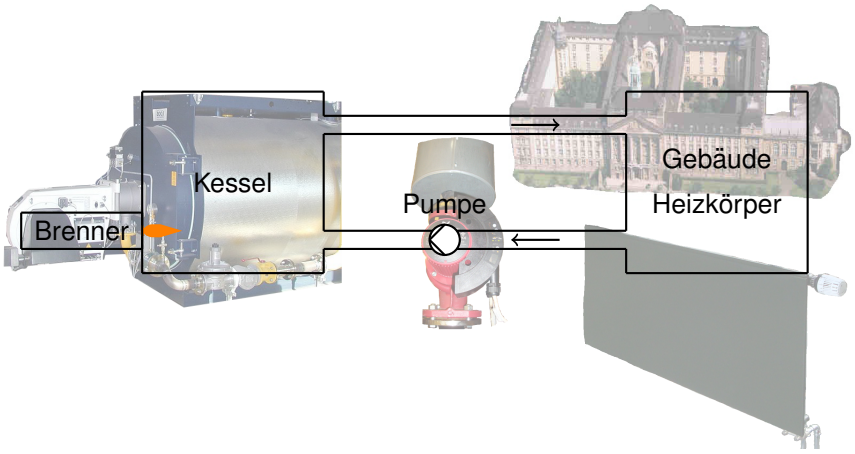


Abbildung 7.3-3: Schema des Heizungssystems

einzelnen Komponenten des Systems werden mit Hilfe von Wärmeleistungsbilanzen modelliert, [22]. Das resultierende Modell besteht aus drei Zuständen und eine Eingang

$$\mathbf{x} = \begin{bmatrix} T_{raum} & T_{rl,h} & T_{vl,k} \end{bmatrix}^T , \; u = \alpha \; , \tag{7.3-43}$$

wobei $T_{vl,k}$ die Vorlauftemperatur, $T_{rl,h}$ die Rücklauftemperatur, T_{raum} die Gebäudetemperatur ist und α das Modulationssignal der Kesselleistung ist, [37].

Die Pumpe legt nach Gleichung (7.2-23) den Volumenstrom $\dot{V}$ in Abhängigkeit der Differenz zwischen der Gebäudetemperatur T_{raum} und gewünschten Gebäudetemperatur $T_{raum,ref}$ fest. Die Gleichungen für die Vorlauftemperatur $T_{vl,k}$ des Kessel (7.2-20), die Rücklauftemperatur $T_{rl,h}$ des Heizkörpers (7.2-21) und die Raumtemperatur T_{raum} des Gebäudes (7.2-22) bilden ein System von Differentialgleichungen welche das dynamische Verhalten des Heizungssystems wiedergeben. Ein zeitdiskrete Modell der Anlage wird durch die Anwendung des Euler-Verfahrens mit der Abtastzeit t_s erstellt, wodurch die Differentialgleichungen zu Differenzengleichungen werden. Dabei sind die rechten Seiten der Differenzengleichungen input lineare multilineare Funktionen, sodass das Verhalten des Heizungssystems durch ein input lineares MTI System abgebildet werden kann, mit der Übergangsmatrix und dem Eingangsvektor

$$\hat{\mathbf{F}} = \begin{bmatrix} 0 & \hat{f}_{1,2} & \hat{f}_{1,3} & 0 & 0 & \hat{f}_{1,6} & \hat{f}_{1,7} & 0 \\ 0 & \hat{f}_{2,2} & \hat{f}_{2,3} & 0 & \hat{f}_{2,5} & \hat{f}_{2,6} & \hat{f}_{2,7} & 0 \\ \hat{f}_{3,1} & 0 & \hat{f}_{3,3} & 0 & \hat{f}_{3,5} & 0 & 0 & 0 \end{bmatrix} , \; \mathbf{B} = \begin{bmatrix} b_1 & 0 & 0 \end{bmatrix}^T ,$$

welche alle Parameter des Modells beinhalten.

Um zu prüfen, ob das Optimierungsproblem der modellprädiktiven Regelung eines solchen Systems für zwei Zeitschritten konvex ist, muss die Strukturbedingung (7.3-37) erfüllt sein.

Die Anzahl der Eingänge $l = 1$ führt nach Gleichung (7.3-40) auf die Vergleichsmatrix

$$\mathbf{S} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Der Vergleich mit der Strukturmatrix von $\hat{\mathbf{F}}$ nach (7.3-37) führt zu

$$P(\hat{\mathbf{F}}) = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \leq \mathbf{S}.$$

Damit wurde gezeigt, dass das Model die Strukturanforderungen der Definition 7.3.1 erfüllt. Nach dem Theorem 7.3.1 bedeutet dies, dass das Optimierungsproblem der modellrädiktiven Regelung für einen Vorhersagehorizont von $H_p = 2$ konvex ist. Außerdem wurde die Menge $\mathcal{U}$ der Eingänge so eingeschränkt, dass $((k)u)$ und $u(k+1)$ in dem Intervall $[0, 1]$ liegen. Damit ist auch die Bedingung, dass die Menge $\mathcal{U}$ der Optimierungsvariablen konvex ist, erfüllt. Damit ist das Optimierungsproblem

$$\min_{u(k),u(k+1)\in\mathcal{U}} J(u(k), u(k+1)) \tag{7.3-44}$$

konvex.

Auf Grund der Konvexität des Optimierungsproblems 7.3-44 können für $\mathbf{u}_0 = \begin{bmatrix} u(k)_0 & u(k+1)_0 \end{bmatrix}^T$ beliebige Startwerte gewählt werden, solange sie in dem Intervall $[0, 1]$ liegen. Die Referenzen für die drei Zustände T_{raum}, $T_{rl,h}$ und $T_{vl,k}$ des Modells wurden wie folgt festgelegt

$$\mathbf{r}(k+1) = \begin{bmatrix} 292K & 318K & 338K \end{bmatrix}^T,\ \mathbf{r}(k+2) = \begin{bmatrix} 292K & 318K & 338K \end{bmatrix}^T.$$

Die Optimierung wurde mehrmals mit unterschiedlichen Startwerten $\mathbf{u}_0$ mit dem interior point Algorithmus ausgeführt. Das Ergebnis war immer das globale Minimum

$$\mathbf{u}_{opt} = \begin{bmatrix} 0.2285 & 0.4624 \end{bmatrix}^T.$$

Auch konvergierte der Algorithmus nach wenigen Iterationen. Es hat sich gezeigt, dass die Optimierung der modellprädiktiven Regelung für eine Unterklasse der input linearen MTI Systeme sehr effizient für eine Vorhersagehorizont von zwei gelöst werden kann und das Ergebnis zuverlässig das globale Minimum ist.

7.4 Iterativ lernende Regelung für die prädiktive Regelung

In diesem Abschnitt wird ein Regelungskonzept vorgestellt welches iterative lernenden Regelung mit der modellprädiktiven Regelung (MPC - model predictive control) kombiniert. Eine solche Kombination erscheint sinnvoll für Anwendungen, bei denen es Modelle gibt, welche die Systemdynamiken näherungsweise abbilden und gleichzeitig die relevanten Störungen eine Periodizität aufweisen, wie z.B. die Außentemperatur bei Heizungssystemen.

Die modellprädiktive Regelung für Heizungssysteme welche die Wettervorhersage berücksichtigt ist eine vielversprechender Ansatz um den Energieverbrauch zu optimieren, [36, 40, 19]. Ein Modell zu erstellen welches die Systemdynamiken richtig wiedergibt kann auf der einen Seite schwierig sein und auf der anderen Seite zu nichtlinearen Modellen und damit zur nichtlinearen modellprädiktiven Regelung (NMPC - nonlinear model predictive control) führen. Die Lösung des nichtlinearen Optimierungsproblems der modellprädiktiven Regelung erfordert im allgemeinen einen hohen Rechenaufwand. Auch ist nicht sichergestellt, dass das globale Minimum gefunden wird, da das Optimierungsproblem nicht konvex ist. Im Gegensatz dazu führt ein lineares Modell des Systems zu einem linearen Optimierungsproblem welches mit bekannten Algorithmen schnell und effizient gelöst werden kann, [9, 30].

Die iterative lernenden Regelung (ILC - iterative learning control) ist für periodische Prozesse geeignet. Beispielsweise für die Ausregelung von periodischen Störungen oder einer periodischen Referenzfolge. Weit verbreitet ist die iterativ lernenden Regelung für industrielle Prozesse bei denen eine Maschine, z.B. ein Roboterarm immer wieder die selbe Aufgabe ausführt und der selben Referenz folgt, [10, 43]. Auch für hochpräzise Maschinen wie dem Freien-Elektron-Laser FLASH am DESY (Deutsches Elektronen-Synchrotron) in Hamburg kommt die iterative lernenden Regelung zum Einsatz, [38]. Dabei wird in jeder neuen Iteration von der voherigen Iteration gelernt und das neue Eingangssignal für das System berechnet. Auch für Heizungssysteme oder im allgemeinen für HVAC (Heating Ventilation and air-conditioning) Systeme gibt es Ansätze wie die iterativ lernende Regelung genutzt werden kann, [44, 31, 13].

Bei Heizungsystemen drückt sich die Periodizität in in den Störungen in Form von Tagesläufen der Außentemperatur aus. Im Vergleich zu industriellen Prozessen bei der eine Iteration der nächsten gleicht, ändern sich die Wetterverhätlnisse täglich. Das heißt, dass im Vergleich zu den präzissen industriellen Prozessen, bei Heizungssystemen nicht unbedingt von dem vorherigen Tag „gelernt“ werden kann. Aber über einen längeren Zeitraum betrachtet, wird es Tage in der Vergangenheit geben, welche dem nächsten Tag in den Umgebungsbedingungen ähneln. Wenn alle vorherigen Iterationen gespeichert werden, kann diejenige aus dem gesamten Datensatz ausgewählt werden, welche am besten zu der folgenden Iteration passt, auf Grund ähnlicher Umgebungsbedingungen. Im Folgenden wird ein solcher datenbasierter iterativ lernender Regelungsansatz eingeführt und vorgestellt wie dieser für die lineare modellprädiktive Regelung genutzt werden kann. Ein einfaches lineares Modell bildet die dynamiken eines Systems approximativ ab. Der datenbasierte ILC soll genutzt werden um mögliche Modellierungsfehler auszugleichen und eine angepasste Referenz für den MPC zu berechnen. Für die Anwendung hätte das den großen Vorteil, dass einfache lineare Modelle genutzt werden können, wodurch der Moedellierungsaufwand erheblich reduziert werden würde. Die iterative lernende prädiktive Regelung wird anschließend auf ein Heizungssystem angewandt. Teile dieses Abschnittes sind in [27] veröffentlicht.

7.4.1 Iterativ lernende Regelung

Für periodische Prozesse bei denen sich die Periodizität von einer Iteration d zur nächsten Iteration $d+1$ nicht ändert hat sich gezeigt, dass iterativ lernende Algorithmen besonders gut geeignet sind. Bei der iterativ lernenden Reglung wird das Eingangssignal $\mathbf{u}_{d+1}$ der nächsten Iteration aus dem Eingangssignal der vorherigen Iteration $\mathbf{u}_d$, sowie der Abweichung des Ausgangssignals $\mathbf{y}_d$ von einer Referenz $\mathbf{r}_{ilc}$, gewichtet mit dem Verstärkungsfaktor γ berechnet, [7]

$$\mathbf{u}_{d+1}(k) = \mathbf{u}_d(k) + \mathbf{u}_d(k) + \gamma \mathbf{e}_d(k), \tag{7.4-45}$$

mit dem Index der Iterationen $d \in \mathbb{N}$ und der Abweichung $\mathbf{e}_d = \mathbf{r}_{ilc} - \mathbf{y}_d$ von der Referenz.

Abbildung 7.4-4 zeigt die schematische Darstellung des Regelkreises mit einem iterativ lernenden Regler.

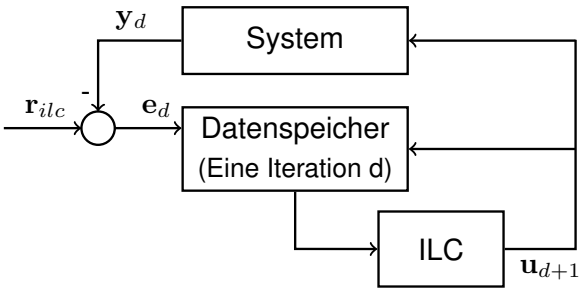
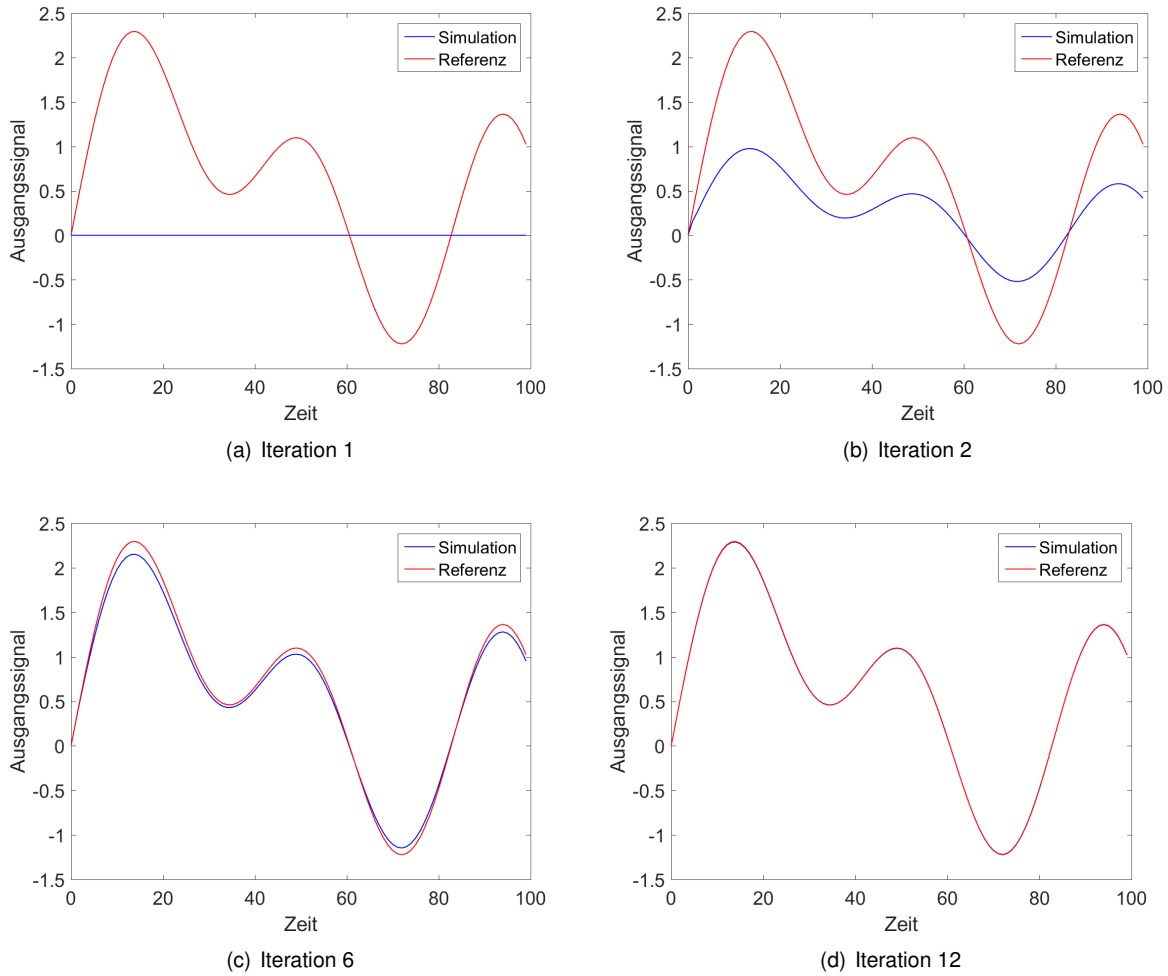


Abbildung 7.4-4: Schematische Darstellung eines Regelkreises mit einem ILC

Die Abbildungen 7.4.1 (a)-(d) zeigen an einem Beispiel die Funktionsweise des Algorithmusses (7.4-45), dabei nähert sich das Ausgangssignal $\mathbf{y}_d$ von Iteration zu Iteration der Referenz $\mathbf{r}_{ilc}$ an. Für die Simulation wurde ein System zweiter Ordnung verwendet, mit einem Eingang u sowie einem Ausgang y. Die Abtastzeit wurde auf 0.1 Sekunden gesetzt und ein Verstäkungsfaktor von $\gamma = 0.6$ gewählt. Das Eingangssignal der ersten Iteration war null. Das Ergebnis ist in 7.4.1 (a) zu sehen. Die Anderen Abbildungen zeigen die Ergebnisse nach 2, 6 und 12 Iterationen, wobei sich nach 12 Iterationen das Ausgangssignal der Referenz angenähert hat.



(a) Iteration 1

(b) Iteration 2

(c) Iteration 6

(d) Iteration 12

Abbildung 7.4-5: Simulationsergebnisse unterschiedlicher Iterationen eines ILC's

Das einfache Beispiel zeigt wie ein ILC arbeitet wenn das Eingangssignals $\mathbf{u}_{d+1}$ nach Gleichung (7.4-45) berechnet wird. Dies funktioniert sehr gut wenn sich die Periodizität von Iteration zu Iteration nicht ändert. Im folgenden Abschnitt wird ein datenbasierter iterativ lernender Regler eingeführt durch den es möglich ist einen ILC für Prozesse zu nutzen, bei denen sich die Periodizität von Iteration zu Iteration ändert aber bei der Betrachtung eines längeren Zeitraums von mehreren Iterationen doch immer wieder ähnlich ist.

7.4.2 Datenbasierte iterativ lernende Regelung

Für Systeme bei denen sich die Periodizität nicht in jeder Iteration wiederholt kann das Eingangssignal nicht nach Gleichung (7.4-45) berechnet werden. Aus diesem Grund wird für die Berechnung des Eingangssignals $\mathbf{u}_{d+1}$ eine der Iterationen aus einem vergangenen und größerem Zeitraum für die Berechnung ausgewählt. Die historischen Daten der vorangegangenen Iterationen werden in einer Datenbank oder auf einer Festplatte gespeichert um diese für die Berechnung zu nutzen. Somit verändert sich Gleichung (7.4-45) zu

$$\mathbf{u}_{d,next}(k) = \mathbf{u}_{d,db}(k) + \gamma \mathbf{e}_{d,db}(k). \tag{7.4-46}$$

Die Vektoren $\mathbf{u}_{d,db}$ und $\mathbf{e}_{d,db}$ stammen nicht mehr von der vorherigen Iteration sondern werden aus einer Vielzahl von gespeicherten Iterationen ausgewählt. Um zu entscheiden welcher historische Datensatz ausgewählt wird werden die Umgebungsbedingungen $\mathbf{a}_d$ einer Iteration zusätzlich zu den Vektoren $\mathbf{u}_{d,db}$ und $\mathbf{e}_{d,db}$ in der Datenbank abgelegt. Die gespeicherten Umgebungsbedingungen werden mit denen der nächsten Iteration verglichen um zu entscheiden welcher Datensatz ausgewählt wird. Dies geschieht über den sogenannten Elementselektor und bedeutet auch, dass es eine Vorhersage der Umgebungsbedingungen $\mathbf{a}_{d,pre}$ zugänglich sein muss. Abbildung 7.4-6 zeigt die schematische Darstellung eines datenbasierten iterativ lernenden Reglers.

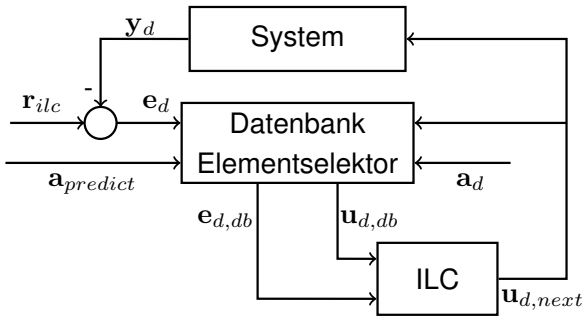


Abbildung 7.4-6: Schematische Darstellung eines Regelkreises mit einem datenbasierten ILC

Für den Elementselektor werden zwei Auswahlkriterien definiert um zu entscheiden welcher historische Datensatz ausgewählt wird. Als erstes wird ein Ähnlichkeitskriterium definiert, welches die Abweichung der gespeicherten Umgebungsbedingungen $\mathbf{a}_d$ einer Iteration d, von der Vorhersage der Umgebungsbedingungen $\mathbf{a}_{d,pre}$ bewertet. Dazu wird die Quadratsumme der Differenz zwischen $\mathbf{a}_d$ und $\mathbf{a}_{d,pre}$ zum Zeitpunkt k gebildet

$$\mathbf{E}_T(d) = \sum_k (\mathbf{a}_d(k) - \mathbf{a}_{d,pre}(k))^2. \tag{7.4-47}$$

Als zweites wird ein Performancekriterium eingeführt um die Abweichung $\mathbf{e}_d$ zu bewerten, welche sich von Iteration zu Iteration ändert. Um die Iteration mit der kleinsten Abweichung zu finden wird die Quadratsumme von $\mathbf{e}_d$ gebildet

$$\mathbf{E}_e(d) = \sum_k \mathbf{e}_d^2(k). \tag{7.4-48}$$

In den meisten fällen wird das Minimum von $\mathbf{E}_e(d)$ und das Minimum von $\mathbf{E}_T(d)$ nicht zu der selben Iteration d gehören. Aus diesem Grund wird ein weiteres konstantes Kriterium eingeführt. Der Wert $\mathbf{E}_T(d)$ der ausgewählten Iteration soll kleiner sein als der konstante Wert E_M. Zusammen führen diese Kriterien

auf das Optimierungsproblem

$$\min_{d \in D} \mathbf{E}_e(d), \qquad s.t.\, \mathbf{E}_T(d) < E_M \tag{7.4-49}$$

mit dem konstanten Parameter E_M. Für den Elementselektor bedeutet dies, dass das Optimierungsproblem (7.4-49) gelöst werden muss, um den passenden Datensatz in den vergangenen Iterationen zu finden. Für den datenbasierten ILC müssen neben den Vektoren $\mathbf{e}_d$ und $\mathbf{u}_d$ auch die Umgebungsbedingungen aller vergangener Iterationen gespeichert werden, was zu einem hohen Speicherbedarf in einer Datenbank oder einem lokalen Speicher führen kann. Abbildung 7.4.2 zeigt eine Auswertung des Optimierungsproblems (7.4-49). Dabei markiert der rote Punkt die nach Gleichung (7.4-49) ausgewählte Iteration. Die x-Achse repräsentiert das Performancekriterium und die y-Achse das Ähnlichkeitskriterium. Für beide Werte gilt, je näher sie am Ursprung liegen umso besser sind sie.

