Erstellung einer Funktionsbibliothek zur energetischen Simulation netzgekoppelter Batteriespeichersysteme

Fachbereich Technik
Abteilung Elektrotechnik und Informatik
der Hochschule Emden-Leer

Abschlussarbeit
zur Erlangung des akademischen Grades
Bachelor of Engineering
vorgelegt von

Kai Rösken
Matr. Nr. 7010216
geboren am 14.10.1991 in Leer
im Dezember 2020

Erstprüfer: Johannes Rolink, Prof. Dr.-Ing.
Zweitprüfer: Tjarko Tjaden, M. Sc.

Rechtliche Erklärungen

Rechtliche Erklärung

Soweit meine Rechte berührt sind, erkläre ich mich einverstanden, dass die vorliegende Arbeit Angehörigen der Hochschule Emden-Leer für Studium / Lehre / Forschung uneingeschränkt zugänglich gemacht werden kann.

Eidesstattliche Versicherung

Ich, der/die Unterzeichnende, erkläre hiermit an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Alle Quellenangaben und Zitate sind richtig und vollständig wiedergegeben und in den jeweiligen Kapiteln und im Literaturverzeichnis wiedergegeben. Die vorliegende Arbeit wurde nicht in dieser oder einer ähnlichen Form ganz oder in Teilen zur Erlangung eines akademischen Abschlussgrades oder einer anderen Prüfungsleistung eingereicht. Mit ist bekannt, dass falsche Angaben im Zusammenhang mit dieser Erklärung strafrechtlich verfolgt werden können.

Datum: _____ Unterschrift: _____

Abstract

Um einen Beitrag zur Bewältigung der Klimakrise zu leisten, werden verschiedene Ansätze verfolgt. Einer dieser Ansätze ist die elektrische Energieversorgung aus regenerativen Energiequellen. Unter den Einsatz von PV-Systemen kann die Energie der Sonne genutzt werden. Um das volle Potenzial dieser Energiequelle auszunutzen, braucht es Speichermöglichkeiten. Diese können in den Sonnenstunden geladen und bei Bedarf entladen werden.

Das Ziel dieser Arbeit ist die Entwicklung einer quelloffenen Funktionsbibliothek zur energetischen Simulation netzgekoppelter PV-Batteriespeichersysteme. Eine quelloffene Auslegung dieser Funktionsbibliothek ermöglicht neben der eigenständigen Nutzung, die Möglichkeit der Integration ihrer Funktionen in weitere Auslegungstools. Unter Zuhilfenahme dieser Funktionsbibliothek lässt sich der Einsatz von PV-Batteriespeichern simulieren und planen. Die Basis der Modelle zur Simulation von PV-Batteriespeichersystemen bildet das MATLAB-Tool PerMod. Es enthält Modelle zur Simulation von PV-Batteriespeichersystemen.

Vorausgehend zu der Entwicklung der Funktionsbibliothek wurden Anforderungen definiert. Die wichtigsten Anforderungen sind eine quelloffenen Auslegung des Programmcodes und die Möglichkeit diese zu jeder Zeit um Funktionen zu erweitern. Aufbauend auf diesen Anforderungen wurde ein Konzept entwickelt. Es sieht sowohl die Entwicklung neuer Funktionen, so wie die Erweiterungen der PerMod Modelle vor. Zur Bewertung der Simulationsergebnisse wurden sie zunächst mit den Berechnungen der PerMod Modelle verglichen. Neben diesem Vergleich ist zusätzlich ein Vergleich der Performance durchgeführt worden. Hierzu ist der Arbeitsspeicherverbrauch und die Rechenzeit der Simulation eines PV-Batteriespeichersystems verglichen worden. Später folgte ein Vergleich der Simulation als digitaler Zwilling zu einem realen Batteriespeichersystem.

Die Ergebnisse der erweiterten Modelle zeigen nur geringe Abweichungen. Der Vergleich der Funktionsbibliothek und den Messungen eines realen Batteriespeichersystems zeigen, dass die Modelle ein reales System hinreichend genau nachbilden. Neben einer eigenständigen Nutzung lässt sich durch eine quelloffenes Auslegung in weitere Auslegungstools integrieren.