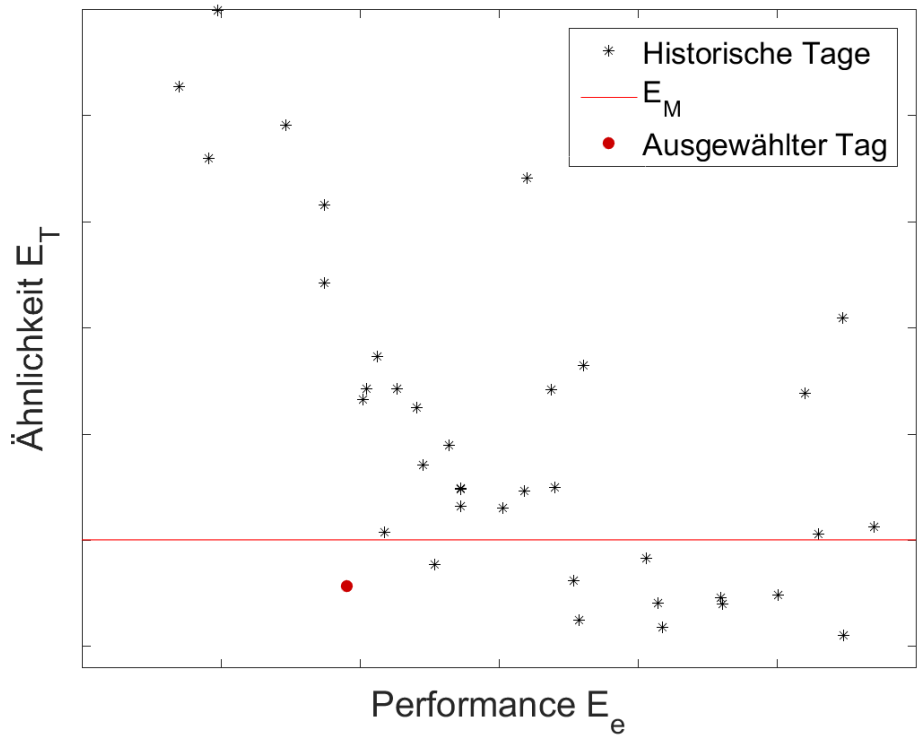
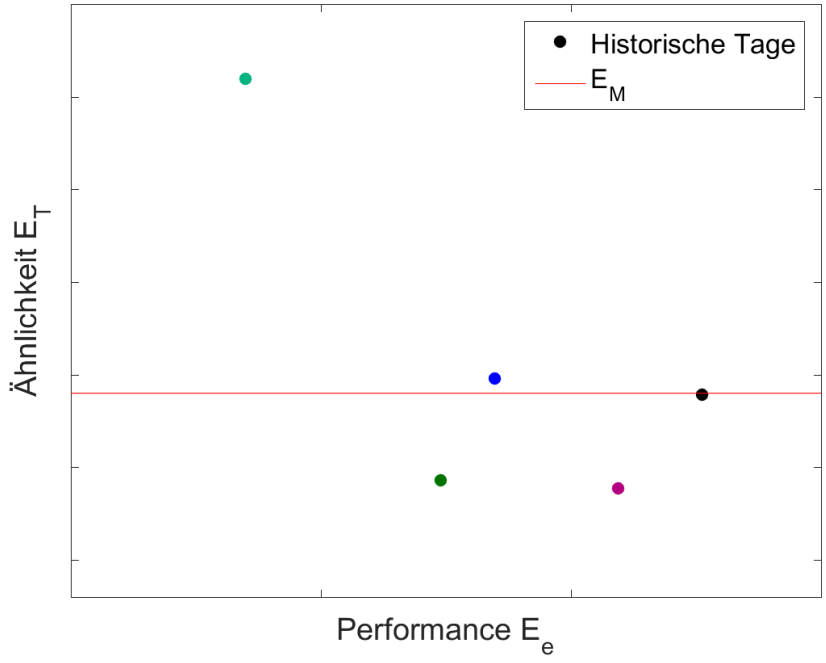


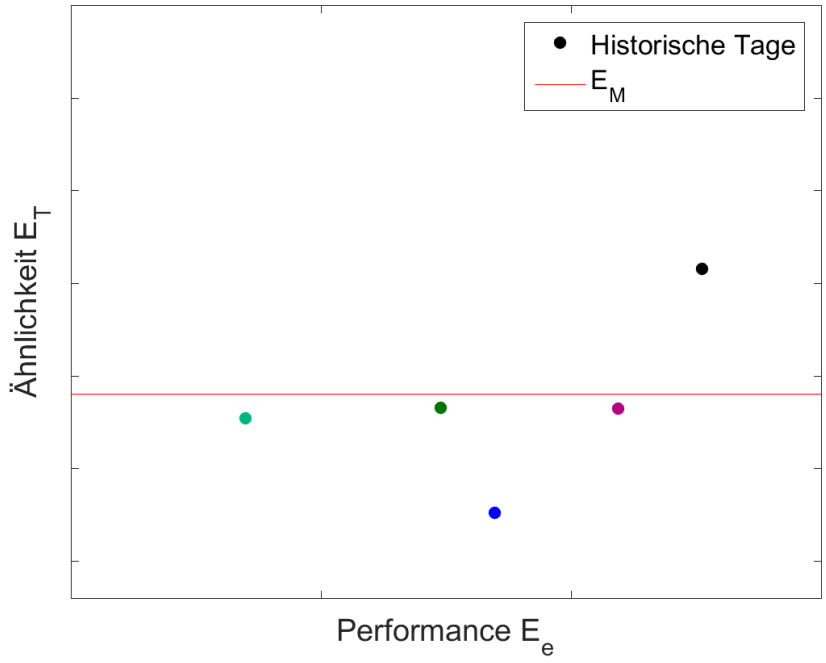
Abbildung 7.4-7: Berechnungsergebnisse des Optimierungsproblems

Die Abbildungen7.4.2 (a) und (b) zeigen beispielhaft, für fünf vergangene Iterationen und zwei unterschiedliche Vorhersagen $\mathbf{a}_{d,pre}$ wie sich die Werte der Auswertung des Optimierungsproblems (7.4-49) ändern.

Der Vergleich der beiden Bilder zeigt, dass sich die berechneten Werte auf Vertikalen des Performancekriteriums bewegen, wohingegen sich die Werte des Ähnlichkeitskriteriums für unterschiedliche Vorhersagen $\mathbf{a}_{d,pre}$ ändert.



(a) Vorhersage 1



(b) Vorhersage 2

Abbildung 7.4-8: Berechnungsergebnisse des Optimierungsproblems mit zwei unterschiedlichen Vorhersagen $\mathbf{a}_{d,pre}$

Wie die datenbasierte iterativ lernende Regelung für die modellprädiktive Regelung genutzt werden kann wird im nächsten Abschnitt dargestellt.

7.4.3 Datenbasiert lernende modellprädiktive Regelung

Für die modellprädiktive Regelung wird ein einfaches lineares zeit-diskretes Zustandsraummodel des Systems, wie es in Abschnitt 7.2.1.3 eingeführt wurde, verwendet. Das bekannte Optimierungsproblem

$$\min_{\mathbf{u}\in\mathcal{U}} J(\mathbf{u}) \tag{7.4-50}$$

der linearen zeitdiskreten modellprädiktiven Regelung wurde im Abschnitt 7.3.2 eingeführt. Auch die Kostenfunktion

$$J(\mathbf{u}) = \sum_{i=1}^{H_p} \|\mathbf{y}(k+i) - \mathbf{r}_{mpc}(k+i)\|^2_{\mathbf{Q}(i)} + \sum_{i=0}^{H_u-1} \|\Delta\mathbf{u}(k+i)\|^2_{\mathbf{R}(i)} \tag{7.4-51}$$

ist aus dem Abschnitt bereits bekannt und wird im folgenden für die Optimierung genutzt, wobei die Abweichung des Ausgangssignals $\mathbf{y}(k+i)$ von einer Referenz $\mathbf{r}_{mpc}(k+i)$ bewertet wird, sowie der Stellaufwand $\Delta\mathbf{u}(k+i)$ in jedem Zeitschritt.

Der datenbasierte ILC wie im Abschnitt 7.4.2 eingeführt, wird genutzt um eine Korrektur der Referenz $\mathbf{r}_{mpc} = \mathbf{r} + \mathbf{u}_{d,next}$ für den MPC zu berechnen. Der datenbasierte ILC nutzt die gemessenen Ausgangssignale $\mathbf{y}_d$ des Systems und die Referenz $\mathbf{r}_{ilc}$ für die Berechnung der Abweichung $\mathbf{e}_d = \mathbf{r}_{ilc} - \mathbf{y}_d$. Das Signal $\mathbf{e}_d$ wird zusammen mit dem Signal $\mathbf{u}_{d,next}$ und den Umgebungsbedingungen $\mathbf{a}_d$ einer Iteration d gespeichert, was mit jeder neuen Iteration zu einen Anstieg der zu speichernden Daten führt. Abbildung 7.4-9 zeigt eine schematische Darstellung des Systems mit einem MPC und einem datenbasierten ILC.

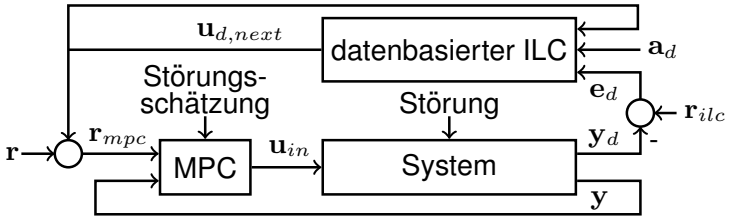


Abbildung 7.4-9: Schematisceh Darstellung eines Systems mit einem MPC und datenbasiertem ILC

Im nächsten Abschnitt wird das Konzept der datenbasierten lernenden modellprädiktiven Regelung auf ein Heizungssystem angewandt und die Simulationsergebnisse gezeigt.

7.4.4 Anwendung auf ein Heizungssystem: Simulation und Implementierung

Das betrachtete Heizungssystem beinhaltet einen Kessel als Wärmerzeuger, ein Vier-Wege-Ventil, eine Pumpe und einen Verbraucher als Wärmeabnehmer. Abbildung 7.4-10 zeigt eine schematische Darstellung des Heizungssystems mit seinen Komponenten. Der Erzeuger und der Verbraucher des Modells

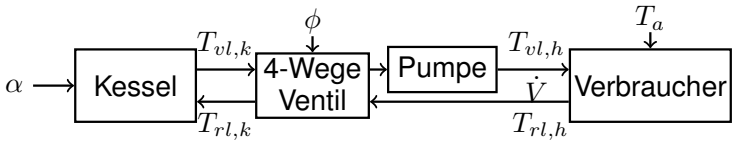


Abbildung 7.4-10: Schematische Darstellung des Heizungssystems

wurden mit Hilfe von Wärmeleistungsbilanzen erstellt und im Abschnitt 7.2.2 bereits eingeführt. Das Modell des Vier-Wege-Ventils ist ein lineares Black-Box-Modell welches mittels Messdaten geschätzt wurde und zwei Zustände, drei Eingänge und zwei Ausgänge hat. Die Eingänge sind die Vorlauftemperatur des Kessels $T_{vl,k}$, die Rücklauftemperatur der Heizkörper $T_{rl,h}$ und dem Stellsignal des Ventils $\phi \in [0, 1]$. Die Ausgänge sind die Vorlauftemperatur der Heizkörper $T_{vl,h}$ und die Rücklauftemperatur des Kessels $T_{rl,k}$.

Je nach Ventilstellung wird kaltes Wasser vom Rücklauf der Heizkörper in der Vorlauf gemischt und warmes Wasser vom Vorlauf des Kessels in den Rücklauf. Die unbekannten Parameter der Differentialgleichungen (7.2-20), (7.2-21) und (7.2-22) werden mittels Messdaten geschätzt und validiert. Das Modell hat die zwei Stellsignale α und ϕ für die Leistung des Kessels und die Stellung des Vier-Wege-Ventils als Eingänge sowie die Außentemperatur und den Volumenstrom T_a und $\dot{V}$ als zwei weitere Eingänge welche als Störung interpretiert werden. Die Vor- und Rücklauftemperatur der Heizung $T_{vl,h}$ und $T_{rl,h}$, die Raumtemperatur T_{raum} und die Vorlauftemperatur des Kessels $T_{vl,k}$ sind die vier Ausgänge des Modells.

Der modellprädiktive Regler nutzt ein linearisiertes zeitdiskretes Zustandsraummodell des Heizungssystems, um die beiden Stellsignale α und ϕ zu berechnen. Die Referenz r_{mpc} für den MPC ist die Vorlauftemperatur der Heizkörper. Die Referenz wird mit Hilfe einer Heizkurve, abhängig von der Außentemperatur T_a bzw. der Außentemperaturvorhersage $T_{a,vor}$, festgelegt. Der MPC berechnet die Stellsignale für den gesamten Vorhersagehorizont H_p in jedem Abtastschritt t_s neu. Die Stellsignale für den jeweils nächsten Zeitschritt werden an die Anlage gegeben. Eine geeignete Referenztrajektorie r_{mpc} zu finden kann schwierig sein. Aus diesem Grund wird ein datenbasierter ILC wie er im Abschnitt 7.4.2 eingeführt wurde verwendet, um auf der eine Seite die Referenztrajektorie zu optimieren und auf der anderen Seite um mögliche Modellierungsfehler welche durch die Nutzung eines einfachen linearen Modells auftreten können auszugleichen.

Die Tagesläufe der Außentemperatur bilden bei Heizungssystemen die periodischen Störungen. Abbildung 7.4.4 zeigt Tagesläufe der Außentemperatur und wie sich diese von Iteration zu Iteration bzw. von Tag zu Tag ändern, was die Verwendung eines datenbasierten iterativ lernenden Reglers nahelegt. Die Außentemperaturvorhersage, welche ebenfalls aus Messwerten generiert wurde verdeutlicht wie sich die einzelnen Tagesläufe unterscheiden, aber auch Ähnlichkeiten mit der Vorhersage aufweisen können.

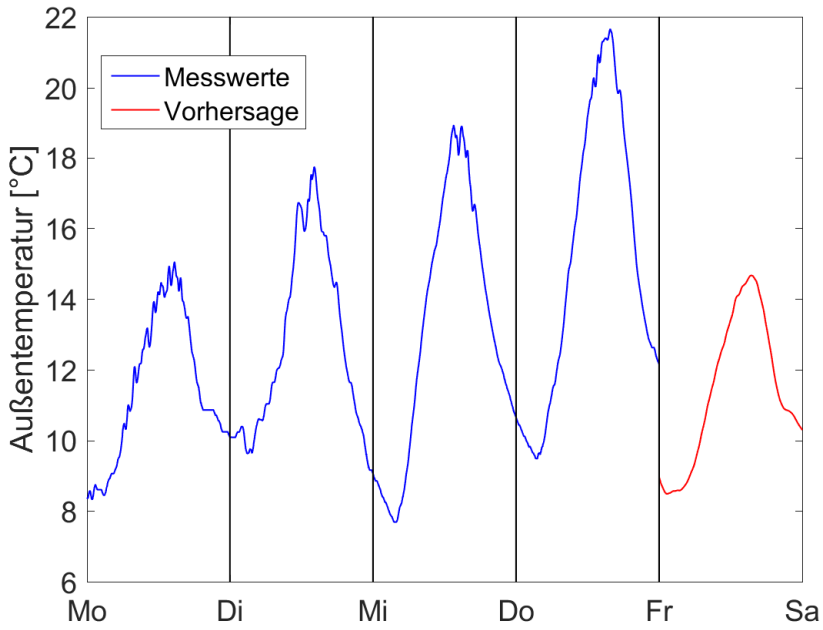


Abbildung 7.4-11: Unterschiedliche Tagesläufe der Außentemperatur

Der Vergleich der Tage zeigt, dass es vom Tageslauf der Außentemperatur mehr Sinn macht die Daten des Montags zu nehmen als vom vorherigen Tag um das Update (7.4-46) zu berechnen.

Der datenbasierte iterativ lernenden Regler optimiert die Referenz des MPC, sowie in Abbildung 7.4-9 dargestellt, um periodische Störungen durch die Außentemperaturverläufe auszugleichen, mit dem Ziel die Raumtemperatur T_{raum} in einem definierten Komfortbereich zu halten. Aus diesem Grund ist die Referenz r_{ilc} ein Profil der gewünschten Raumtemperatur mit einer Reduktion der Raumtemperatur währen der Nacht. Die Referenz der Raumtemperatur wird nach der DIN-Norm [14] festgelegt. Die Norm definiert eine Komfortraumtemperatur von 22 ± 2 °C bei einer Außentemperatur kleiner als 16 °C. Die Außentemperatur T_a wird als Umgebungsbedingung für den datenbasierten ILC genutzt und die

Vorhersage der Außentemperatur $T_{a,vor}$ um das Ähnlichkeitskriterium (7.4-47) für die Auswahl einer vergangenen Iteration zu berechnen.

Die Simulationsergebnisse mit einem MPC und einem ILC angewandt auf ein Heizungssystem werden im folgenden Abschnitt dargestellt.

7.4.4.1 Simulationsergebnisse

Für die Simulation wird das Modell verwendet welches im vorherigen Abschnitt eingeführt wurde. Die Simulation und die Programmierung erfolgte in MATLAB/Simulink, [3, 5]. Für den modellprädiktiven Regler wurde der MPC Block der Model Predictive Control Toolbox von MathWorks genutzt, [4].

Für die Simulation wird ein Vorhersagehorizont von $H_p = 5$ Stunden, ein Regelungshorizont von $H_u = 3$ Stunden, ein Verstärkungsfaktor von $\gamma = 2$ und eine Abtastzeit von $t_s = 60$ Sekunden gewählt. Die Simulation wird zweimal ausgeführt. Einmal mit dem MPC alleine und einmal mit einem zusätzlichen datenbasierten ILC, wobei alle anderen Parameter gleich gelassen werden und die selbe Heizkurve genutzt wird. Die Raumtemperaturreferenz für den datebasierten ILC wird auf 21 °C festgelegt mit einer Nachtabsenkung von 2 °C um Energie einzusparen.

Die gespeicherten Daten für $\mathbf{u}_{d,db}$ und $\mathbf{e}_{d,db}$ vergangener Iterationen für den datenbasierten ICL wurden mittels Simulation erzeugt. Ebenfalls wurden die dazugehörigen Tagesläufe der Außentemperatur abgespeichert welche auf Messdaten beruhen. Insgesamt besteht der Datensatz an historischen Daten aus 25 Wochen, also ungefähr einem halben Jahr. Nach Gleichung (7.4-49) wird die Iteration aus der Datenbank ausgewählt, welche für die Berechnung des Eingangssignals $\mathbf{u}_{d,next}$ nach Gleichung (7.4-46) verwendet wird.

Abbildung 7.4-12 zeigt, exemplarisch für den ganzen Februar, die Simulationsergebnisse von fünf Tagen. Die durchgezogene blaue Linie zeigt die obere und untere Grenze des Komfortbereichs für die Raumtemperatur. In diesem Korridor sollte sich die Raumtemperatur T_{raum} während der Nutzungszeiten befinden. Da es sich um ein Nichtwohngebäude handelt ist die Nutzungszeit auf den Tag beschränkt und während der Nacht darf die Temperatur absinken. Die gestrichelte rote Linie zeigt das Ergebnis wenn der modellprädiktive Regler alleine läuft und die punkt-gestrichelte schwarze Linie zeigt das Ergebnis mit einem zusätzlichen datenbasierten ILC.

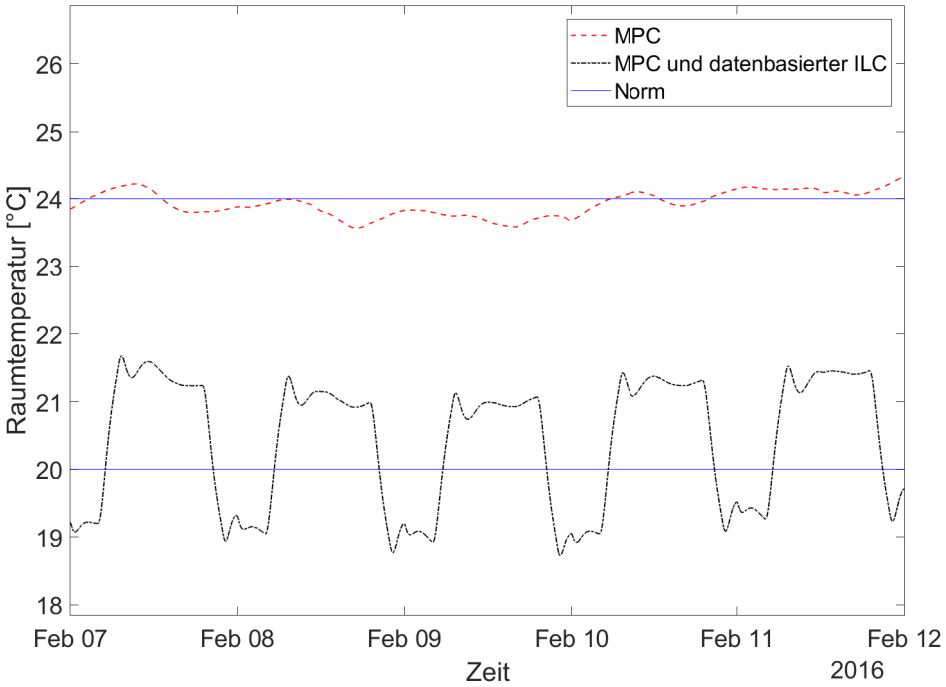


Abbildung 7.4-12: Vergleich der Simulaitonsergebnisse mit MPC und datenbasiertem ILC

Der Vergleich der Simulationsergebnisse zeigt, das die Raumtemperatur T_{raum} reduziert werden kann wenn der zusätzliche datenbasierte ILC genutzt wird und die Raumtemperatur trotzdem in der definierten Komfortzone gehalten wird. Zusätzlich wird durch die Wahl der Referenz für die Raumtemperatur eine Nachtabsenkung realisiert. Die niedrigeren Raumtemperaturen führen zu einem geringeren Energieverbrauch, da der Wärmebedarf mit der steigenden und fallenden Raumtemperatur korreliert. Das heißt, je tiefer die Raumtemperatur gewählt wird umso mehr Energie wird gespart. Auf der anderen Seite muss der Nutzerkomfort im Blick behalten werden, welcher durch die DIN-Norm vorgegeben ist. Weitere Ergebnisse sind in dem Paper [27] veröffentlicht. Im nächsten Abschnitt werden die Messergebnisse der Implementierung an einer realen Testanlage vorgestellt.

7.4.4.2 Implementierungsergebnisse

Die datenbasierte lernende modellprädiktive Regelung, wie in Abschnitt 7.4.3 eingeführt, wurde bei einer Testanlage in einem Büro eines Nichtwohngebäudes implementiert. Die Testanlage wurde von Kieback&Peter zur Verfügung gestellt und betreut. Im Kapitel 8 „AP A.5: Implementierung und Evaluation in Gebäudeautomationssystemen“ wird die Testanlage detailliert beschrieben. Die Implementierung erfolgte in enger Zusammenarbeit mit Kieback&Peter. Die Abbildung 7.4-13 zeigt den Aufbau des realen Heizungssystems und Abbildung 7.4-10 eine schematische Darstellung.

Abbildung 7.4-13: Abbildung des Teststandes

Für die Implementierung wurde das Modell verwendet, welches im vorherigen Abschnitt eingeführt und ebenfalls für die Simulation im Abschnitt 7.4.4.1 genutzt wurde. Der MPC, genauso wie der datenbasierte und wurde mittels Codegenerierung auf die Hardwareplattform gebracht. Als Hardwareplattform für die Implementierung des Reglers diente das Echtzeitsystem von National Instruments welches im Abschnitt 7.7 vorgestellt wird und mit labVIEW programmiert wird. Das Echtzeitsystem kommuniziert über das BACnet Protokoll mit der DDC (Direct-Digital-Control) des Heizungssystems und liest auch die Werte der Wetterprognose über das BACnet Protokoll aus. Die Wetterprognosedaten wurden von einer Wetterstation zur Verfügung gestellt wie sie im Abschnitt 7.7 beschrieben wird. Die Infrastruktur für die Implementierung wurde von Kieback&Peter installiert und zur Verfügung gestellt.

Für den Echtzeittest wurde die Abtastzeit auf von $t_s = 60$ Sekunden gewählt und ein Vorhersagehorizont von $H_p = 2$ Stunden, sowie einen Regelungshorizont von $H_u = 1$ Stunde. Für den MPC wurde die selbe Heizkurve wie für die Simulation in Abschnitt 7.4.4.1 verwendet um die Referenz der Vorlauftemperatur des Heizkörpers festzulegen. Für den datenbasierten ILC wurde die Referenz für die Raumtemperatur auf 22 °C, mit einer Nachtabsenkung um 2 °C festgelegt.

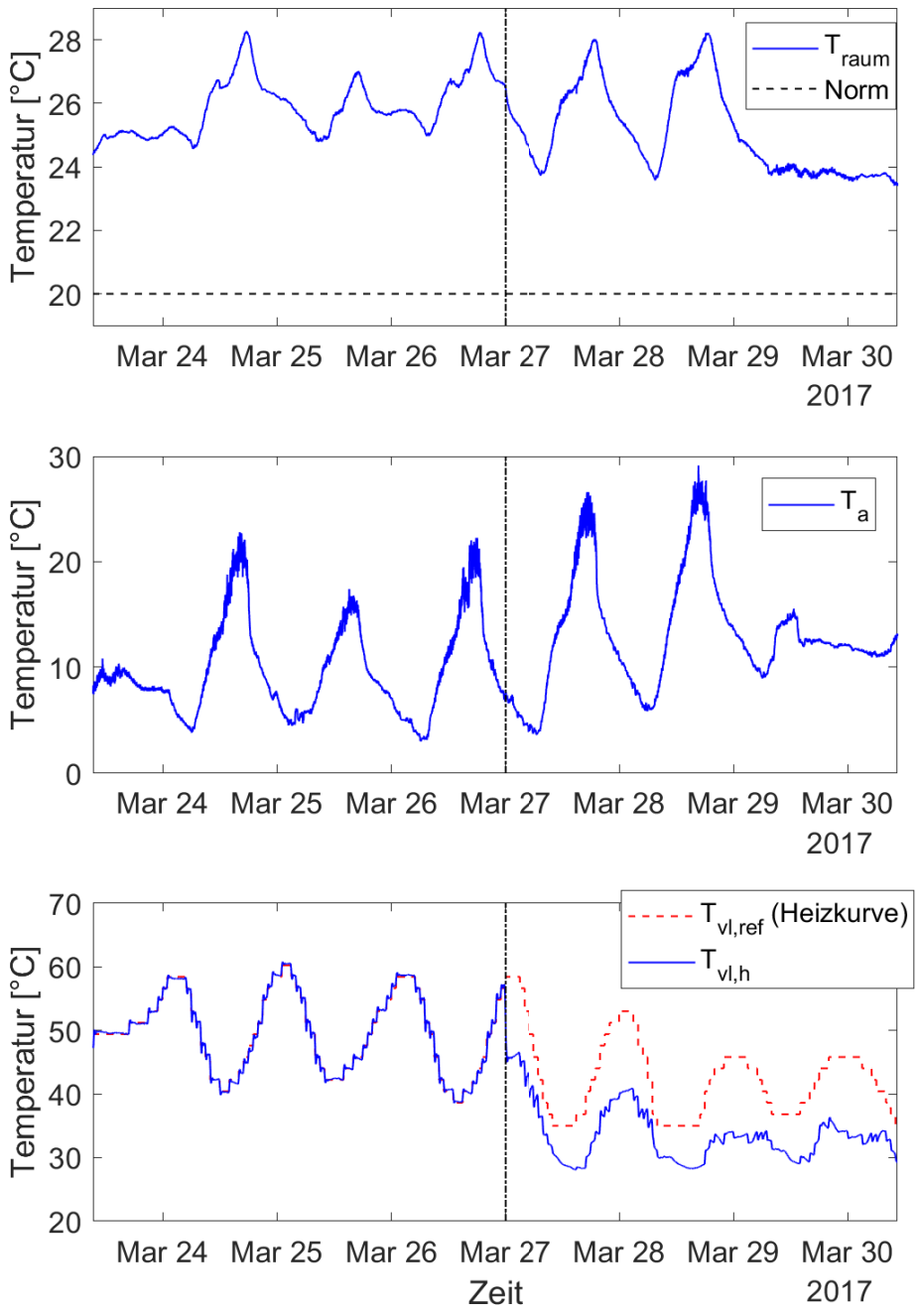


Abbildung 7.4-14: Messergebnisse des datenbasiert lernenden modellprädiktiven Regler eines Heizungssystems

Abbildung 7.4-14 zeigt die ersten Messergebnisse der Heizungsanlage von sechs Tagen. Bei den ersten drei Tagen läuft der MPC noch ohne den datenbasierten ILC. Das liegt zum einen daran, dass der datenbasierte ILC zumindest ein paar Daten von vergangenen Tagen (Iterationen) braucht um das Eingangssignal gemäß Gleichung (7.4-46) für den nächsten Tag berechnen zu können. Auf der anderen Seite gibt es so Messdaten, mit und ohne datenbasierte ILC welche verglichen werden können. Der datenbasierte ILC wurde am 27.3. gestartet und ist durch die vertikale Linie gekennzeichnet. Schaut man nur auf die Raumtemperaturen T_{raum}, so stellt man auf den ersten Blick keine signifikanten Unterschiede für die Tagestemperaturen fest. Der Vergleich mit dem Profil de Außentemperatur T_a zeigt, dass die höheren Raumtemperaturen auf Grund des Einflusses der Außentemperatur zustande kommen. Ein gezielter Blick auf die Raumtemperaturen bei Nacht zeigt, dass ab dem 27.3. niedriger Raumtemperaturen

bei ähnlichen Außentemperaturen erreicht werden. Dies kann auf den Einfluss des datenbasierten ILC zurückgeführt werden. Dies wird besonders bei der Betrachtung der Vorlauftemperaturen $T_{vl,h}$ deutlich. Die rote gestrichelte Linie in Abbildung 7.4-14 zeigt die Referenz welche durch die Heizkurve bestimmt wurde. Die blaue Linie stellt die gemessenen Vorlauftemperaturen $T_{vl,h}$ dar. Für die ersten drei Tage folgt die Vorlauftemperatur der Referenz wie erwartet, da nur der MPC läuft. Ab dem Zeitpunkt mit dem datenbasierten ILC wird eine deutlich niedrigere Vorlauftemperatur erreicht. Das heißt, durch den datenbasierten ILC wir die Referenz für die Vorlauftemperatur, auf Grund der zu hohen Raumtemperaturen nach unten korrigiert und der datenbasierte ILC reagiert wie erwartet.

Auch mit einem Testzeitraum von sechs Tagen und historischen Daten von anfangs drei Tagen, haben die Messergebnisse gezeigt, dass der datenbasierte ILC wie erwartet mit einer Absenkung der Referenz der Vorlauftemperatur auf die hohen Raumtemperaturen reagiert. Weiterführende Tests über einen längeren Zeitraum und bei tieferen Außentemperaturen währen hilfreich um das Verhalten noch genauer analysieren zu können.

Durch das Speichern jeder vergangenen Iteration nimmt der Speicherbedarf stetig zu, sodass eine Reduktion des Speicherbedarfs gerade im Hinblick auf Zielplattformen die über keinen Datenbankzugriff oder großen internen Speicher verfügen sinnvoll ist. Aus diesem Grund wird im nächsten Abschnitt auf Grundlagen von Tensorzerlegunsverfahren eine Möglichkeit zur Datenreduktion betrachtet und auf das Heizungssystembeispiel angewandt.

7.5 Tensordarstellung für datenbasierte iterativ lernende Regelung

Tensoren und die Tensorzerlegungen werden als mathematisches Werkzeug in unterschiedlichen Bereichen verwendet, wie der Modellbildung, der Fehlererkennung und Diagnose oder der Datenreduktion [21, 24, 25, 33, 32, 41, 34, 39].

Der Algorithmus eines datenbasierten iterativ lernenden Reglers wie er in Abschnitt 7.4 eingeführt wurde, speichert die Daten aller historischen Iterationen des Reglers. Das bedeutet, je länger dieser Regler läuft umso mehr Daten müssen gespeichert werden. Nicht jede Zielplattform für die Implementierung verfügt über einen Datenbankzugriff oder einen großen internen Speicher. Das heißt, für die Anwendung wäre es sehr nützlich wenn der Speicherbedarf reduziert werden kann und somit eine Anwendung auf solchen Zielplattformen möglich wird. Verfahren der Tensorzerlegung stellen eine Möglichkeit dar die Anzahl der zu Speichernden Elemente zu reduzieren. Im allgemeinen können Daten beliebig umgeordnet und in einem Tensor gespeichert werden. Des weiteren können Strukturinformationen über die Daten genutzt werden um die Dimensionen des Tensors in einer nützlichen Weise anzupassen. Im Folgenden wird gezeigt wie die Tensorzerlegung für die datenbasierte iterativ lernende Regelung genutzt werden kann. Dazu werden als erstes die benötigten mathematischen Tensoroperationen eingeführt.

7.5.1 Tensoralgebra und kanonisch-polyadische Dekomposition

Es werden allgemein gebräuchliche Definitionen eines Tensors sowie der mathematischen Operationen in diesem Kapitel verwendet und können unteranderem in [11] oder [21] gefunden werden. Für die Tensoren wird eine Notation genutzt wie sie in MATLAB üblich ist.

Definition 7.5.1 *Ein Tensor der Ordnung n ist ein n-dimensionales Array*

$$\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_n}. \tag{7.5-52}$$

Die Elemente $x(i_1, i_2, \ldots, i_n)$ des Tensors werden durch $i_j \in \{1, 2, \ldots, I_j\}$ indiziert, mit $j = 1, \ldots, n$.

Definition 7.5.2 *Das Äußere-Produkt zweier Tensoren $\mathcal{X}$ und $\mathcal{Y}$ der Dimension $I_1 \times I_2 \times \cdots \times I_n$ und $J_1 \times J_2 \times \cdots \times J_m$ ist wie folgt definiert*

$$\mathcal{Z} = \mathcal{X} \circ \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_n \times J_1 \times J_2 \times \cdots \times J_m}, \tag{7.5-53}$$

mit den Elementen des Tensors $\mathcal{Z}$

$$z(i_1, \ldots, i_n, j_1, \ldots, j_m) = x(i_1, \ldots, i_n) y(j_1, \ldots, j_m). \tag{7.5-54}$$

Beispiel 7.5.1 *Das Äußere-Produkt zweier Vektoren $\mathbf{a} \in \mathbb{R}^I$ und $\mathbf{b} \in \mathbb{R}^J$ führen auf die Matrix*

$$\mathbf{C} = \mathbf{a} \circ \mathbf{b} = \begin{pmatrix} a_1 \\ \vdots \\ a_I \end{pmatrix} \begin{pmatrix} b_1 & \cdots & b_J \end{pmatrix} = \begin{pmatrix} a_1 b_1 & \cdots & a_1 b_J \\ a_2 b_1 & \cdots & a_2 b_J \\ \vdots & \vdots & \vdots \\ a_I b_1 & \cdots & a_I b_J \end{pmatrix} \in \mathbb{R}^{I \times J}.$$

Definition 7.5.3 *Das Hadamard-Produkt zweier Tensoren $\mathcal{X}$ und $\mathcal{Y}$ der selben Dimension $I_1 \times I_2 \times \cdots \times I_n$ ist als das elementweise Produkt zweier Tensoren definiert. Das Ergebnis ist wieder ein Tensor*

$$\mathcal{Z} = \mathcal{X} \circledast \mathcal{Y} \in \mathbb{R}^{I_1 \times \cdots \times I_n} \tag{7.5-55}$$

mit den Elementen

$$z(i_1, \ldots, i_n) = x(i_1, \ldots, i_n) y(i_1, \ldots, i_n). \tag{7.5-56}$$

Beispiel 7.5.2 *Das Hadamard-Produkt zweier Matrizen $\mathbf{A} \in \mathbb{R}^{I\times J}$ und $\mathbf{A} \in \mathbb{R}^{I\times J}$ ist wieder eine Matrix*

$$\mathbf{C} = \mathbf{A} \circledast \mathbf{B} = \begin{pmatrix} a_{11}b_{11} & \cdots & a_{1J}b_{1J} \\ a_{21}b_{21} & \cdots & a_{2J}b_{2J} \\ \vdots & \vdots & \vdots \\ a_{I1}b_{I1} & \cdots & a_{1J}b_{1J} \end{pmatrix} \in \mathbb{R}^{I\times J}.$$

Definition 7.5.4 *Das Tensor-Vektor-Produkt der k_{ten}-Mode eines Tensors $\mathcal{X} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_n}$ und eines Vektors $\mathbf{a} \in \mathbb{R}^{I_k}$ führt zu einem Tensor*

$$\mathcal{Y} = \mathcal{X} \times_k \mathbf{a} \in \mathbb{R}^{I_1\times\cdots\times I_{k-1}\times I_{k+1}\times\cdots\times I_n} \tag{7.5-57}$$

mit den Elementen

$$y(i_1,\ldots,i_{k-1},i_{k+1},\ldots,i_n) = \sum_{i_k=1}^{I_k} x(i_1,\ldots,i_N)a(i_k). \tag{7.5-58}$$

Für die Reduzierung der Komplexität und des Speicherbdarfs von Tensoren existieren Zerleguns- und Faktorisierungsverfahren. Für viele Tensorstrukturen wie den Tensor Trains (TT), Tucker (TU), oder Kanonisch Polyadische (CP - Canonical Polyadic engl.), gibt es Dekompositionsverfahren, [11, 17, 21]. In diesem Kapitel wird der CP Tensor, sowie die CP Zerlegung betrachtet.

Definition 7.5.5 *Ein kanonisch-polyadischer Tensor $\mathcal{T} \in \mathbb{R}^{I_1\times\cdots\times I_n}$ wird durch die Faktormatrizen $\mathbf{T}_i \in \mathbb{R}^{I_i\times r}$ dargestellt, wobei r der Rang der Zerlegung ist und $\boldsymbol{\lambda}$ der Wichtungsvektor*

$$\mathcal{T} = [\mathbf{T}_1, \mathbf{T}_2, \ldots, \mathbf{T}_n] \cdot \boldsymbol{\lambda} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_n}. \tag{7.5-59}$$

Die Summe der Äußeren-Produkte der Spaltenvektoren $\mathbf{t}_i(j) \in \mathbb{R}^{I_i}$ der Faktormatrizen $\mathbf{T}_i \in \mathbb{R}^{I_i\times r}$ gewichtet mit dem Faktoren des Vektors $\boldsymbol{\lambda}$ bilden den Tensor

$$\mathcal{T} = \sum_{j=1}^{r} \lambda(j)\mathbf{t}_1(j) \circ \mathbf{t}_2(j) \circ \cdots \circ \mathbf{t}_n(j). \tag{7.5-60}$$

Die elementweise Darstellung ist gegeben durch

$$T(i_1,\ldots,i_n) = \sum_{j=1}^{r} \lambda(j)\; t_1(i_1,j)\ldots t_n(i_n,j). \tag{7.5-61}$$

Wenn keine Wichtungsvektor gegeben ist, wird angenommen das jedes Element des Vektors eins ist $\boldsymbol{\lambda} = (1 \quad 1 \quad \cdots \quad 1)^T$.

Abbildung 7.5.1 zeigt eine grafische Darstellung der CP Zerlegung eines dreidimensionalen Tensors.

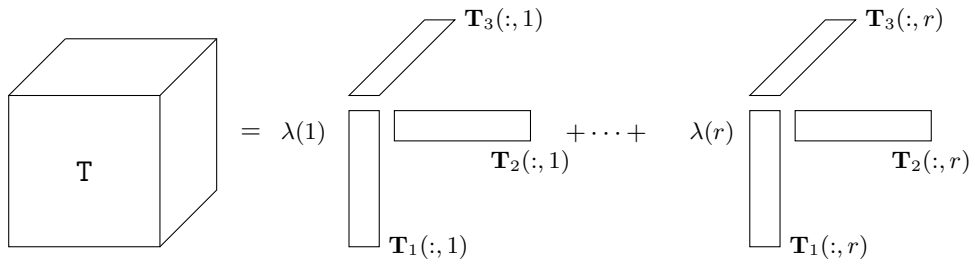


Abbildung 7.5-15: Grafische Darstellung der CP Zerlegung

7.5.1.1 Quadrat eines CP Tensors

Das Quadrat eines CP Tensors $\mathtt{T} \in \mathbb{R}^{I_1 \times \cdots \times I_n}$ mit den Faktormatrizen $\mathbf{T}_i \in \mathbb{R}^{I_i \times r}$, dem Rang r und $i = 1, \ldots, n$, ist wieder ein CP Tensor mit den neuen Faktormatrizen $\mathbf{S}_i \in \mathbb{R}^{I_i \times r_{max}}$ und dem maximalen Rang $r_{max} = r + \frac{(r^2-r)}{2}$. Die neuen Faktormatrizen des CP Tensors $\mathtt{S}$ können wie folgt berechnet werden

$$\mathtt{S}(i_1, \cdots, i_n) := \mathtt{T}^2(i_1, \cdots, i_n) = \left(\sum_{j=1}^{r} t_1(i_1, j) \cdots t_n(i_n, j) \right)^2 \tag{7.5-62}$$

$$= \sum_{j=1}^{r} t_1^2(i_1, j) \cdots t_n^2(i_n, j) \tag{7.5-63}$$

$$+ 2 \sum_{j<k}^{r} t_1(i_1, j) t_1(i_1, k) \cdots t_n(i_n, j) t_n(i_n, k) \tag{7.5-64}$$

Alle quadratischen Terme werden durch die erste Summe (7.5-63) dargestellt und können durch das Äußere-Produkt der ersten r Spaltenvektoren $\mathbf{s}_i(j)$ der Faktormatrizen $\mathbf{S}_i$, mit $i = 1, \ldots, n$ und $j = 1, \ldots, r$, berechnet werden

$$\sum_{j=1}^{r} \mathbf{s}_1(j) \circ \cdots \circ \mathbf{s}_n(j). \tag{7.5-65}$$

Die Spaltenvektoren $\mathbf{s}_i(j)$ werden wie folgt berechnet

$$\begin{aligned}
\mathbf{s}_1(j) &= \mathbf{t}_1(j) \circledast \mathbf{t}_1(j), \qquad j = 1, \ldots, r \\
\mathbf{s}_2(j) &= \mathbf{t}_2(j) \circledast \mathbf{t}_2(j), \qquad j = 1, \ldots, r \\
&\vdots \quad \vdots \quad \vdots \\
\mathbf{s}_n(j) &= \mathbf{t}_n(j) \circledast \mathbf{t}_n(j), \qquad j = 1, \ldots, r.
\end{aligned}$$

Die Mischterme werden durch die zweite Summe (7.5-64) dargestellt. Auch diese können als das Äußere-Produkt der letzten $\frac{r^2-r}{2}$ Spaltenvektoren $\mathbf{s}_i(j)$, $i = 1, \ldots, n$ und $j = r+1, \ldots, r_{max}$, der Faktormatrizen $\mathbf{S}_i$, berechnet werden

$$\sum_{j=r+1}^{r_{max}} \mathbf{s}_1(j) \circ \cdots \circ \mathbf{s}_n(j). \tag{7.5-66}$$

Die Spaltenvektoren $\mathbf{s}_i(j)$ werden wie folgt berechnet

$$\begin{aligned}
\mathbf{s}_1(r+1) &= 2\mathbf{t}_1(1) \circledast \mathbf{t}_1(2), \cdots, \mathbf{s}_1(r_{max}) = 2\mathbf{t}_1(r-1) \circledast \mathbf{t}_1(r) \\
\mathbf{s}_2(r+1) &= \mathbf{t}_2(1) \circledast \mathbf{t}_2(2), \cdots, \mathbf{s}_2(r_{max}) = \mathbf{t}_2(r-1) \circledast \mathbf{t}_2(r) \\
&\vdots \qquad \vdots \qquad \vdots \qquad \vdots \qquad \vdots \\
\mathbf{s}_n(r+1) &= \mathbf{t}_n(1) \circledast \mathbf{t}_n(2), \cdots, \mathbf{s}_n(r_{max}) = \mathbf{t}_n(r-1) \circledast \mathbf{t}_n(r).
\end{aligned}$$

Das Bedeutet, dass der CP Tensor $\mathtt{S}$ gleich dem Quadrat des CP Tensors $\mathtt{T}$ ist

$$\begin{aligned}
\mathtt{T}^2 &= \left(\sum_{j=1}^{r} \mathbf{t}_1(j) \circ \cdots \circ \mathbf{t}_n(j) \right)^2 \\
&= \sum_{j=1}^{r} \mathbf{s}_1(j) \circ \cdots \circ \mathbf{s}_n(j) + \sum_{j=r+1}^{r_{max}} \mathbf{s}_1(j) \circ \cdots \circ \mathbf{s}_n(j) = \mathtt{S}.
\end{aligned} \tag{7.5-67}$$

Das Ergebnis der vorangegangenen Untersuchung zeigt, dass das Quadrat eines CP Tensors wieder ein CP Tensor mit einem neuen Rang ist. Die Berechnungen des Quadrates eines CP Tensors kann folglich

mit den Faktormatrizen erfolgen ohne den vollen Tensor zu berechnen. Daraus folgt, dass die Vorteile bezüglich des Speicherbedarfs eines Zerlegten CP Tensors erhalten bleiben. Ein höherer Rang der CP Zerlegung führt zu einen höheren Rang des quadrierten Tensors aber die Dimensionalität des Tensors ändert sich nicht.