Inhaltsverzeichnis

Rechtliche Erklärung ii

Eidesstattliche Versicherung ii

Abstract iv

Abbildungsverzeichnis iv

Tabellenverzeichnis v

Abkürzungsverzeichnis vi

Nomenklatur vii

1 Einleitung 2
- 1.1 Motivation . . . 2
- 1.2 Zielsetzung . . . 3
- 1.3 Aufgabenbeschreibung . . . 3
- 1.4 Struktur der Arbeit . . . 4

2 Grundlagen zu PerMod 6
- 2.1 Einführung . . . 6
- 2.2 Anwendungsbereiche und abgebildete Systemtopologien . . . 7
- 2.3 Simulationsumfang der Modelle . . . 8
- 2.4 Struktureller Ablauf der Modelle . . . 12
- 2.5 Referenzfälle und Parameter . . . 19

3 Konzeption von openBatLib 20
- 3.1 Anforderungen an openBatLib . . . 20
- 3.2 Vergleich der Anforderung mit PerMod . . . 21
- 3.3 Lösungsansätze . . . 23

4 Entwicklung von openBatLib 26
- 4.1 Konzeptentwicklung nach der MVC-Architektur . . . 26
- 4.2 Erweiterung und Übersetzung der PerMod Modelle . . . 28
- 4.3 Schnittstelle zu realen Systemen . . . 31
- 4.4 Wrapper-Klassen . . . 32

5 Evaluierung von openBatLib 34
- 5.1 Vergleich zu PerMod . . . 34
  - 5.1.1 Vergleich der Simulationsergebnisse . . . 34

5.1.2 Vergleich der Performance 40
5.2 Vergleich zum realen Speichersystem 41

6 Zusammenfassung **48**
6.1 Fazit 49
6.2 Ausblick 50

Literaturverzeichnis **52**

Abbildungsverzeichnis

1 Durch PerMod abgebildete Systemtopologien 7
2 Struktureller Ablauf des AC-gekoppelten PV-Batteriesystems mit P_{L} als Last und P_{R} als Residualleistung. 13
3 Übersicht über den Programmablauf der MVC-Architektur. 24
4 Systematische Übersicht des Konzepts nach der MVC Architektur von openBatLib. 27
5 Erweiterter Ablauf des AC-gekoppelten Batteriemodells. 30
6 Simulierte Leistungsflüsse des AC-gekoppelten Systems am 29. Juni 2016. 39
7 Wochenverlauf des Erzeuger- und Lastprofils. 42
8 Wochenverlauf des Ladezustands der Batterie 44
9 Wochendauerlinie der Ladezustände der Batterie. 44
10 Ermittelte Wirkungsgrade des Batteriesystems. 45
11 Umgesetzte Energiemengen einer Woche. 46
12 Darstellung der lastabhängigen Ladeeffizienz. 47
13 Darstellung der lastabhängigen Entladeeffizienz. 47

Tabellenverzeichnis

1 Durch PerMod modellierte Verlustmechanismen 9
2 Übersicht der Umwandlungspfade 11
3 Übersicht der verfügbaren Systeme 19
4 Vergleich der Eigenschaften von PerMod zu den Anforderungen an openBatLib. 22
5 Energiesummen und Abweichungen des AC-gekoppelten Systems in kWh. 35
6 Energiesummen und Abweichungen des DC-gekoppelten Systems in kWh. 37
7 Energiesummen und Abweichungen des PV-gekoppelten Systems in kWh. 38
8 Rechenzeiten der Batteriespeichermodelle in Sekunden bei einer Simulationsschrittweite von $\Delta t = 1\,\mathrm{s}$ für ein Jahr 41

Abkürzungsverzeichnis

MVC	Model-View-Control
PV	Photovoltaik
HiL	Hardware in the Loop
SoC	State of Charge engl. für Ladezustand
HTW	Hochschule für Technik und Wirtschaft
AC	(alternating current engl. für Wechselstrom)
DC	(direct current engl. für Gleichstrom)
MPPT	Maximum Power Point Tracking engl. für Maximalleistungspunkt Suche
BVES	Bundesverband Energiespeicher e.V.
BSW	Bundesverband Solarwirtschaft e.V.
PerMod	Performance Simulation Model for PV-Battery Systems