Beispiel 7.5.3 *Das Quadrieren eines dreidimensionalen CP Tensors $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ mit dem Rang $r = 2$ führt zu einem dreidimensionalen CP Tensor $\mathcal{S} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ mit dem neuen Rang $r_{max} = 3$*

$$\begin{aligned}\mathcal{T}^2(i_1, i_2, i_3) &= \left(\sum_{j=1}^{2} t_1(i_1, j) t_2(i_2, j) t_3(i_3, j) \right)^2 \\ &= (t_1^2(i_1, 1) t_2^2(i_2, 1) t_3^2(i_3, 1) \\ &+ 2 t_1(i_1, 1) t_1(i_1, 2) t_2(i_2, 1) t_2(i_2, 2) t_3(i_3, 1) t_3(i_3, 2) \\ &+ t_1^2(i_1, 2) t_2^2(i_2, 2) t_3^2(i_3, 2)) \\ &= \mathcal{S}(i_1, i_2, i_3).\end{aligned}$$

Die Faktormatrizen $\mathbf{S}_i \in \mathbb{R}^{I_i \times 3}$, $i = 1, 2, 3$ ergeben sich zu

$$\begin{aligned}\mathbf{S}_1 &= [\mathbf{t}_1(1) \circledast \mathbf{t}_1(1), \mathbf{t}_1(2) \circledast \mathbf{t}_1(2), 2\mathbf{t}_1(1) \circledast \mathbf{t}_1(2)] \\ \mathbf{S}_2 &= [\mathbf{t}_2(1) \circledast \mathbf{t}_2(1), \mathbf{t}_2(2) \circledast \mathbf{t}_2(2), \mathbf{t}_2(1) \circledast \mathbf{t}_2(2)] \\ \mathbf{S}_3 &= [\mathbf{t}_3(1) \circledast \mathbf{t}_3(1), \mathbf{t}_3(2) \circledast \mathbf{t}_3(2), \mathbf{t}_3(1) \circledast \mathbf{t}_3(2)]\end{aligned}$$

Nachdem die für dieses Kapitel benötigten Grundlagen der Tensoralgebra eingeführt wurden, zeigt der nächsten Abschnitt wie saisonale Daten in Tensoren abgelegt werden können, wobei Strukturinformationen für die Dimensionalität des Tensors genutzt werden.

7.5.2 Ähnlicheitskriterium auf Basis von saisonal strukturierten Daten in Tensoren

Im allgemeinen können Daten in einer beliebigen Art und Weise in einer Tensorstruktur abgelegt werden. Wenn es allerdings zusätzliche Informationen über die Daten gibt können diese Verwendet werden um eine logische und für die weitere Nutzung hilfreiche Tensorstruktur zu erhalten. Für viele Anwendungen spielen Umgebungsbedingungen eine entscheidende Rolle, z.B. die Außentemperatur für Heizungssysteme. Im Bereich der Heizungssysteme gibt es durch das Nutzerverhalten Abhängigkeiten vom Wochentag, z.B. sind Daten von einem Dienstag ähnlich wie die eines anderen Dienstages und weniger wie die eines Samstags usw. Aber auch Tages und Jahresläufe spielen eine entscheidende Rolle. Wird dies berücksichtigt ergibt sich eine vierdimensionale Tensorstruktur $\tilde{\mathtt{T}} \in \mathbb{R}^{I_1 \times I_2 \times I_3 \times I_4}$ für die Datenspeicherung.

Die Daten eines Tages sind in der ersten Dimension I_1 abgelegt, z.B. 1440 Datenpunkte bei einer Abtastrate von einer Minute. Die zweite Dimension I_2 ist die Anzahl der Tage einer Woche. Die dritte Dimension I_3 ist die Anzahl der Wochen eines Jahres und die vierte Dimension ist die Anzahl der gespeicherten Jahre. Das bedeutet für einen Datensatz mit Jahresdaten und einer Abtastzeit von einer Minuten ergibt sich ein Tensor der Dimension $1440 \times 7 \times 52 \times 1$ mit 524160 Elementen. Das heißt, im Vergleich zu einer Speicherung der Daten in einer Zeitreihe stellt die Ablage der Daten in einem Tensor lediglich eine Umstrukturierung der Daten dar und die volle Information bleibt erhalten. Mit Hilfe der CP Tensorzerlegung kann die Anzahl der zu speichernden Elemente signifikant reduziert werden. Wird ein CP Zerlegungsalgorithmus auf den Tensor $\tilde{\mathtt{T}}$ angewandt, erhält man einen CP zerlegten Tensor $\mathtt{T}$ des Ranges r mit den vier Faktormatrizen $\mathbf{T}_i$ ($i = 1, 2, 3, 4$), wie er nach (7.5.5) definiert wurde.

Um einen solchen CP Tensor für einen datenbasierten ILC nutzen zu können, wird im folgenden gezeigt wie das Ähnlichkeitskriterium (7.4-47) in CP Darstellung berechnet werden kann, ohne die volle Darstellung des Tensors $\mathtt{T}$ berechnen zu müssen. Das heißt, für die Berechnungen werden lediglich die Faktormatrizen $\mathbf{T}_i$ genutzt. Die Daten sind Tagesweise strukturiert und eine Iteration des datenbasierten iterativ lernenden Reglers ist gleich einem Tag d, welcher durch die erste Dimension I_1 repräsentiert wird und mit Hilfe der Indices i_2, i_3, i_4 ausgewählt werden kann.

Das Ähnlichkeitskriterium (7.4-47) in CP Darstellung ergibt sich wie folgt

$$\mathtt{E}_T(i_2,i_3,i_4) = \sum_{i_1}(\mathtt{T}(i_1,i_2,i_3,i_4) - \hat{\mathbf{t}}(i_1))^2 \tag{7.5-68}$$

$$= \sum_{i_1}(\mathtt{T}^2(i_1,i_2,i_3,i_4) - 2\mathtt{T}(i_1,i_2,i_3,i_4)\hat{\mathbf{t}}(i_1) + \hat{\mathbf{t}}^2(i_1)) \tag{7.5-69}$$

mit dem Datenvektor $\hat{\mathbf{t}}(i_1) \in \mathbb{R}^{I_1}$ welcher die Vorhersage der Umgebungsbedingungen des nächsten Tages d enthält.

Das Ergebnis der Berechnung des Ähnlichkeitskriteriums ist für alle Tage d in dem CP Tensor $\mathtt{E}_T$ der Dimension $I_2 \times I_3 \times I_4$ gespeichert. Da es sich um eine Summe handelt können die drei Terme der Gleichung (7.5-69) getrennt von einander berechnet werden.

Der erste Term $\sum_{i_1} \mathtt{T}^2(i_1,i_2,i_3,i_4)$ kann auf Basis der Faktormatrizen $\mathbf{T}_i$ berechnet werden. Dabei erhält man das Quadrat des CP Tensors $\mathtt{T}$ wie es im vorherigen Abschnitt 7.5.1 eingeführt wurde. Anschließend wird das Matrix-Vektor-Produkt (7.5.4) der ersten-Mode mit einem Vektor $\mathbf{O} \in \mathbb{R}^{I_1}$ in dem jedes Element eins ist, ausgeführt

$$\mathtt{Y}_1(i_2,i_3,i_4) = \sum_{i_1}\left(\sum_{j=1}^{r} t_1(i_1,j)t_2(i_2,j)t_3(i_3,j)t_4(i_4,j)\right)^2$$

$$= \left(\sum_{j=1}^{r} t_1(i_1,j)t_2(i_2,j)t_3(i_3,j)t_4(i_4,j)\right)^2 \times_1 \mathbf{O}.$$

Das Ergebnis ist ein CP Tensor der Dimension $I_2 \times I_3 \times I_4$. Der Algorithmus zur Berechnung des Tensor-Vektor-Produktes in CP Darstellung ohne die volle Darstellung des Tensors berechnen zu müssen ist in der Tesor Toolbox implementiert, [8].

Der zweite Term $\sum_{i_1}(-2\mathtt{T}(i_1,i_2,i_3,i_4)\hat{\mathbf{t}}(i_1))$ kann mit dem bekannten Matrix-Vektor-Produkt der ersten-Mode (7.5-57 berechnet werden

$$\mathtt{Y}_2(i_2,i_3,i_4) = -2\mathtt{T}(i_1,i_2,i_3,i_4) \times_1 \hat{\mathbf{t}}(i_1). \tag{7.5-70}$$

Das Ergebnis ist ebenfalls ein CP Tensor der Dimension $I_2 \times I_3 \times I_4$.

Der dritte Term $\sum_{i_1} \hat{\mathbf{t}}^2(i_1)$ ist für alle i_2, i_3 und i_4 konstant.

Daraus folgt, dass das Ähnlichkeitskriterium (7.4-47) auf Basis der Faktormatrizen berechnet werden kann ohne die volle Darstellung des CP Tensors zu nutzen, wodurch die Vorteile der CP Zerlegung in Form von Speicherbedarfsreduzierung erhalten bleibt. Der CP Tensor $\mathtt{E}_T \in \mathbb{R}^{I_2 \times I_3 \times I_4}$ enthält die Werte aller Tage d.

Im nächsten Abschnitt wird die CP Zerlegung anhand von Jahresdaten der Außentemperatur durchgeführt und die Größenordnung der Reduzierung des Speicherbedarfs aufgezeigt.

7.5.3 CP Tensorzerlegung am Beispiel von Jahresdaten der Außentemperatur

Wie im Abschnitt 7.4 wird die Außentemperatur zum Vergleich der Umgebungsbedingungen genutzt und für die Berechnung des Ähnlichkeitskriteriums herangezogen. Der Tensor $\tilde{\mathtt{T}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ ist ein dreidimensionaler Tensor und beinhaltet die Außentemperatur Daten eines Jahres, mit den einzelnen Elementen $t(i_1,i_2,i_3)$. Die erste Dimension I_1 enthält die Anzahl der Daten pro Tag im Minutentakt, die zweite Dimension I_2 die Anzahl der Tage pro Woche und die dritte Dimension I_3 die Wochen pro Jahr. Damit erhält man einen Tensor $\tilde{\mathtt{T}} \in \mathbb{R}^{1440 \times 7 \times 52}$ mit 524160 Elementen, welcher die Messwerte der Außentemperatur für jede Minute eines Jahres beinhaltet. Eine Näherung diese Tensors $\tilde{\mathtt{T}}$, welcher die Messwerte enthält, wird durch die Faktormatrizen der CP Zerlegung dargestellt. Die Algorithmen für die CP Zerlegung sind, z.B. in der Tensor oder Tensorlab Toolbox für MATLAB enthalten,[8, 42].

Die Reduktion des Speicherbedarfs hängt direkt von dem gewählten Rang r der CP zerlegung ab, da die

Dimension der Faktormatrizen von dem Rang abhängt, siehe Definition (7.5.5). Das führt für eine Rang eins ($r = 1$) CP Zerlegung des Tensors $\tilde{\mathrm{T}}$ auf die drei Faktormatrizen

$$\mathbf{T}_1 \in \mathbb{R}^{1440\times 1}, \mathbf{T}_2 \in \mathbb{R}^{7\times 1}, \mathbf{T}_3 \in \mathbb{R}^{52\times 1}.$$

Damit haben die drei Faktormaritzen insgesamt 1499 Elemente welche gespeichert werden müssen. Im Vergleich zu dem Datentensor $\tilde{\mathrm{T}}$ mit 524160 bedeutet dies eine Reduzierung der zu speichernden Elemente um den Faktor ≈ 350. Wird ein Rang von zwanzig ($r = 20$) gewählt für die CP Zerlegung gewählt führt dies zu Faktormatrizen der Dimension 1440×20, 7×20 und 52×20 und somit zu 29980 Elementen die gespeichert werden müssen und damit zu einer Reduzierung um einen Faktor von ≈ 17.5. Diese Überlegungen zeigen, dass der Rang der CP Zerlegung so klein wie möglich gewählt werden sollte um eine hohe Reduktion des Speicherbedarfs zu erzielen. Auf der anderen Seite gibt es einen Trade-off zwischen dem relativen Fehler, welcher von der Differenz zwischen dem ursprünglichen Datentensor $\tilde{\mathrm{T}}$ und dem CP Tensor T abhängt, und dem Rang r der Zerlegung.

Abbildung 7.5-16 zeigt den Zusammenhang zwischen dem Rang und dem relativen Fehler der CP Zerlegung des Datentensors $\tilde{\mathrm{T}}$.

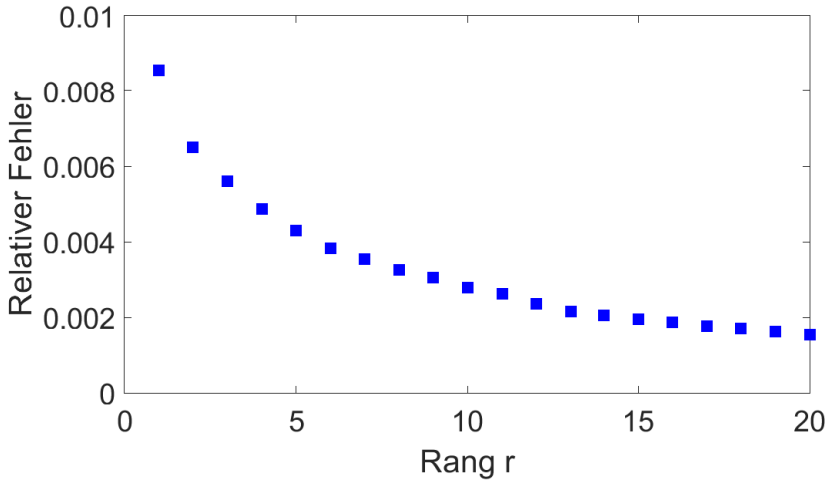


Abbildung 7.5-16: Relativer Fehler in Abhängigkeit des Ranges

Für die folgenden Untersuchungen wird eine CP Zerlegung mit einem Rang fünf $r = 5$ verwendet um eine gute Approximation der Außentemperatur zu erhalten bei gleichzeitiger signifikanter Reduzierung des Speicherbedarfs. Damit ergeben sich für die Faktormatrizen die Dimensionen 1440×5, 7×5 und 52×5. Das heißt, es sind 7495 Elemente zu Speichern was eine Reduzierung des Speicherbedarfs um einen Faktor ≈ 70 entspricht.

Wird ein Zeitraum von zwei Jahren betrachtet wird der eingesparte Speicherbedarf noch größer. Wenn historische Daten der Außentemperatur von zwei Jahren zur Verfügung stehen, führt das zu einen Messwerttensor der Dimension $\tilde{\mathrm{T}} \in \mathbb{R}^{1440\times 7\times 52\times 2}$. Das entspricht einer Anzahl von 1048320 Elementen. Im Vergleich dazu führt eine Rang fünf CP Zerlegung zu den vier Faktormatrizen $\mathbf{T}_1 \in \mathbb{R}^{1440\times 5}$, $\mathbf{T}_2 \in \mathbb{R}^{7\times 5}$, $\mathbf{T}_3 \in \mathbb{R}^{52\times 5}$, und $\mathbf{T}_4 \in \mathbb{R}^{2\times 5}$. Das bedeutet, es siind gerade einmal 7505 Elemente, was eine Speicherbedarfsreduzierung um einen Faktor von ≈ 140 entspricht. Dies kann gerade für die Anwendung entscheidend sein, wenn auf der jeweiligen Zielplattform kein großer Speicher vorhanden ist.

Die Abbildungen 7.5-17 und 7.5-18 zeigen die Jahresverläufe der Außentemperatur. Es wird die gemessene Außentemperatur eines Jahres mit der durch die CP Zerlegung approximierten Außentemperatur verglichen. Abbildungen 7.5-17 zeigt den Vergleich der Messwerte der Außentemperatur mit der Approximation einer Rang eins CP Zerlegung und Abbildungen 7.5-18 zeigt den Vergleich mit einer Rang fünf CP Zerlegung.

Im nächsten Abschnitt werden die Außentemperaturen in einem Rang fünf CP Tensor gespeichert und für die Berechnung des Ähnlichkeitskriteriums verwendet. Die Ergebnisse welche auf Basis ei-

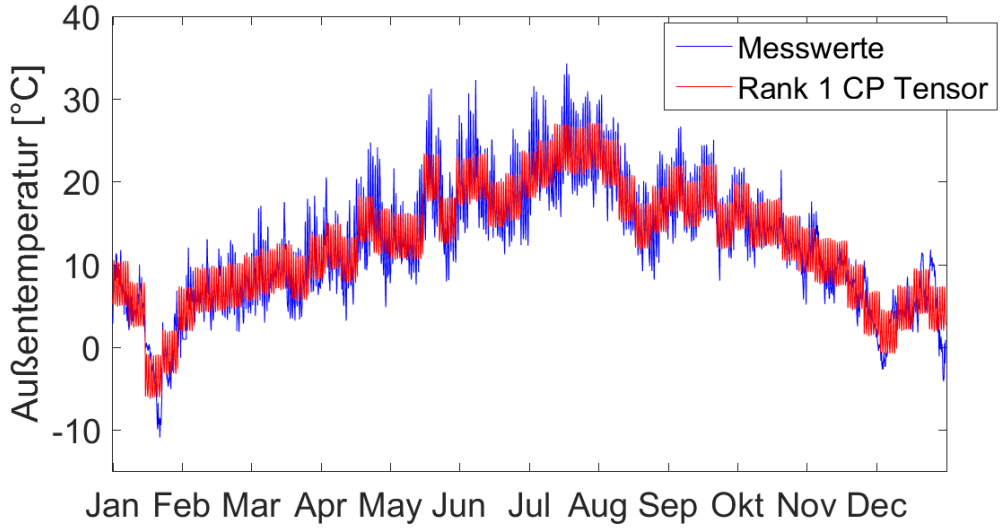


Abbildung 7.5-17: Gemessene Außentemperatur in Vergleich zur CP Approximation, $R = 1$

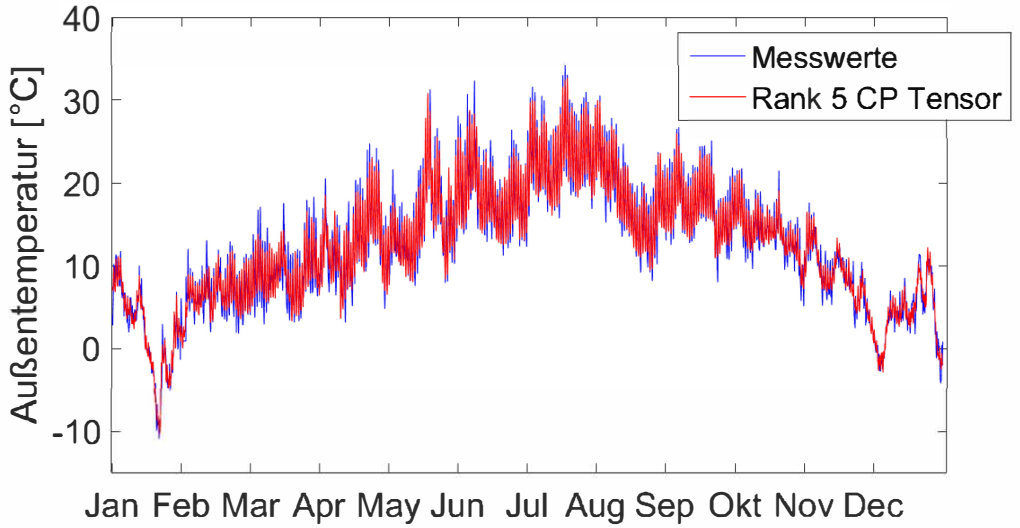


Abbildung 7.5-18: Gemessene Außentemperatur in Vergleich zur CP Approximation, $R = 5$

nes CP Tensors erzielt wurden, werden mit den Ergebnissen verglichen, welche mit dem Tensor der Außentemperaturmesswerte erzielt wurden.

7.5.4 Anwendungsbeispiel auf ein Heizungssystem

Das Anwendungsbeispiel wurde bereits in dem Abschnitt 7.4.4 eingeführt und wird für die folgenden Untersuchungen genutzt. Die Simulation wird einmal durchgeführt, in dem die Messwerte der Außentemperatur in einem Tensor $\tilde{\mathrm{T}}$ gespeichert und mit diesen Messwerten die Berechnungen des Ähnlichkeitskriteriums (7.5-69) durchgeführt werden, wie es in Abschnitt 7.4 eingeführt wurde. Die Ergebnisse werden mit den Simulationsergebnissen verglichen bei denen die Außentemperaturen in einem Zerlegten CP Tensor T gespeichert wurden und die Berechnungen des Ähnlichkeitskriteriums nach Gleichung 7.5-69 auf Basis der Faktormatrizen erfolgte. Die Vorhersage der Außentemperatur wurde in dem Vektor $\hat{\mathbf{t}}$ gespeichert.

Für alle Simulationen wurde ein Vorhersagehorizont von $H_p = 5$ Stunden und ein Regelunngshorizont

von $H_u = 3$ Stunden, einen Verstärkungsfaktor von $\gamma = 2$ und eine Abtastzeit $t_s = 60$ Sekunden gewählt. Die gespeicherten Werte für das Eingangssignale und die Abweichung, $\mathbf{u}_{d,db}$ und $\mathbf{e}_{d,db}$, des datenbasierten iterativ lernenden Reglers wurden durch Simulation erzeugt. Insgesamt stehen für den Vergleich Eingangs-, Abweichungs- und Außentemperaturdaten von 175 Tagen also 25 Wochen zur Verfügung. Dies entspricht historischen Daten von ungefähr einem halben Jahr auf welche der datenbasierte ILC für die Berechnungen des nächsten Tages zugreifen kann. Für die Berechnung des ILC Updates (7.4-46) wird die Gleichung (7.4-49) verwendet um den entsprechenden Datensatz aus den historischen Daten auszuwählen. Für die CP Zerlegung der Außentemperaturdaten wurde die Tensorlab Toolbox für MATLAB verwendet, [42], wobei alle Einstellungen auf den Standardwerten gelassen wurden.

Als erstes wird das Ergebnis der Berechnungen des Auswahlkriteriums (7.4-49) mit einem Tensor mit Messwerten der Außentemperatur verglichen mit der Berechnung auf Basis eines Rang fünf und eines Rang eins CP Tensors welcher eine Approximation des Verlaufs der Außentemperatur darstellt. Dazu wurden für alle Berechnungen die selben 25 Wochen an historischen Daten verwendet. Die Berechnung wurde insgesamt 63 mal, mit unterschiedlichen Wettervorhersagen für die Außentemperatur durchgeführt. Die Ergebnisse dieser Untersuchungen sind in der Tabelle 7.5-1 zusammengefasst. Die Ergebnisse

Tabelle 7.5-1: Vergleich der Berechnung des Auswahlkriteriums: Messwerttenosr - CP Tensor

	Messwerte	CP, Rang 5	CP, Rang 1
Anzahl	63	51	14
Prozent	100	83	22

dieser Untersuchung zeigen, dass in 83 Prozent der Fälle der selbe historische Tag ausgewählt wird, wenn eine Rang fünf CP Approximation der Außentemperatur verwendet wird wie bei der Berechnung mit dem Messwerttensor. Bei einer Rang eins CP Approximation sind es dagegen lediglich 22 Prozent.

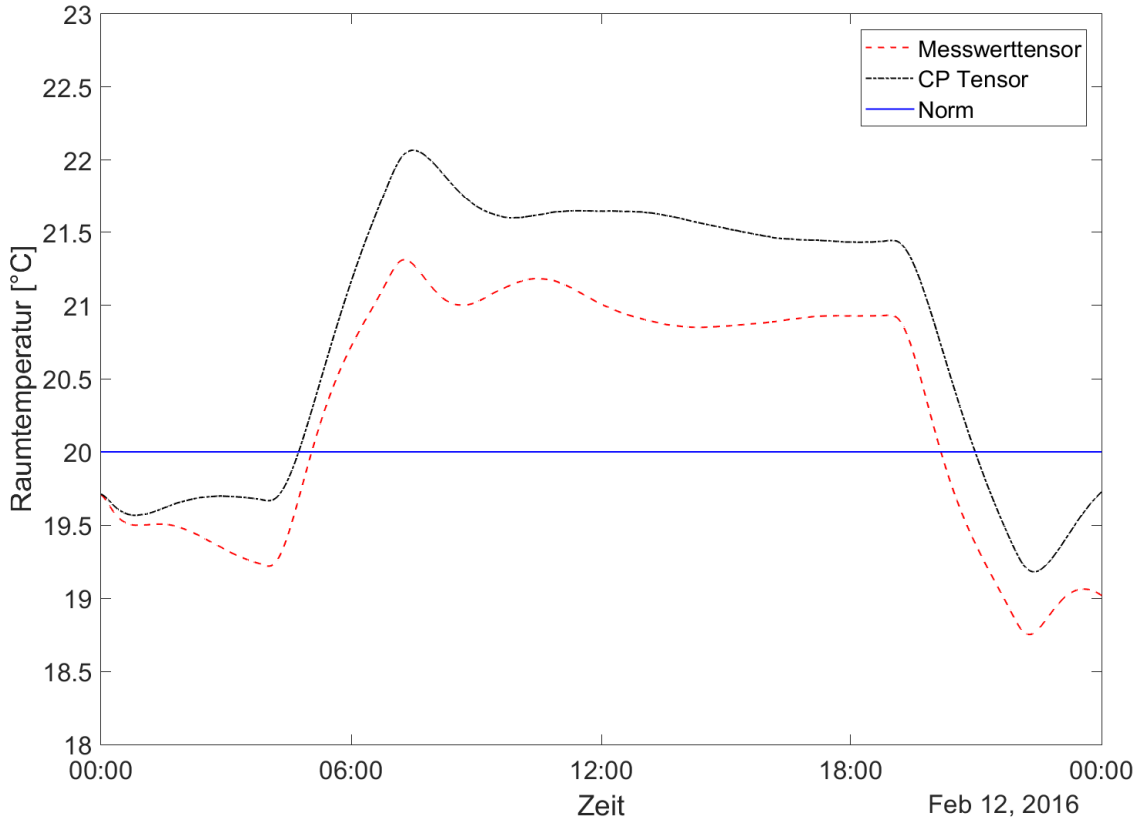


Abbildung 7.5-19: Vergleich der Raumtemperaturen bei denen unterschiedliche Tage d ausgewählt wurden

Abbildung 7.5-19 zeigt das Simulationsergebnis eines Tages, wobei die Auswertung des Auswahlkri-

teriums (7.4-49) zu unterschiedlichen Tagen d geführt. Dabei wurden die Berechnungen einmal mit dem Messwerttensor und einmal mit den Faktormatrizen eines Rang fünf CP Tensors durchgeführt. Die rote gestrichelte Linie zeigt das Simulationsergebnis wenn der Messwerttensor für die Berechnung des Auswahlkriteriums (7.4-49) genutzt wird. Die schwarze punkt-gestrichelte Linie zeigt das Ergebnis, wenn alle Parameter genauso gelassen werden, außer dass anstelle des Messwerttensor die Faktormatrizen eines Rang fünf CP Tensors für die Berechnung des Auswahlkriteriums (7.4-49) verwendet wird. Dabei wird anstelle von Gleichung (7.4-47) die Gleichung (7.5-69) für die Berechnung des Ähnlichkeitskriteriums verwendet. Die blaue durchgezogenen Linie zeigt die untere Grenze der gewünschten Raumtemperatur bzw. des definierten Komforbereichs welcher durch die DIN-Norm [14] festgelegt wurden.

Ein Vergleich der beiden Ergebnisse zeigt, dass die beiden Kurven sich die meiste Zeit um lediglich 0.5 K unterscheiden. Auch bleibt dir Raumtemperatur am Tag in dem definierten Komfortbereich. Wenn durch das Ähnlichkeitskriterium der selbe Tag ausgewählt wird, unterscheidet sich das Simulationsergebnis verständlicherweise nicht und die beiden Linien der Raumtemperaturen liegen direkt übereinander.

Der nächste Abschnitt dieses Kapitels beschäftigt sich mit der modellprädiktiven Regelung mit einer lineare Kostenfunktion dem EMCP (economic model predictive control). Auf Grund der linearen Kostenfunktion ist es möglich, anstelle von Wichtungsfaktoren auch reale Kosten in das Gütefunktional einzusetzen, also eine Optimierung unter wirtschaftlichen Gesichtspunkten durchzuführen, welche direkt die Kosten reduziert.

7.6 Modellprädiktive Regelung mit linearer Kostenfunktion

Die modellprädiktive Regelung mit linearer Kostenfunktion, auch EMPC (Economic Model Predictive Control) genannt, welche in diesem Kapitel vorgestellt wird unterscheidet sich von dem MPC aus den vorangegangenen Kapiteln in der Kostenfunktion. Im Vergleich zu einem MPC mit einem quadratischen Gütefunktional wird beim EMPC eine lineare Kostenfunktion verwendet, [15]. Auf Grund der linearen Kostenfunktion ist es möglich anstelle von Wichtungsfaktoren reale Kosten zu verwenden wenn diese bekannt sind. Damit ist es möglich direkt die Kosten eines Prozesses zu optimieren. Im Bereich der Anwendung eines EMPC auf HVAC Systeme beschäftigen sich die Veröffentlichungen im wesentlichen mit der Ausnutzung von variablen Strompreisen für die Heizungs- oder Kühlungssysteme eines Gebäudes, [18, 29, 12]. Im folgenden wird ein EMPC Ansatz am Beispiel einer Heizungsanlage vorgestellt und implementiert, welcher direkt die Heizleistung der Kessel regelt sowie die Stellsignale von Ventilen einzelner Heizkreise. Dabei wird sowohl auf kontinuierliche Stellsignale als auch auf schaltende Stellsignale eingegangen, wie sie häufig bei Kesseln vorzufinden sind.

7.6.1 EMPC für kontinuierliche Stellsignale

In diesem Abschnitt wird der EMPC für kontinuierliche Stellsignale vorgestellt. Das Prinzip der modellprädiktiven Regelung mit einem quadratischen Gütefunktion wurde in diesem Kapitel in Abschnitt 7.4.3 vorgestellt. Auch bei einem EMPC wird ein Modell des Systems genutzt um das dynamische Systemverhalten abzubilden. Der Unterschied zum linearen MPC liegt zum einen in der Kostenfunktion und zum anderen in der Referenzfolge. Bei einem EMPC wird nicht die Abweichung zu einer Referenz bewertet sondern eine Optimierung einer lineare Kostenfunktion unter Randbedingungen ausgeführt. Abbildung 7.6-20 zeigt das Schema eines Regelkreises mit einem EMPC.

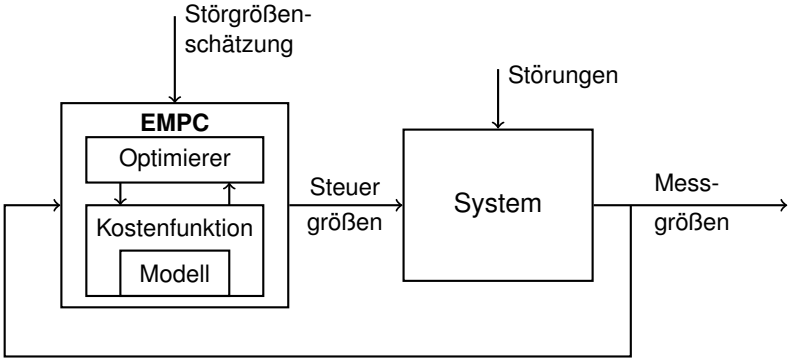


Abbildung 7.6-20: Schematische Darstellung des EMPC Regelkreises

Die lineare Kostenfunktion

$$J(\mathbf{u}) = \sum_{i=0}^{H_p-1} (c_1 u_1(k+i) + c_2 u_2(k+i) + ...) \tag{7.6-71}$$

mit den Kosten c_j und den Eingängen u_j des Systems, welche auch gleichzeitig die Optimierungsvariablen sind und dem Index $j = 1, \ldots, m$, wobei m die Anzahl der Eingänge ist, führt auf das Minimierungsproblem

$$\min_{\mathbf{u}} J(\mathbf{u}) \tag{7.6-72}$$

$$\begin{aligned} s.t.\ \mathbf{x}(k+1) &= \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) \\ \mathbf{y}(k) &= \mathbf{C}\mathbf{x}(k) \\ \mathbf{u}_{min} \leq &\mathbf{u}(k) \leq \mathbf{u}_{max} \\ \Delta\mathbf{u}_{min} \leq &\Delta\mathbf{u}(k) \leq \Delta\mathbf{u}_{max} \\ \mathbf{y}_{min} \leq &\mathbf{y}(k) \leq \mathbf{y}_{max}. \end{aligned} \tag{7.6-73}$$

welches unter Randbedingungen gelöst wird [18, 15]. Auf der einen Seite muss die Optimierung dem linearen Zustandsraummodell genügen, wie es in Abschnitt 7.2.1.3 eingeführt wurde, welches das dynamische Verhalten des Systems approximativ wiedergibt. Zusätzlich können Randbedingungen für die Eingangssignale $\mathbf{u}(k)$, die Ausgangssignal $\mathbf{y}(k)$ und die Änderung $\Delta\mathbf{u}(k)$ definiert werden. Es wird angenommen, dass die Randbedingungen genauso wie die Kostenfunktion linear sind. Damit kann das Opimierungsproblem mit einem linearen Optimierer gelöst werden, wie z.B. dem linprog Optimierer der Optimization Toolbox von MATLAB [6].

Für die linprog Funktion wird das Optimierungsproblem (7.6-72) so umgeschrieben, dass es die allgemeine Form

$$\min_{\mathbf{x}} \mathbf{c}^T\mathbf{x} \tag{7.6-74}$$

$$s.t.\ \mathbf{A}\mathbf{x} \leq \mathbf{b} \tag{7.6-75}$$

erfüllt, wobei $\mathbf{x}$ der Vektor der Optimierungsvariablen ist, die Matrix $\mathbf{A}$ und der Vektor $\mathbf{b}$ die linearen Randbedingungen darstellen. Dabei geht die zeitliche Entwicklung des Zustandsraummodells über den Vohersagehorizont H_p als lifted System in die Ungleicheitsbedingungen ein. Auch die weiteren Randbedingungen des Optimierungsproblems (7.6-72) werden in die allgemeine Form der Ungleicheitsbedingung (7.6-75) überführt.

Um die starren Randbedingungen aufzuweichen werden zusätzliche Optimierungsvariablen, sogenannte Slack-Varialben $\mathbf{s}(k)$ eingeführt. Durch diese Variablen bleibt das Optimierungsproblem lösbar auch wenn die Randbedingungen verletzt werden sollten. Diese Zusätzlichen Variablen werden mit hohen Kosten $\boldsymbol{\rho}$ belegt, sodass eine Verletzung der Randbedingungen hoch bestraft wird. Diese zusätzlichen Variablen werden für die Randbedingungen der Ausgänge $\mathbf{y}(k)$ genutzt. Dies führt zu der Kostenfunktion

$$J(\mathbf{u}) = \sum_{i=0}^{H_p-1} \left(c_1u_1(k+i) + c_2u_2(k+i) + \ldots + \rho_1s_1(k+i) + \rho_2s_2(k+i) + \ldots\right) \tag{7.6-76}$$

und dem Optimierungsproblem

$$\min_{\mathbf{u}} J(\mathbf{u}) \tag{7.6-77}$$

$$\begin{aligned} s.t.\ \mathbf{x}(k+1) &= \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) \\ \mathbf{y}(k) &= \mathbf{C}\mathbf{x}(k) \\ \mathbf{u}_{min} &\leq \mathbf{u}(k) \leq \mathbf{u}_{max} \\ \Delta\mathbf{u}_{min} &\leq \Delta\mathbf{u}(k) \leq \Delta\mathbf{u}_{max} \\ \mathbf{y}_{min} &\leq \mathbf{y}(k) + \mathbf{s}(k) \\ \mathbf{y}_{max} &\geq \mathbf{y}(k) - \mathbf{s}(k) \\ \mathbf{s}(k) &\geq 0. \end{aligned} \tag{7.6-78}$$

wobei die Randbedingungen der Ausgänge $\mathbf{y}(k)$ durch die Slack-Variablen angepasst wurden, [18].

Im nächsten Abschnitt wird der EMPC Ansatz so verändert, dass zusätzlich zu den kontinuierlichen Stellsignalen auch diskrete Stellsignale verwendet werden können.

7.6.2 EMPC für schaltende und kontinuierliche Stellsignale

Damit schaltende und kontinuierliche Stellsignale abgebildet werden können, wird anstelle eines linearen Zustandsraummodells ein hybrides Zustandsraummodell genutzt wie es im Abschnitt 7.2.1.4 eingeführt wurde. Die schaltenden Eingänge werden mit $\mathbf{u}_{dis}$ bezeichnet und die dazugehörigen Kosten mit $\mathbf{c}_{dis}$. Um das Hoch- bzw. Runterschalten abzubilden werden für jeden schaltenden Eingang $\mathbf{u}_{dis}$ zwei zusätzliche Variablen $\mathbf{u}_{up}$ und $\mathbf{u}_{down}$ eingeführt und es gilt $\mathbf{u}_{dis}$, $\mathbf{u}_{up}$ und $\mathbf{u}_{down} \in \mathbb{N}_0$. Das Eingangssignal des nächsten Zeitschrittes ergibt sich zu

$$\mathbf{u}_{dis}(k+1) = \mathbf{u}_{dis}(k) + \mathbf{u}_{up}(k) - \mathbf{u}_{down}(k) \tag{7.6-79}$$

Mit Hilfe dieser Zusatzvariablen kann das Hoch- und Runterschalten mit individuellen Kosten $\mathbf{c}_{up}$ und $\mathbf{c}_{down}$ belegt werden. Außerdem kann mit Hilfe der Zusatzvariablen festgelegt werden um wie viele Stufen sich $\mathbf{u}_{dis}$ in einem Zeitschritt ändern darf in dem die Variablen $\mathbf{u}_{up}$ und $\mathbf{u}_{down}$ eingeschränkt werden

$$\begin{aligned}\mathbf{u}_{up,min} \leq \mathbf{u}_{up}(k) \leq \mathbf{u}_{up,max}\\ \mathbf{u}_{down,min} \leq \mathbf{u}_{down}(k) \leq \mathbf{u}_{down,max}.\end{aligned}$$

Damit ergibt sich die neue Kostenfunktion zu

$$\begin{aligned}J(\mathbf{u}) = \sum_{i=0}^{H_p-1} (&c_1 u_1(k+i) + c_2 u_2(k+i) + \ldots + c_{dis,1} u_{dis,1}(k+i) + c_{dis,2} u_{dis,2}(k+i) + \ldots + \\ &c_{up,1} u_{up,1}(k+i) + c_{up,2} u_{up,2}(k+i) + \ldots + c_{down,1} u_{down,1}(k+i) + \\ &c_{down,2} u_{down,2}(k+i) + \ldots + \rho_1 s_1(k+i) + \rho_2 s_2(k+i) + \ldots)\end{aligned} \tag{7.6-80}$$

und das Optimierungsproblem zu

$$\min_{\mathbf{u}} J(\mathbf{u}) \tag{7.6-81}$$

$$\begin{aligned}s.t.\ \mathbf{x}(k+1) &= \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) + \mathbf{B}_{dis}\mathbf{u_{dis}}(k) \\ \mathbf{y}(k) &= \mathbf{C}\mathbf{x}(k) \\ \mathbf{u}_{dis}(k+1) &= \mathbf{u}_{dis}(k) + \mathbf{u}_{up}(k) - \mathbf{u}_{down}(k) \\ \mathbf{u}_{min} &\leq \mathbf{u}(k) \leq \mathbf{u}_{max} \\ \Delta\mathbf{u}_{min} &\leq \Delta\mathbf{u}(k) \leq \Delta\mathbf{u}_{max} \\ \mathbf{u}_{dis,min} &\leq \mathbf{u}_{dis}(k) \leq \mathbf{u}_{dis,max} \\ \mathbf{u}_{up,min} &\leq \mathbf{u}_{up}(k) \leq \mathbf{u}_{up,max} \\ \mathbf{u}_{down,min} &\leq \mathbf{u}_{down}(k) \leq \mathbf{u}_{down,max} \\ \mathbf{y}_{min} &\leq \mathbf{y}(k) + \mathbf{s}(k) \\ \mathbf{y}_{max} &\geq \mathbf{y}(k) - \mathbf{s}(k) \\ \mathbf{s}(k) &\geq 0.\end{aligned} \tag{7.6-82}$$

Durch die Einführung der schaltenden Signale, welche nur ganzzahlige Werte annehmen dürfen, wird das lineare Optimierungsproblem zu einem sogenannten Mixed-Integer Optimierungsproblem. Das bedeutet, dass anstelle eines linearen Optimierers ein Optimierer ausgewählt werden muss, welcher auch Mixed-Integer Optimerungsprobleme lösen kann, wie z.B. der intlinprog Optimierer der Optimization Toolbox von MATLAB [6].

7.6.3 Implementierung eines EMPC an einer Heizungsanlage

In diesem Abschnitt wird der EMPC an einer realen Anlage implementiert und getestet. Dabei wird zunächst der Ansatz an einem Teststand mit kontinuierlichen Stellsignalen implementiert und anschließend auf eine Anlage mit schaltenden und kontinuierlichen Stellsignalen übertragen. Die Implementierung erfolgte in enger Abstimmung mit Kieback&Peter.

7.6.3.1 Implementierung eines EMPC mit kontinuierlichen Stellsignalen

Der erste Schritt ist die Implementierung eines EMPC mit kontinuierlichen Stellsignalen wie er im Abschnitt 7.6.1 vorgestellt wurde. Als Anlage diente der Teststand welcher im Abschnitt 7.4.4 bereits eingeführt wurde und für die Implementierung der datenbasierten lernenden modellprädiktiven Regelung genutzt wurde. Eine schematische Darstellung des Regelkreises mit einem EMPC zeigt Abbildung 7.6-21.

Der EMPC regelt zwei kontinuierliche Stellsignale, das Modulationssignal α der Kesselleistung und das Stellsignal des Vier-Wege-Ventils ϕ. Die Ausgänge des Modells sind die Raumtemperatur T_{raum}, die

Vorlauftemperatur der Kessel $T_{vl,k}$ und die Rücklauftemperatur der Heizung $T_{rl,h}$. Für die Berechnungen der Eingangssignale α und ϕ nutzt der EMPC die Vorhersage der Außentemperatur $T_{a,vor}$.

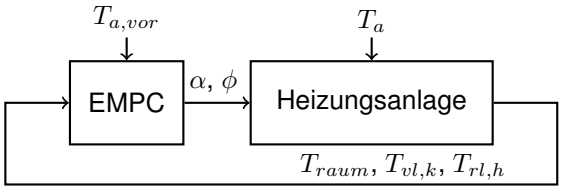


Abbildung 7.6-21: Schematische Darstellung der Heizungsanlage und eines EMPC

Die Implementierung des Reglers erfolgte zunächst auf dem Echtzeitsystem von National Instruments welches in Abschnitt 7.7 vorgestellt wird und schon bei der Implementierung des datenbasierten lernenden modellprädiktiven Reglers im Abschnitt 7.4.4 zum Einsatz kam. Die Kommunikation erfolgte ebenfalls über die BACnet Schnittstelle und die Vorhersage der Außentemperatur wurde über die Wetterstation, wie sie im Abschnitt 7.7 beschrieben wird, bezogen. Der komplette Programmablauf wurde mit der Programmierumgebung LabVIEW realisiert und der lineare Optimierer von LabVIEW verwendet welcher auf dem Simplex-Algorithmus basiert. Nach den ersten Testläufen hat sich herausgestellt, dass mit dem implementierten Workaround kein stabiler Betrieb des Reglers über mehrere Tage möglich war. In LabVIEW gab es wenig Einstellungsmöglichkeiten für den verwendeten Optimierer, sodass an dieser Stelle keine Anpassungen möglich waren. Aufgrund dieser Schwierigkeiten, wurde für die Implementierung des EMPC von dem Echtzeitsystem auf eine Laptop basierte Lösung gewechselt. Die Kommunikation und der Programmablauf erfolgte weiterhin mit LabVIEW und dem BACnet Protokoll, da dieser weg Bereits etabliert war und erfolgreich getestet wurde. Die Messdaten wurden über LabVIEW abgerufen und in einer Datei abgelegt. Anschließend werden die gespeicherten Daten in MATLAB eingelesen und die Optimierung gestartet. Das Optimierungsergebnis wird in einer Datei abgelegt, die Ergebnisse in LabVIEW eingelesen und die Stellsignale mittels BACnet an die DDC der Testanlage geschrieben. Mit diesem geänderten Workaround war ein stabiler Testbetrieb möglich.