Nomenklatur

Energien

E_{AC2BS}	AC-Input des Batteriesystems	[Ws]
E_{AC2G}	In das Netz eingespeiste Energie	[Ws]
E_{AC2PVBS}	AC-Input des PV-Batteriesystems	[Ws]
$E_{\mathrm{Bat,\ in}}$	DC-Input der Batterie	[Ws]
$E_{\mathrm{Bat,\ out}}$	DC-Output der Batterie	[Ws]
E_{BS2AC}	AC-Output des Batteriesystems	[Ws]
E_{BS2G}	Aus der Batterie in das Netz eingespeiste Energie	[Ws]
E_{BS2L}	Energiebedarfsdeckung durch das Batteriesystem	[Ws]
E_{CT}	Abgeregelte PV-Energie	[Ws]
E_{G2AC}	Aus dem Netz bezogene Energie	[Ws]
E_{G2BS}	Ladeenergie der Batterie aus dem Netz	[Ws]
E_{G2L}	Bedarfsdeckung aus dem Netz bezogene Energie	[Ws]
E_{G2PVBS}	Bedarfsdeckung des Batteriesystems aus dem Netz	[Ws]
E_{L}	Energiedarf	[Ws]
E_{Peri}	Energiebedarf der peripheren Bauteile	[Ws]
E_{PV2BS}	PV-Ladeenergie der Batterie	[Ws]
E_{PV2G}	In das Netz eingespeiste PV-Energie	[Ws]
E_{PV2L}	Direktverbrauch der PV-Energie	[Ws]
E_{PVBS2AC}	AC-Output des PV-Batteriesystems	[Ws]
E_{PVBS2L}	Bedarfsdeckung durch das PV-Batteriesystem	[Ws]
E_{PVS}	AC-Output des PV-Systems	[Ws]
E_{PV}	DC-Output des PV-Generators inklusive Begrenzungen	[Ws]

Leistungen

$P_{\mathrm{AC2BAT,\ loss}}$	Ladeverlustleistung des Batteriesytems	[W]
$P_{\mathrm{Bat,\ SB}}$	Durch Standby-Verluste begrenzte DC-Ladeleistung der Batterie	[W]
$P_{\mathrm{BAT2AC,\ loss}}$	Entladeverlustleistung des Batteriesytems	[W]
P_{Bat}	DC-Ladeleistung der Batterie	[W]
$P_{\mathrm{BS,\ \tau}}$	Durch Einschwingzeit begrenzte Leistung des Konverters	[W]
$P_{\mathrm{BS,\ in}}$	AC-Eingangsleistung des Batteriekonverters	[W]
$P_{\mathrm{BS,\ lim}}$	Begrenzte AC-Eingangsleistung des Batteriesytems	[W]
$P_{\mathrm{BS,\ lim}}$	Durch Regelabweichung begrenzte AC-Eingangsleistung des Batteriesystems	[W]
$P_{\mathrm{BS,\ SB}}$	Durch Standby-Verluste begrenzte AC-Leistung des Batteriesystems	[W]

$P_{\mathrm{BS,\ tot}}$	Durch Totzeit begrenzte Leistung des Batteriesytems	[W]
P_{BS}	AC-Leistung des Batteriesystems	[W]
P_{Peri}	Verlustleistung peripherer Baugruppen	[W]
$P_{\mathrm{Pv,\ lim}}$	Begrenzte DC-Eingangsleistung des PV-Wechselrichters	[W]
$P_{\mathrm{PV,\ out}}$	Potenzielle AC-Ausgangsleistung des PV-Wechselrichters	[W]
$P_{\mathrm{PV2AC,\ loss}}$	Verlustleistung des PV-Wechselrichters	[W]
P_{PVS}	Umgesetzte AC-Ausgangsleistung des PV-Wechselrichters	[W]
P_{PV}	PV-Leistung	[W]
P_{R}	Residualleistung	[W]

1 Einleitung

1.1 Motivation

Die Erzeugung elektrischer Energie und dessen Transport und Verteilung befindet sich zunehmend im Wandel. Durch die vermehrt fluktuierende Energieversorgung [1] aufgrund der Volatilität regenerativer Energien und insbesondere der Photovoltaik (PV) wird mittel- bis langfristig der Bedarf an Stromspeichern in den Verteilnetzen steigen [2]. Im Jahr 2019 wurden mit steigender Tendenz mehr als 60.000 Batteriespeicher errichtet, sodass bereits heute mehr als 50 % aller neuerrichteten PV-Anlagen mit einem Batteriespeicher ausgestattet werden [3].

Die Integration von Energiespeichersystemen in das Netz und auch in den Heimbereich bedarf einer sorgfältigen Planung. Zurzeit existiert weder eine VDI-Richtlinie noch eine DIN-Norm, die zur Unterstützung der Planung und Auslegung von Batteriespeichern zur Erhöhung der Eigenversorgung herangezogen werden könnte. Planer und Installateure sind auf Hinweise aus Fachzeitschriften oder kommerzielle Simulationssoftware und Tools von Herstellern angewiesen. Hier besteht Grund zur Annahme, dass Batteriespeicher aus ökonomischer Sicht, nicht immer optimal ausgelegt werden. Die Standardisierung von Datenblattangaben stationärer Batteriespeicher schreitet durch Normarbeitskreise wie "Kennwerte von stationären Batteriespeichern"(AK 371.0.9.) der DKE (Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE) stetig voran [4]. Frei zugängliche Ergebnisse diverser Forschungsprojekte wie EffiBat [5] und das Stromspeicher-Monitoring [6], mit dem Fokus auf die Erstellung und Validierung von Simulationsmodellen, bieten eine Grundlage zur Entwicklung verlässlicher Auslegungstools für stationäre Batteriespeichersysteme. Diese lassen sich frei nutzen und tragen zusätzlich dazu bei, Richtlinien und Normen zu definieren.