Ziel der Implementierung war es die Raumtemperatur in einem definierten Korridor zwischen 22 °C und 26 °C zu halten. Die Optimierung erfolgte mit den Randbedingungen

$$\begin{aligned} 0 \leq &\alpha \leq 1 \\ 0.5 \leq &\phi \leq 1 \\ -0.1 \leq &\Delta\alpha \leq 0.1 \\ -0.1 \leq &\Delta\phi \leq 0.1 \\ 22°C \leq &T_{raum} \leq 26°C. \end{aligned}$$

Es wurden keine Referenzen, wie eine Heizkurve oder ähnliches verwendet. Als fiktive Kosten bzw. Wichtungen wurden folgende Werte festgelegt: $c_\alpha = 2$, $c_\phi = 0.5$ und für die Kosten aller Slack-Variablen $\rho = 1000$. Die Tests wurden mit einer Abtastzeit von $t_s = 2$ min und einem Vorhersagehorizont von $H_p = 60$ durchgeführt. Die Ergebnisse sind in Abbildung 7.6-22 dargestellt.

Die Ergebnisse Zeigen, dass die Raumtemperatur innerhalb der definierten Grenzen gehalten wird. Steigt die Raumtemperatur an, so wird nicht mehr geheizt und das Stellsignal α des Kessels wird auf null gefahren. Am 6.11. steigt die Temperatur über die obere Grenze von 26 °C. Dieser anstieg korrespondiert mit der Außentemperatur. Das heißt, der Raum wird durch die Sonneneinstrahlung aufgeheizt und nicht durch den Kessel, wie ein Blick auf das Stellsignal des Kessels für diesen Zeitraum zeigt. Da es keine Kühlfunktion gibt, kann die Raumtemperatur durch äußere Einflüsse wie die Sonneneinstrahlung auch über die obere Grenze hinaus gehen.

In einem nächsten Schritt wurden für die Ausgangssignale zeitabhängige Randbedingungen eingeführt

$$\begin{aligned} \mathbf{y}_{min}(k) \leq &\mathbf{y}(k) + \mathbf{s}(k) \\ \mathbf{y}_{max}(k) \geq &\mathbf{y}(k) - \mathbf{s}(k), \end{aligned}$$

um zum Beispiel eine Absenkung der Raumtemperatur währen der Zeiten zu realisieren, in denen das

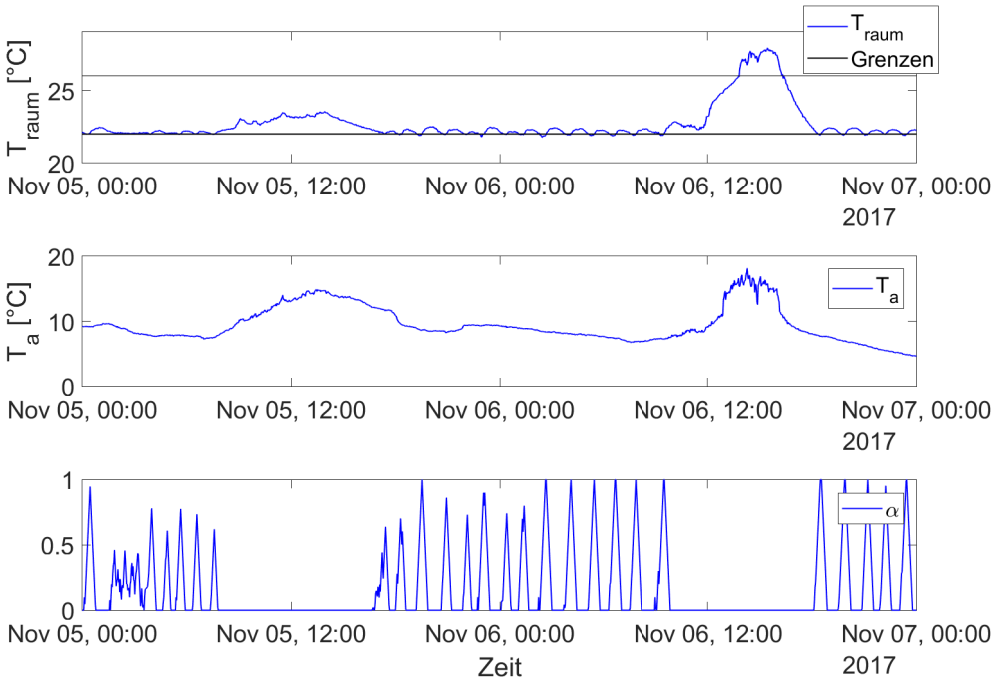


Abbildung 7.6-22: Messergebnisse EMPC

Büro bzw. das Gebäude ungenutzt ist. Das heißt, dass für ein Bürogebäude im allgemeinen währen der Nachtstunden die Temperatur abgesenkt werden kann um Energie einzusparen. Für den folgenden Test wurde das Minimum der Raumtemperatur für die Zeit von 19 bis 6 Uhr um 3 °C auf 19 °C abgesenkt. Die Ergebnisse sind in der Abbildung 7.6-23 dargestellt.

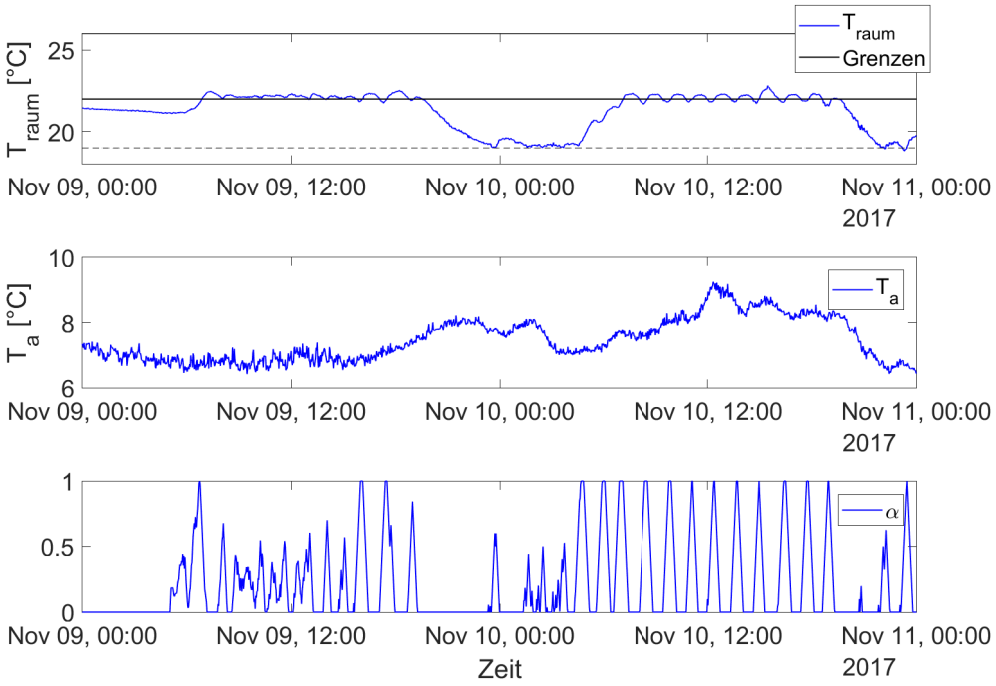


Abbildung 7.6-23: Messergebnisse EMPC mit Nachtabsenkung

Die Ergebnisse in Abbildung 7.6-23 zeigen, dass mit Hilfe von zeitabhängigen Randbedingungen auch eine Nachtabsenkung realisiert werden kann, um außerhalb der Nutzungszeiten Energie einzusparen. Das in der Nacht auf dem 9. November die Raumtemperatur nicht bis auf 19 °C absinkt ist den äußeren Bedingungen geschuldet, da der Raum in diesem Zeitraum nicht beheizt wie ein Blick auf das Stellsignal α verdeutlicht. Auch bei dem Test mit einer Nachtabsenkung der Raumtemperatur wurde nur dann geheizt wenn die Raumtemperatur den definierten Korridor verlässt.

Die ersten Versuche mit der Testanlage zeigte, dass mit dem EMPC die Raumtemperatur in einem definierten Korridor gehalten wird und das ohne eine Heizkurve oder eine andere Referenz festzulegen. Durch die Einführung von zeitabhängigen Randbedingungen kann der Korridor beliebig angepasst werden und somit auch Temperaturabsenkungen realisiert werden, welche bei Bedarf an das Nutzerverhalten angepasst werden können.

7.6.3.2 Implementierung eines EMPC mit kontinuierlichen und diskreten Stellsignalen

Der zweite Schritt war die Übertragung des EMPC Konzeptes auf die Hausanlage des Kieback&Peter Gebäudes in Berlin welches in Abbildung 7.6-24 zu sehen ist. Dabei handelt es sich um ein Bürogebäude mit sieben Etagen. Die Heizungsanlage wird durch zwei Kessel mit der nötigen Wärmeleistung versorgt. Die von den Kesseln bereitgestellten Wärmleistung versorgt auf der Verbraucherseite sieben statische Heizkreise, welche jeweils eine Etage des Gebäudes versorgen, sowie eine RLT Anlage (Raumlufttechnik Anlage). Eine genaue Beschreibung der Anlage erfolgt in dem Kapitel 8 „AP A.5: Implementierung und Evaluation in Gebäudeautomationssystemen“.

Abbildung 7.6-24: Bürogebäude von Kieback&Peter, [16]

Da die Leistung der beiden Kessel nicht kontinuierlich geregelt werden kann, sondern nur in den drei Stufen 0-1-2, kann das Konzept welches an der Testanlage implementiert wurde nicht direkt auf die Heizungsanlage des Gebäudes übertragen werden. Das bedeutet, dass anstatt des EMPC für kontinuierliche Stellsignale wird der EMPC für schaltende und kontinuierliche Stellsignale verwendet, welcher in Abschnitt 7.6.2 eingeführt wurde. Der Workaround, in dem die Kommunikation zwischen dem Laptop und der DDC über LabVIEW realisiert wird und die Optimierung mit MATLAB erfolgt, bleibt erhalten. Auch der Abruf der Vorhersagedaten der Außentemperatur wird genauso Implementiert wie bei dem Test mit dem kontinuierlichen EMPC am Teststand.

Die Modellbildung erfolgte auf Basis der in Abschnitt 7.2.2 eingeführten Differentialgleichungen und analog zu dem in Abschnitt 7.4.4 beschriebenen Vorgehen für die Testanlage. Dabei wird jede Etage zu einer Zone mit einem Raum und einem Heizkörper zusammengefasst. Zusätzlich verfügt jeder

statische Heizkreis über eine Pumpe und ein Drei-Wege-Ventil. Abbildung 7.6-25 zeigt eine schematische Darstellung eines statischen Heizkreises.

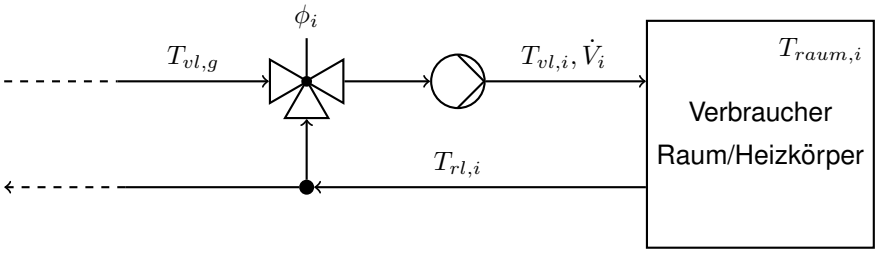


Abbildung 7.6-25: Schematische Darstellung eines Heizkreises

Für die Berechnung der Rücklauftemperatur $T_{rl,i}$ und der Raumtemperaturen $T_{raum,i}$ ($i = 1, \ldots, 7$) eines Heizkreises werden die Differentialgleichungen (7.2-21) und (7.2-22) verwendet. Mittels eines Drei-Wege-Ventil kann die Vorlauftemperatur $T_{vl,g}$, welche die Kesselanlage bereitstellt, für jeden Heizkreis einzeln abgesenkt werden in dem Wasser aus dem Rücklauf des Heizkreises mit der Temperatur $T_{rl,i}$ beigemischt wird. Dies erfolgt nach der Gleichung (7.2-26). Somit gibt es für jeden Heizkreis ein Stellsignal ϕ_i welches die jeweilige Ventilstellung bestimmt. Die Rücklauftemperaturen der einzelnen Heizkreise mischen sich zur Rücklauftemperatur $T_{rl,g}$ des gesamten Systems, welche zurück zu der Kesselanlage fließt. Der Volumenstrom $\dot{V}_i$ welcher die Pumpe für einen Heizkreis erzeugt, geht als zusätzlicher Störgrößeneingang in das Modell ein. Das fünfte Stockwerk wird sowohl über eine Lüftungsanlage mit der Vorlauftemperatur $T_{vl,rlt}$, dem Volumenstrom $\dot{V}_{rlt}$ und der Rücklauftemperatur $T_{rl,rlt}$, als auch über den statischen Heizkreis mit Wärme versorgt. Im sechsten Obergeschoss gibt es keinen Raumtemperatursensor aus diesem Grund wird für diese Temperatur ein Mittelwert aus den übrigen Raumtemperaturen genutzt.

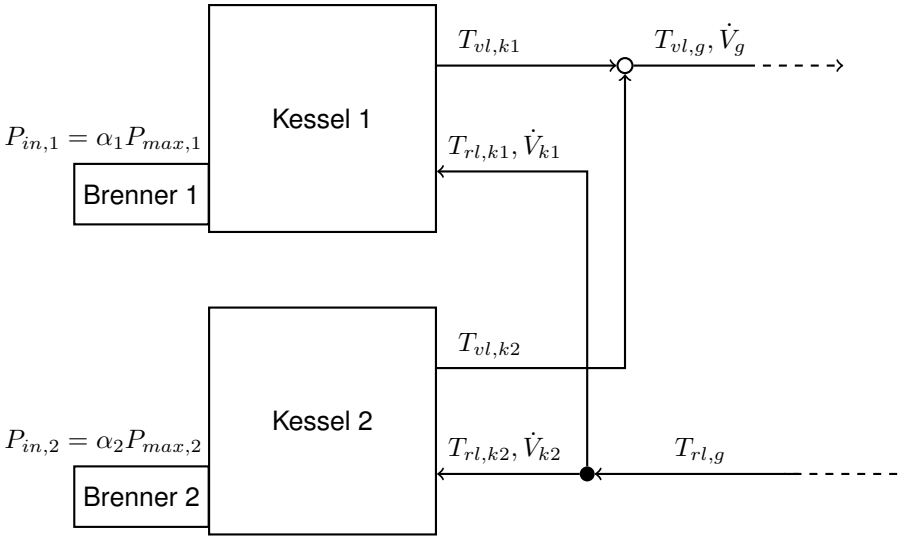


Abbildung 7.6-26: Schematische Darstellung der Erzeugeranlage

Die Kesselanlage besteht aus zwei Kesseln welche Wasser mit den Vorlauftemperaturen $T_{vl,k1}$ und $T_{vl,k2}$ bereitstellen welche sich zum Gesamtvorlauf mit der Temperatur $T_{vl,g}$ mischen. Abbildung 7.6-26 zeigt eine schematische Darstellung der Kesselanlage. Jeder Kessel verfügt über eine eigene Pumpe. Es wird angenommen, dass der Volumenstrom des jeweiligen Kessel $\dot{V}_{k1}$ und $\dot{V}_{k2}$ konstant ist und die Pumpen parallel zu den Kesseln eingeschaltet werden. Damit ergibt sich die Temperatur und der Volumenstrom

des gesamten Vorlaufs zu

$$T_{vl,g} = \frac{T_{vl,k1}\dot{V}_{k1} + T_{vl,k2}\dot{V}_{k2}}{\dot{V}_{k1} + \dot{V}_{k2}} \tag{7.6-83}$$

$$\dot{V}_g = \dot{V}_{k1} + \dot{V}_{k2}. \tag{7.6-84}$$

Die Erzeugerseite und die Verbraucher der Heizungsanlage sind durch eine Hydraulischeweiche voneinander getrennt. Zur Vereinfachung des Modells wird angenommen, dass diese nicht überspült wird.

Damit ergibt sich ein Modell mit 17 Zuständen, 18 Eingängen, und 11 Ausgängen. Von den 18 Eingängen sind 9 Stellsignale und 9 Störgrößen. Tabelle 7.6-2 zeigt eine Übersicht dieser Größen.

Tabelle 7.6-2: Liste der Zustände, Stellsignale, Störgrößen und Ausgänge des Modells

Zustände	Stellsignale	Störgrößen	Ausgänge
$T_{raum,1}(EG)$	α_1	T_a	$T_{raum,1}(EG)$
$T_{raum,2}(1.OG)$	α_2	$\dot{V}_1(EG)$	$T_{raum,2}(1.OG)$
$T_{raum,3}(2.OG)$	$\phi_1(EG)$	$\dot{V}_2(1.OG)$	$T_{raum,3}(2.OG)$
$T_{raum,4}(3.OG)$	$\phi_2(1.OG)$	$\dot{V}_3(2.OG)$	$T_{raum,4}(3.OG)$
$T_{raum,5}(4.OG)$	$\phi_3(2.OG)$	$\dot{V}_4(3.OG)$	$T_{raum,5}(4.OG)$
$T_{raum,6}(5.OG)$	$\phi_4(3.OG)$	$\dot{V}_5(4.OG)$	$T_{raum,6}(5.OG)$
$T_{raum,7}(6.OG)$	$\phi_5(4.OG)$	$\dot{V}_6(5.OG)$	$T_{raum,7}(6.OG)$
$T_{vl,k1}$	$\phi_6(5.OG)$	$\dot{V}_7(6.OG)$	$T_{vl,g}$
$T_{vl,k2}$	$\phi_7(6.OG)$	$\dot{V}_{rlt}$	$T_{rl,g}$
$T_{rl,1}(EG)$			$T_{vl,k1}$
$T_{rl,2}(1.OG)$			$T_{vl,k2}$
$T_{rl,3}(2.OG)$			
$T_{rl,4}(3.OG)$			
$T_{rl,5}(4.OG)$			
$T_{rl,6}(5.OG)$			
$T_{rl,7}(6.OG)$			
$T_{rl,rlt}$			

Die standardmäßig implementierte Regelung wird im folgenden als konventionelle Regelung bezeichnet. Bei der konventionellen Regelung wird jeder der sieben Heizkreise über eine eigene Heizkurve geregelt. Das bedeutet, es gibt insgesamt sieben Heizkurven. Die Vorlauftemperaturanforderungen welche sich aus den Heizkurven der jeweiligen Etage ergeben, werden an die Kesselanlage übermittelt und die Kesselanlage stellt die höchste angeforderte Vorlauftemperatur zur Verfügung. Die Drei-Wege-Ventile regeln, durch Beimischung von kälterem Wasser aus dem Rücklauf, die Vorlauftemperatur $T_{vl,i}$ für den jeweiligen Heizkreis nach der entsprechenden Heizkurve. Da die Kessel der Heizungsanlage nicht kontinuierlich Regelbar sind, schalten diese ein wenn die gemessene Vorlauftemperatur $T_{vl,g}$ unter der von den Heizkreisen angeforderten Temperatur liegt und wieder ab wenn diese überschritten wird. Die Kessel können in die Stufen 0-1-2 schalten. Um permanentes Takten der Kessel zu unterbinden ist eine Hysterese für das Schalten eingestellt.

Das Ziel der Regelung mit dem EMPC ist es die Raumtemperaturen der jeweiligen Heizkreise in einem definierten Korridor zu halten und das bei einer möglichst geringen Kesseltaktung. Dabei werden keine Referenzsignale, wie beispielsweise eine Heizkurve verwendet, sondern das Optimierungsproblem (7.6-81) unter Randbedingung gelöst. Tabelle 7.6-3 zeigt eine Übersicht der verwendeten unteren und oberen Grenzen für die Ausgänge und die Stellsignale, wobei die untere Grenze der Raumtemperaturen währen der Nacht um 2 °C abgesenkt werden. Die unterschiedlichen Wichtungen bzw. Kosten für die Stellsignale oder deren Änderungen, wie sie in die Kostenfunktion des Optimierungsproblems eingehen, wurden von Kieback&Peter zur Verfügung gestellt und werden im Kapitel 8 „AP A.5: Implementierung und Evaluation in Gebäudeautomationssystemen“ genauer erläutert. Die Kosten für den Betrieb eines Brenners mit einer Leistung von 175 kW wurde mit 10.6 Euro pro Stunde beziffert. Da sich die Brenner in der Leistung unterschieden wurde dieser Wert für den jeweiligen Kessel entsprechend angepasst. Die Vorspülverluste bei einer Leistung von 175 kW wurden mit 0.33 Cent berechnet. Bei den verwendeten

Kesseln gibt es keine Nachspülfunktion. Das heißt, dass das Hochschalten eines Brenners mit Kosten belegt wird, das Runterschalten aber nicht. Das Stellen eines Ventils führt zu geringen elektrischen Verlusten von $6.25 \cdot 10^{-5}$ Cent pro Stellschritt. Diese ermittelten Kosten gehen direkt in die Kostenfunktion des Optimierungsproblems ein.

Für die Implementierung wurden von Kieback&Peter in Abstimmung mit der HAW Sicherheitsmechanismen entwickelt, um ein Umschalten auf den konventionellen Betrieb zu gewährleisten, wenn bei dem Testlauf des EMPC etwas nicht funktionieren sollte. Damit wurde sichergestellt, dass das Gebäude nicht auskühlt und ein Bürobetrieb im Gebäude nicht gestört wird. So wurde beispielsweise ein Keep-Alive Signal welches der EMPC sendet überprüft um im Falle eines Ausfalls automatisch auf die konventionelle Regelung umzuschalten. Eine ausführliche Beschreibung dieser Mechanismen erfolgt im Kapitel 8 „AP A.5: Implementierung und Evaluation in Gebäudeautomationssystemen“.

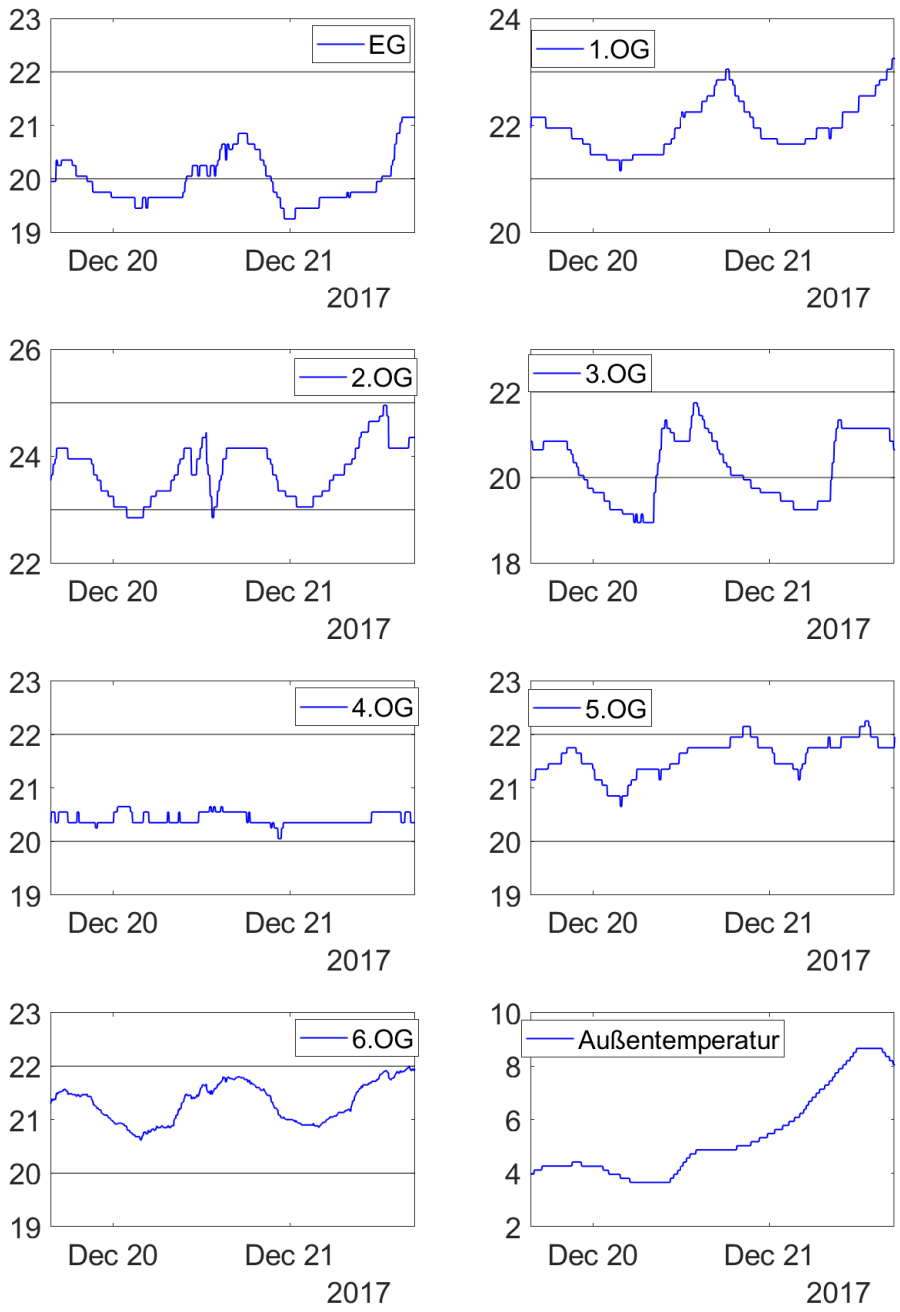


Abbildung 7.6-27: Ergebnisse der gemessenen Raumtemperaturen und der Außentemperatur in °C

Die Optimierung wurde mit einer Abtastzeit von $t_s = 120$ s und einem Vorhersagehorizont von $H_p = 60$ ausgeführt. Abbildung 7.6-27 und 7.6-29 zeigen die Ergebnisse eines zweitägigen Testlaufs mit dem EMPC.

Tabelle 7.6-3: Randbedingungen für die Ausgänge und Stellsignale

Ausgänge	Unt. Grenze [°C]	Ob. Grenze [°C]	Stellsignale	Unt. Grenze	Ob. Grenze
$T_{raum,1}(EG)$	20	22	α_1	0	2
$T_{raum,2}(1.OG)$	21	23	α_2	0	2
$T_{raum,3}(2.OG)$	23	25	$\phi_1(EG)$	0	1
$T_{raum,4}(3.OG)$	20	22	$\phi_2(1.OG)$	0	1
$T_{raum,5}(4.OG)$	20	22	$\phi_3(2.OG)$	0	1
$T_{raum,6}(5.OG)$	20	22	$\phi_4(3.OG)$	0	1
$T_{raum,7}(6.OG)$	20	22	$\phi_5(4.OG)$	0	1
$T_{vl,g}$	40	80	$\phi_6(5.OG)$	0	1
$T_{rl,g}$	30	80	$\phi_7(6.OG)$	0	1
$T_{vl,k1}$	50	80			
$T_{vl,k2}$	50	80			

Auch wenn eine Auswertung der Messergebnisse von zwei Tagen noch keine Aussagen über das Langzeitverhalten zulässt, so kann anhand der ersten Messergebnisse gezeigt werden, dass die Raumtemperaturen tagsüber in den vorgegebenen Korridoren für jeden Etage gehalten werden, (Abbildung 7.6-27). Das unterschreiten dieser Grenzen während der Nacht ist erlaubt und spiegelt die Nachtabsenkung wieder, welche zu einem reduzierten Energieverbrauch während der Nacht führt, da die Heizkreise mit geringeren Vorlauftemperaturen versorgt werden. Einen deutlichen Unterschied zur konventionellen Regelungsstrategie spiegelt sich in dem Vergleich der Vorlauftemperaturen während der Nachtabsenkung wieder. So wurden die Vorlauftemperaturen beider Kessel mit der EMPC Regelungsstrategie auf dem eingestellten Minimum von 50 °C gehalten, im Vergleich dazu wurde die Vorlauftemperatur mit der konventioneller Regelung nicht so stark abgesenkt, (Abbildung 7.6-28 und 7.6-29).

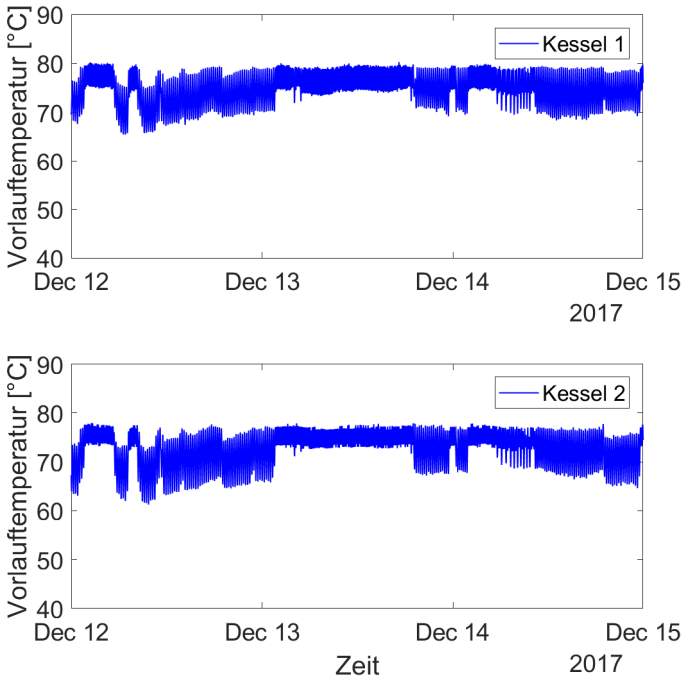


Abbildung 7.6-28: Vorlauftemperaturen der Kessel, konventionell

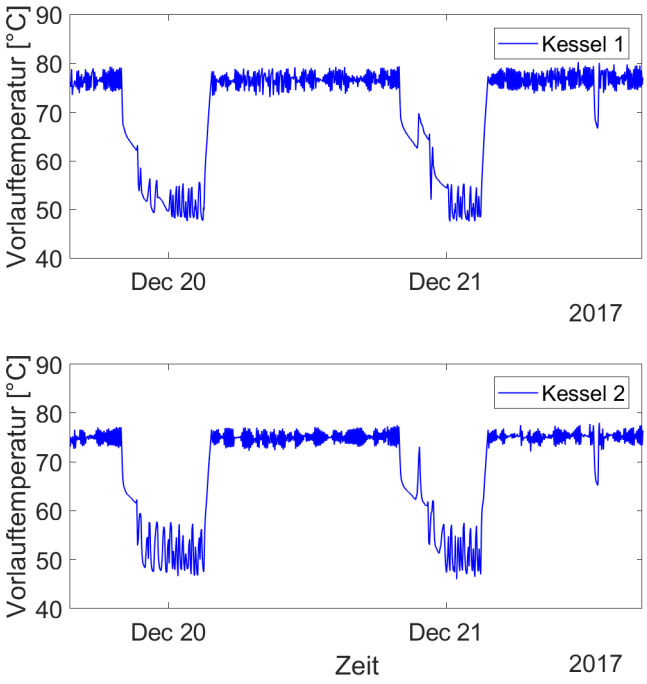


Abbildung 7.6-29: Vorlauftemperaturen der Kessel, EMPC

Das Schaltverhalten der Kessel konnte reduziert werden. Dazu wurde ausgewertet wie oft der jeweilige Kessel im Durchschnitt pro Tag angeschaltet wird. Für die konventionelle Regelung wurden 7 Tage im November ausgewertet und mit den Taktungen der zwei Tage verglichen in denen der EMPC lief. Die sieben Tage im November wiesen einen ähnlichen Außentemperaturbereich auf wie die beiden Testtage im Dezember. Die Ergebnisse sind in der Tabelle 7.6-4 zusammengefasst.

Tabelle 7.6-4: Vergleich der Kesseltaktung: Konventionell - EMPC

	Konventionell	EMPC
Kessel 1	40.42	14.9
Kessel 2	32.28	16.36

Für den ersten Kessel konnte die Taktung um ≈ 63 % während der zwei Testtage reduziert werden und für den zweiten Kessel um ≈ 49 %. Es hat sich während der Testphase gezeigt, dass es für die Anwendung des EMPC wichtig ist den Korridor in dem die Raumtemperatur gehalten werden soll an die jeweiligen Referenzräume anzupassen. Je nach Wahl des Referenzraums kann die mittlere Temperatur unterschiedliche ausfallen. Wenn Büroräume als Referenzräume genutzt werden fallen diese Unterschiede deutlicher ins Gewicht, da jeder Nutzer eine andere Komforttemperatur einstellt und somit der Korridor in dem die Raumtemperatur gehalten werden soll an diese Temperatur angepasst werden muss. Wenn der Korridor höher gewählt wird als die eingestellte Raumtemperatur im Referenzraum wird die Heizungsanlage dauerhaft Heizen und somit eine Überversorgung auftreten. Im Gegensatz dazu würde ein zu niedrig eingestellter Korridor eine Unterversorgung des Gebäudes nach sich ziehen. Dadurch kommt der Wahl der Referenzräume eine entscheidende Bedeutung zu, da die Regelungsstrategie direkt auf das Einhalten von vorher festgelegten Komfortgrenzen der Raumtemperaturen abzielt. Weiterführende Überlegungen um den Einfluss einer Referenztemperatur für die Anwendung abzuschwächen sind, anstelle eines Referenzraums pro Etage, zwei oder mehr Räume zu nutzen. Eine weitere Überlegung ist auf Temperaturänderungen mit zeitlicher Verzögerung zu reagieren um kurzfristige Raumtemperaturänderungen abzufangen wie sie beispielsweise durch die Öffnung eines Fensters auftreten können. Eine Untersuchung des Langzeitverhaltens des EMPC ist unumgänglich um eine bessere Vergleichbarkeit zwischen den beiden Ansätzen zu erzielen.

7.7 Verwendete Hardware für die Implementierung

Für die Implementierung, sowie Vorabtests wurde unterschiedliche Hard- und Software verwendet, welche im folgenden kurz vorgestellt wird.

Als Hardware Plattform wurden zwei Echtzeitsysteme von National Instruments, sowie die entsprechenden I/O-Karten angeschafft. Bei den Echtzeitsystemen handelt es sich um das CompactRIO-9030, sowie das CompactRIO-9035. Der CompactRIO Controller-9030 hat einen Dual-Core-CPU (1,33 GHz) Prozessor, einen Arbeitsspeicher von 1 GB DRAM , 4 GB lokalen Speicher, einen Kintex-7-70T-FPGA und 4 Slots für die unterschiedlichen I/O-Karten. Der CompactRIO Controller-9035 hat die selben Spezifikationen wie der CompactRIO Controller-9030 nur anstelle von 4 Slots, 8 Slots für I/O-Karten. Die angeschafften I/O-Karten sind analoge, so wie digitale Ein- und Ausgänge. Die I/O-Karten können beliebig zwischen den Geräten gewechselt werden, sodass ein hohes Maß an Flexibilität gewährleistet ist und das System für unterschiedlichste Anwendungsfälle einsetzbar wird. Das Echtzeitsystem wird mit Hilfe der grafischen Programmierumgebung labVIEW programmiert, [2]. Über den Displayport kann ein Bildschirm an das Echtzeitsystem angeschlossen werden, sodass eine visuelle Überwachung während des Betriebes stattfinden kann. Auch ist der Anschluss einer Maus oder Tastatur möglich, so können definierte Parameter auch während des Betriebes angepasst werden. Abbildung 7.7-30 zeigt das Echtzeitsystem compactRIO-9035.

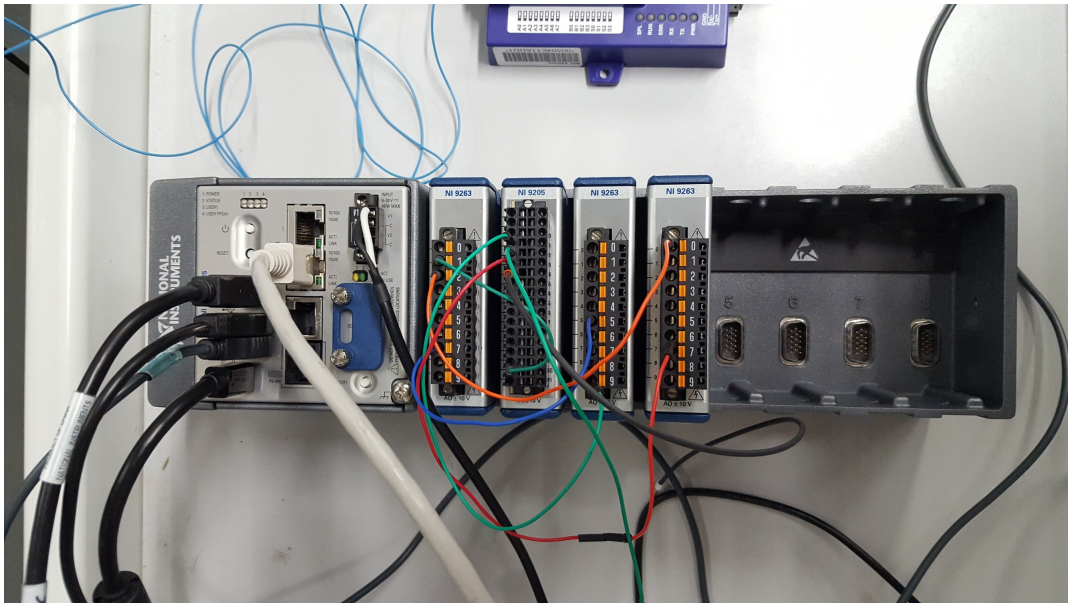
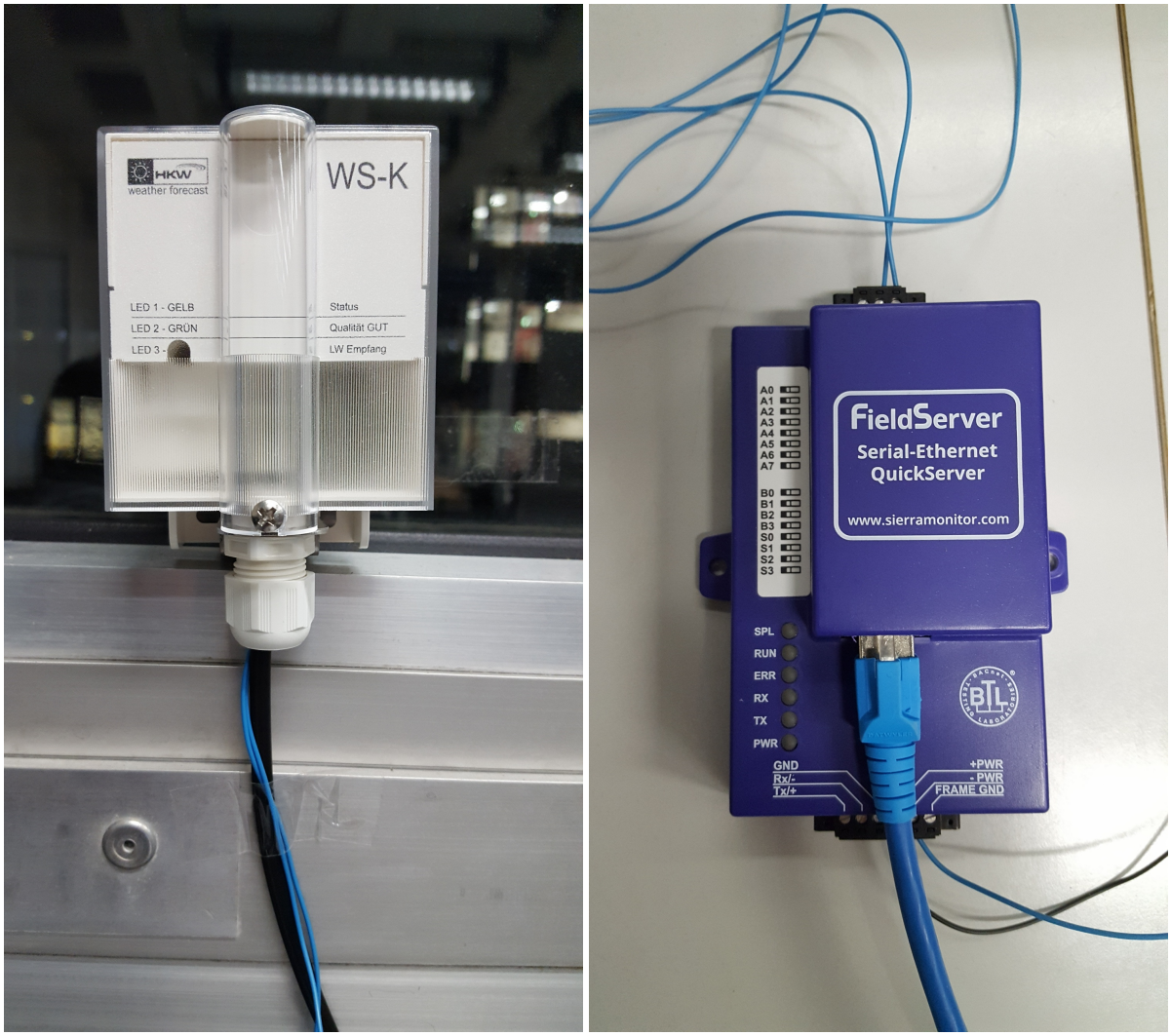


Abbildung 7.7-30: Echtzeitsystems cRIO-9035

Für Regelungskonzepte welche die Wettervorhersage berücksichtigen braucht es eine Möglichkeit für die Echtzeitanwendung auf die Prognosedaten zuzugreifen. Die Wetterstation WS-K ModBus RTU485 der Firma HKW Elektronik GmbH biete die Möglichkeit über einen Langwellenempfänger die Wetterprognosedaten zu Empfangen und über das ModBus Protokoll mit anderen Geräten zu kommunizieren. Mit dieser Wetterstation können beispielsweise die Stundenwerte der Außentemperaturvorhersage für drei Tage Empfangen werden, aber auch Solartrahlungs- und Windprognosen im sechs Stundenraster. Das Gerät biete den besonderen Vorteil, dass lediglich einmal der Anschaffungspreis bezahlt werden muss und anschließend keine weiteren Kosten für die Nutzung der Wetterprognosedaten anfallen. Mit Hilfe des FieldServers EZ Gateway ModBus zu BACnet der Firma Sierra Monitor Corporation wurden das ModBus Protokoll welches die Wetterstation zur Kommunikation nutzt in das BACnet (BACnet - Building Automation and Control Network) Protokoll umgewandelt, sodass die Prognosedaten über das BACnet Protokoll ausgelesen werden konnten. Die Anschaffung dieser Gerätekombination machte an dieser Stelle Sinn, da bei Kieback&Peter bereits Erfahrung mit diesen Geräten vorhanden war und somit von dieser profitiert werden konnte. Die Abbildungen 7.7-31 (a) und (b) zeigen die Wetterstation sowie den Field-Server.



(a) Wetterstation (b) Field-Server

Abbildung 7.7-31: Wetterstation und Field-Server

Die Kommunikation der einzelnen Komponenten erfolgte über das BACnet Protokoll, welches in der Gebäudeautomation ein gängiger Standard für die Netzwerkkommunikation. Mit Hilfe des Softwarepaketes BACnet_IP Protocol der Firma OVAK Technologies für labVIEW konnte die BACnet Kommunikation, sowohl über das Echtzeitsystem von National Instruments als auch über einen Computer realisiert werden. Die Programmierung erfolgt, sowohl für das Echtzeitsystem als auch auf dem Computer über die Programmierumgebung labVIEW.

Die Systeme wurden im Labor aufgebaut um erste Hardware in the Loop Tests durchführen zu können, siehe Abbildung 7.7-32. Aber auch vor Ort bei der realen Heizungsanlage um die Algorithmen live auszuprobieren und die Heizungsanlage zu steuern. Durch die Anschaffung dieser Geräte ergaben sich vielseitige Nutzungsmöglichkeiten, wodurch die Implementierung der entwickelten Algorithmen bei einer Heizungsanlage erst realisiert werden konnten.

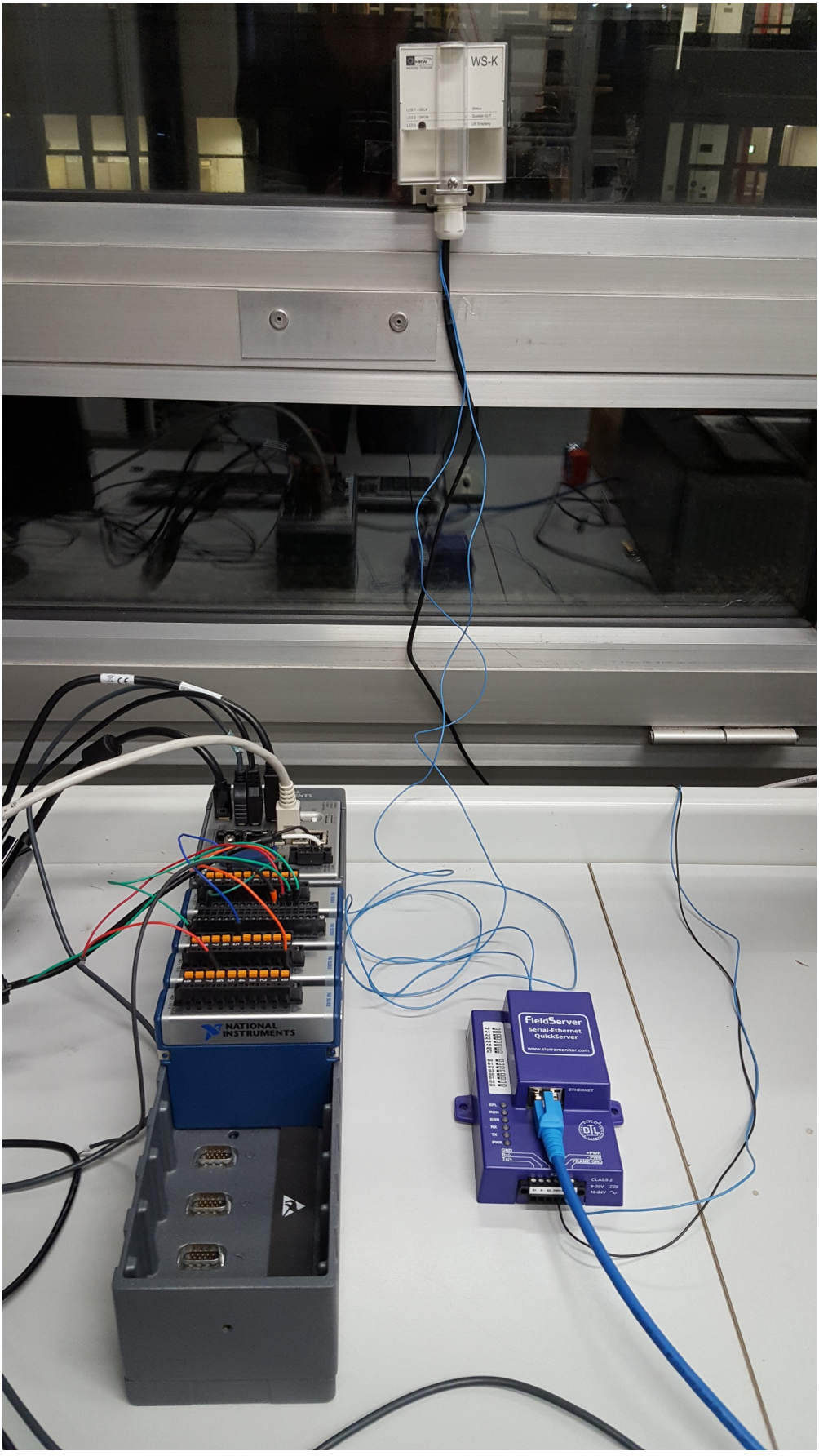


Abbildung 7.7-32: Laboraufbau der Hardwarekomponenten

7.8 Zusammenfassung und Ausblick

In diesem Kapitel, „AP B.4 - Lernende prädiktive Regelung, Methoden und Design“ wurden unterschiedliche Ansätze der modellprädiktiven Regelung diskutiert und der Ansatz eines datenbasierten iterativ lernenden Reglers eingeführt und gezeigt wie dieser für die modellprädiktive Regelung eingesetzt werden kann. Die untersuchten und entwickelten Algorithmen wurden auf Heizungssysteme angewandt und in Simulation, aber auch an realen Heizungssystemen getestet.

Es wurde die Struktur des Optimierungsproblems der modellprädiktiven Regelung für multilineare Systeme auf Konvexität untersucht. Das Optimierungsproblem ist durch eine quadratische Kostenfunktion gegeben welche ein input lineares MTI System mit einem Eingang verwendet. Durch prüfen, ob die Hessematrix der Kostenfunktion positiv-semidefinit ist und einer konvexen Menge der Optimierungsvariablen konnte die Konvexität des Optimierungsproblems bewiesen werden. Dabei hat sich gezeigt, dass für einen Vorhersagehorizont von eins die gesamte Klasse der MTI Systeme mit einem Eingang konvex ist. Das gilt nicht mehr für einen Vorhersagehorizont von zwei. Es konnte gezeigt werden, dass eine spezielle Unterklasse der input linearen MTI Systeme für einem Vorhersagehorizont von zwei konvex ist. Anhand eines Beispiels wurde gezeigt wie das Ergebnis der Optimierung mit einen Vorhersagehorizont von zwei genutzt werden kann um geeignete Startwerte für das Optimierungsproblem mit einem größeren Vorhersagehorizont zu finden. Die Frage, ob es noch weitere Unterklassen der MTI-Systeme gibt für die das modellprädiktive Optimierungsproblem konvex ist, ist von zentraler Bedeutung. Auch mit Hinblick auf die Anwendbarkeit auf unterschiedlichste Systeme, beispielsweise mit mehr Eingängen, öffnet sich ein weites Feld für weitere interessante Untersuchungen der Klasse der MTI Systeme.

Die iterativ lernenden prädiktive Regelung dagegen nutzt ein linearisiertes Modell für den MPC und kombiniert die Vorzüge eines MPC mit denen eines lernenden Reglers. Dabei Arbeitet der datenbasierte lernende Regler mit den gespeicherten Daten, welche in der heutigen Zeit oft für Monitoringzwecke gespeichert, aber nicht für die Regelung genutzt werden. Es wurde ein Auswahlkriterium eingeführt welches festlegt, welche Iteration der Vergangenheit für die Berechnung des Eingangssignals der nächsten Iteration genutzt wird. Dazu wurden die Eingangssignale, die Abweichung, sowie die Umgebungsbedingungen jeder vergangenen Iteration gespeichert. Es wurde gezeigt wie der datenbasierte ILC mit einem prädiktiven Regler kombiniert werden kann. Die Simulationsergebnisse eines Heizungssystems haben gezeigt wie der datenbasierte ILC durch die Anpassung der Referenz des MPC die Raumtemperatur senken konnte ohne die definierten Komfortbedingungen zu verletzen, im Vergleich zu den Ergebnissen mit einem MPC alleine. Mit dem Vorteil, dass eine Reduktion der Raumtemperatur direkt zu einer Einsparung von Energie führt. Auch die Implementierung an eine reale Testanlage eines Heizungssystems erfolgte im Rahmen dieser Arbeit. Die Ergebnisse weniger Tage zeigten hier erste vielversprechende erfolge und würden sich für weiterführende Untersuchungen anbieten. Auch die Fragen, ab wann ein historischer Datensatz gelöscht werden kann, da er sehr weit in der Vergangenheit liegt, könnte Untersucht werden. Auch im Hinblick auf ähnliche Tage die gespeichert sind stellt sich die Frage, welcher Tag im Speicher gehalten werden sollte und welcher gelöscht werden könnte.

Für die Anwendbarkeit des datenbasierten iterativ lernenden Reglers auf Zielplattformen mit keinem Datenbankzugriff oder geringem lokalen Speicher wurde in diesem Kapitel eingegangen. Dazu wurde das CP Zerlegungsverfahren auf die Außentemperatur angewandt und die Berechnungen des Ähnlichkeitskriteriums für den ILC auf Basis der Faktormatrizen durchgeführt, mit dem Ziel den Speicherbedarf zu reduzieren. Bei einer Rang fünf CP Zerlegung von Daten eines Jahres, konnte eine Reduktion des Speicherbedarfs um einen Faktor von ≈ 70 erreicht werden. Am Beispiel eines Heizungssystems, zeigten die Simulationsergebnisse einer rang fünf CP Approximation der Außentemperatur eine Übereinstimmung der Ergebnisse von 83 %, im Vergleich zu den Simulationsergebnissen mit dem Messwerttensor. Die Frage, ob andere Reduktionsverfahren, wie Tucker oder Tensor Trains, angewandt werden können und weiteres Potential bei der Speicherbedarfsreduktion besitzen, ist für weitere Untersuchungen spannend.

Die Anwendung eines EMPC auf die Testumgebung eines realen Heizungssystems und die anschließende Übertragung auf die Gebäudeanlage haben gezeigt, dass es mit Hilfe eines EMPC möglich ist die Raumtemperatur in einem definierten Korridor zu halten, wobei die geforderte Wärmeleistung aufgrund der Randbedingungen an die Raumtemperatur vom Kessel zur Verfügung gestellt wurde und keine Referenzen wie eine Heizkurve zum Einsatz kam. Durch Zeitabhängige Randbedingungen wurde eine Nachtabsenkung für die Zeiten in denen das Gebäude nicht genutzt wird realisiert. Bei der Realisierung eines EMPC mit schaltenden Stellsignale konnte für den Testzeitraum das Takten der Kessel um ≈ 63 %

bzw. ≈ 49 % verringert werden. Auf Grund des relativ kurzen Testzeitraumes von zwei Tagen ist ein Langzeittest in der Zukunft unumgänglich um eine bessere Vergleichbarkeit herstellen zu können. Die ersten Erfahrungen haben gezeigt, dass den Referenzräumen eine besondere Bedeutung zukommt. Dabei ist es wichtig die gewählten Grenzen an den jeweiligen Referenzraum anzupassen um eine Über- bzw. Unterversorgung zu verhindern. Weiterführende Überlegungen sind hier mehrere Referenzräume pro Etage zu nutzen um den Einfluss einzelner Räume zu minimieren. Auch plötzliche Temperaturänderungen, beispielsweise durch das öffnen eines Fensters, könnten durch eine zeitverzögerte Reaktion der Regelung abgefangen werden und könnten in weiterführenden Projekten untersucht werden.

Literaturverzeichnis

[1] *Forschung für eine umweltbezogene, zuverlässige und bezahlbare Energieversorgung, Das 6. Energieforschungsprogramm der Bundesregierung*. Bunderministerium für Wirtschaft u. Technologie. www.bmwi.de. Version: Juli 2011. – Download am 27.04.2015

[2] *LabVIEW, version 2015 SP1*. Austin, Texas : National Instruments, 2015

[3] *MATLAB, version R2015b*. Natick, Massachusetts : The MathWorks Inc., 2015

[4] *Model Predictive Control Toolbox, version R2015a*. Natick, Massachusetts : The MathWorks Inc., 2015

[5] *Simulink, version R2015a*. Natick, Massachusetts : The MathWorks Inc., 2015

[6] *Optimization Toolbox, version R2017b*. The MathWorks Inc., 2017

[7] AHN, Hyo-Sung ; CHEN, YangQuan ; MOORE, Kevin L.: Iterative Learning Control: Brief Survey and Categorization. In: *IEEE Transaction on Systems, Man, and Cybernetics - Part C: Application and Reviews* 37 (2007), Nr. 6

[8] BADER, B. ; KOLDA, T.: *MATLAB Tensor Toolbox Version 2.5*. Available online. http://www.sandia.gov/~tgkolda/TensorToolbox/. Version: 2012

[9] BOYD, Stephen ; VANDENBERGHE, Lieven: *Convex Optimization*. New York, NY, USA : Cambridge University Press, 2009. – ISBN 0521833787

[10] BRISTOW, D. A. ; THARAYIL, M. ; ALLEYNE, A. G.: A survey of iterative learning control. In: *IEEE Control Systems* 26 (2006), Juni, Nr. 3, S. 96–114. http://dx.doi.org/10.1109/MCS.2006.1636313. – DOI 10.1109/MCS.2006.1636313. – ISSN 1066–033X

[11] CICHOCKI, A. ; ZDUNEK, R. ; PHAN, A. ; AMARI, S.: *Nonnegative matrix and tensor factorizations*. Wiley, Chichester, 2009

[12] COLE, Wesley J. ; MORTON, David P. ; EDGAR, Thomas F.: Optimal electricity rate structures for peak demand reduction using economic model predictive control. In: *Journal of Process Control* 24 (2014), Nr. 8, S. 1311 – 1317. http://dx.doi.org/https://doi.org/10.1016/j.jprocont.2014.04.014. – DOI https://doi.org/10.1016/j.jprocont.2014.04.014. – Economic nonlinear model predictive control

[13] COSTANZO, Giuseppe T. ; IACOVELLA, Sandro ; RUELENS, Frederik ; LEURS, T. ; CLAESSENS, Bert: Experimental analysis of data-driven control for a building heating system. In: *CoRR* abs/1507.03638 (2015)

[14] DIN-EN-15251:2007 ; V., DIN Deutsches I. e. (Hrsg.): *Indoor environmental input parameters for design and assessment of energy performance of buildings addressing indoor air quality, thermal environment, lighting and acoustics; German version EN 15251:2007*. Beuth Verlag GmbH, 2007

[15] ELLIS, Matthew ; DURAND, Helen ; CHRISTOFIDES, Panagiotis D.: A tutorial review of economic model predictive control methods. In: *Journal of Process Control* 24 (2014), Nr. 8, S. 1156 – 1178. http://dx.doi.org/https://doi.org/10.1016/j.jprocont.2014.03.010. – DOI https://doi.org/10.1016/j.jprocont.2014.03.010

[16] FRAUNHOFER ISE ; HAW HAMBURG ; INGSOFT GMBH ; PLENUM INGENIEURGESELLSCHAFT ; KIEBACK & PETER GMBH & CO. KG: *Observe Webseite*. – http://www.ob-serve.de/, zuletzt abgerufen am 04.10.2016

[17] GRASEDYK, L. ; KRESSNER, L. ; TOBLER, C.: A literature survey of low-rank tensor approximation techniques. In: *GAMM-Mitteilungen* 36 (2013), S. 53–78

[18] HALVGAARD, R. ; POULSEN, N. K. ; MADSEN, H. ; JØRGENSEN, J. B.: Economic Model Predictive Control for building climate control in a Smart Grid. In: *Proc. IEEE PES Innovative Smart Grid*

Technologies (ISGT), 2012, S. 1–6

[19] HENZE, Gregor P.: Model predictive control for buildings: a quantum leap? In: *Journal of Building Performance Simulation* 6 (2013), Nr. 3, S. 157–158

[20] KELMAN, Anthony ; BORRELLI, Francesco: Bilinear Model Predictive Control of a HVAC System Using Sequential Quadratic Programming. In: *18th IFAC World Congress*, 2011, S. 9869–9874

[21] KOLDA, T. ; BADER, B.: Tensor Decompositions and Applications. In: *SIAM Review* 51 (2009), Nr. 3, S. 455–500

[22] KRUPPA, K.: *Vergleich verschiedener Modellierungswerkzeuge für Wärmeversorgungssysteme*. Institut für Regelungstechnik, TU Hamburg-Harburg, Bachelor Thesis, 2010

[23] KRUPPA, K.: Dokumentation Matlab HeatLib - Zur Vewendung mit Matlab / Institut für Regelungstechnik. TU Hamburg-Harburg, 2012. – Dokumentation

[24] KRUPPA, K. ; PANGALOS, G. ; LICHTENBERG, G.: Multilinear Approximation of Nonlinear State Space Models. In: *19th World Congress* IFAC, 2014, S. 9474–9479

[25] KRUPPA, Kai ; LICHTENBERG, Gerwald: Decentralized State Feedback Design for Multilinear Time-Invariant Systems. In: *IFAC-PapersOnLine* 50 (2017), Nr. 1, 5616 - 5621. http://dx.doi.org/https://doi.org/10.1016/j.ifacol.2017.08.1108. – DOI https://doi.org/10.1016/j.ifacol.2017.08.1108. – ISSN 2405–8963. – 20th IFAC World Congress

[26] LAUTENSCHLAGER, Björn ; KRUPPA, Kai ; LICHTENBERG, Gerwald: Convexity Properties of the Model Predictive Control Problem for Subclasses of Multilinear Time-Invariant Systems. In: *IFAC-PapersOnLine* 48 (2015), Nr. 23, 148 - 153. http://dx.doi.org/https://doi.org/10.1016/j.ifacol.2015.11.275. – DOI https://doi.org/10.1016/j.ifacol.2015.11.275. – ISSN 2405–8963. – 5th IFAC Conference on Nonlinear Model Predictive Control NMPC 2015

[27] LAUTENSCHLAGER, Björn ; LICHTENBERG, Gerwald: Data-driven Iterative Learning for Model Predictive Control of Heating Systems. In: *IFAC-PapersOnLine* 49 (2016), Nr. 13, S. 175 – 180. http://dx.doi.org/http://dx.doi.org/10.1016/j.ifacol.2016.07.947. – DOI http://dx.doi.org/10.1016/j.ifacol.2016.07.947. – ISSN 2405–8963. – 12th {IFAC} Workshop on Adaptation and Learning in Control and Signal Processing {ALCOSP} 2016Eindhoven, The Netherlands, 29 June - 1 July 2016

[28] LICHTENBERG, G.: *Hybrid Tensor Systems*, Hamburg University of Technology, Diss., 2011

[29] MA, Jingran ; QIN, Joe ; SALSBURY, Timothy ; XU, Peng: Demand reduction in building energy systems based on economic model predictive control. In: *Chemical Engineering Science* 67 (2012), Nr. 1, S. 92–100

[30] MACIEJOWSKI, J.: *Predictive Control with Constraints*. Pearson Education Limited 2002, 2002. – ISBN 0 201 39823 0

[31] MINAKAIS, M. ; MISHRA, S. ; WEN, J.T.: Groundhog Day: Iterative learning for building temperature control. In: *Automation Science and Engineering (CASE), 2014 IEEE International Conference on*, 2014, S. 948–953

[32] MÜLLER, Thorsten ; KRUPPA, Kai ; LICHTENBERG, Gerwald ; RÉHAULT, Nicolas: Fault Detection with Qualitative Models reduced by Tensor Decomposition methods. In: *IFAC-PapersOnLine* 48 (2015), Nr. 21, 416 - 421. http://dx.doi.org/https://doi.org/10.1016/j.ifacol.2015.09.562. – DOI https://doi.org/10.1016/j.ifacol.2015.09.562. – ISSN 2405–8963. – 9th IFAC Symposium on Fault Detection, Supervision andSafety for Technical Processes SAFEPROCESS 2015

[33] MÜLLER, Thorsten ; LICHTENBERG, Gerwald: Fault Detection with CP-Decomposed Qualitative Models. In: *IFAC-PapersOnLine* 49 (2016), Nr. 5, 309 - 314. http://dx.doi.org/https://doi.org/10.1016/j.ifacol.2016.07.131. – DOI https://doi.org/10.1016/j.ifacol.2016.07.131. – ISSN 2405–8963. – 4th IFAC Conference on Intelligent Control and Automation SciencesICONS 2016

[34] MÜLLER-EPING, Thorsten ; LICHTENBERG, Gerwald ; VOGELMANN, Vivien: Fault Detection Algorithms based on Decomposed Tensor Representations for Qualitative Models. In: *IFAC-PapersOnLine* 50 (2017), Nr. 1, 5622 - 5629. http://dx.doi.org/https://doi.org/10.1016/j.ifacol.2017.08.1109. – DOI https://doi.org/10.1016/j.ifacol.2017.08.1109. – ISSN 2405–8963. –

20th IFAC World Congress

[35] MORARI, Manfred ; LEE, Jay H.: Model predictive control: past, present and future. In: *Computers & Chemical Engineering* 23 (1999), Nr. 4-5, S. 667 – 682. – ISSN 0098–1354

[36] OLDEWURTEL, Frauke ; PARISIO, Alessandra ; JONES, Colin N. ; GYALISTRAS, Dimitrios ; GWERDER, Markus ; STAUCH, Vanessa ; LEHMANN, Beat ; MORARI, Manfred: Use of model predictive control and weather forecasts for energy efficient building climate control. In: *Energy and Buildings* 45 (2012), S. 15 – 27. – ISSN 0378–7788

[37] PANGALOS, G. ; EICHLER, A. ; LICHTENBERG, G.: Tensor Systems: Multilinear Modeling and Applications. In: *3rd Int. Conference on Simulation and Modeling Methodologies, Technologies and Applications*, 2013. – submitted

[38] PFEIFFER, S. ; LICHTENBERG, G.: Iterative Learning Control exploiting SO(2) Symmetry for a Free-Electron Laser. In: *IFAC International Workshop on Adaptation and Learning in Control and Signal Processing*, 2013

[39] PFEIFFER, S. ; LICHTENBERG, G. ; SCHMIDT, C. ; SCHLARB, H.: Tensor techniques for iterative learning control of a free-electron laser. In: *2012 IEEE International Conference on Control Applications*, 2012. – ISSN 1085–1992, S. 160–165

[40] PRIVARA, Samuel ; SIROKY, Jan ; FERKL, Lukas ; CIGLER, Jiri: Model predictive control of a building heating system: The first experience. In: *Energy and Buildings* 43 (2011), S. 564 – 572. – ISSN 0378–7788

[41] SEWE, Erik ; PANGALOS, Georg ; LICHTENBERG, Gerwald: Fault Detection for Heating Systems using Tensor Decompositions of Multi-Linear Models. In: *7th International Conference on Simulation and Modeling Methodologies, Technologies and Applications* (2017)

[42] VERVLIET, N. ; DEBALS, O. ; SORBER, L. ; VAN BAREL, M. ; DE LATHAUWER, L.: *Tensorlab 3.0*. `http://www.tensorlab.net`. Version: Mar. 2016. – Available online.

[43] WANG, Youqing ; GAO, Furong ; III, Francis J. D.: Survey on iterative learning control, repetitive control, and run-to-run control. In: *Journal of Process Control* 19 (2009), Nr. 10, S. 1589 – 1600. – ISSN 0959–1524

[44] YAN, Xiuying ; REN, Qingchang ; MENG, Qinglong: Iterative learning control in large scale HVAC system. In: *Intelligent Control and Automation (WCICA), 2010 8th World Congress on*, 2010, S. 5063–5066