1.2 Zielsetzung

Das Ziel dieser Arbeit ist die Erstellung einer quelloffenen Funktionsbibliothek zur energetischen Simulation netzgekoppelter Batteriespeichersysteme, basierend auf des unter der MIT-Lizenz veröffentlichten MATLAB-Tools Performance Simulation Model for PV-Battery Systems (PerMod) 2.1 der HTW Berlin [7].

Um die Funktionsbibliothek sowohl für die Integration in Auslegungstools als auch für die Anwendung in Hardware in the Loop (HiL) Umgebungen oder als digitaler Zwilling nutzbar zu machen, erfolgt die Umsetzung in eine Model-View-Control (MVC) Architektur.

1.3 Aufgabenbeschreibung

Zur Erfüllung der Zielsetzung ergeben sich mehrere Aufgabenschritte. Zunächst erfolgt eine Einarbeitung in das MATLAB Tool PerMod, um ein Verständnis für die verwendeten Funktionen und Abläufe aufzubauen. Darauf folgen die Planung und der Entwurf der MVC Architektur. Diese Architektur sieht eine Unterteilung in drei Komponenten vor. Als Kommunikationsschnittstelle zum User dient die VIEW Komponente. Sie liefert eine grafische Ausgabe des zeitlichen Verlaufs der DC- und AC-seitigen Leistungen sowie des State of Charge (den Ladezustand) der Batterie. Sämtliche Funktionen zu Berechnungen und deren Ergebnisse sind in der MODEL Komponente enthalten. Die CONTROL Komponente ist die Steuerungskomponente und koordiniert die Aktionen der VIEW und MODEL Komponente. Sie ist maßgeblich für den Datenaustausch und die Kommunikation zwischen den beiden Modulen zuständig.

Alle zur Erfüllung der Zielsetzung benötigten Funktionen werden in diese MVC Architektur übertragen und in der Programmiersprache Python verfasst. Durch eine quelloffene Auslegung bietet sie sich als gemeinsame Sprache für die Integration in Auslegungstools an. Weiter bildet sie die Grundlage zu einer möglichen Weiterentwicklung der Funktionsbibliothek hin zu einer Web-Anwendung.

Eine beispielhafte Demonstration der Funktionsbibliothek erfolgt abschließend unter der Vorgabe einer frei gewählten, einwöchigen Zeitreihe der Lade- und Entlade-

leistungsvorgaben. Hierbei werden die realisierten Lade- und Entladeleistungen sowie der SoC eines realen Batteriespeichers mit dessen über die Funktionsbibliothek abgebildeten digitalen Zwilling verglichen.

1.4 Struktur der Arbeit

Kapitel 2 stellt zunächst die Grundlagen der Tools PerMod vor. Es wird auf die Hintergründe der Entwicklung von PerMod eingegangen, sowie dessen Anwendungsbereiche aufgezeigt. Darauf aufbauend beschreibt Kapitel 2 die enthaltenen Modelle zur Simulation von PV-Wechselrichtern und Batteriespeichersysteme. Dazu wird zunächst der Simulationsumfang und der strukturelle Ablauf der Modelle dargestellt.

Nach den Grundlagen findet in Kapitel 3 eine Konzeption der zu entwickelnden Funktionsbibliothek statt. Es definiert hierzu zunächst die Anforderungen an die Funktionsbibliothek. Die so definierten Anforderungen werden mit den Eigenschaften von PerMod verglichen. Aus diesem Vergleich ergeben sich Lösungsansätze zur Erstellung der Funktionsbibliothek.

Sind die Lösungsansätze erörtert, findet ab Kapitel 4 die Dokumentation der Entwicklung der Funktionsbibliothek statt. Ab diesem Zeitpunkt trägt die Funktionsbibliothek den Eigennamen openBatLib. Nach der Dokumentation schließt eine Evaluierung in Kapitel 5 die Entwicklung von openBatLib im Rahmen dieser Arbeit ab. Im Laufe der Evaluierung werden die Simulationsergebnisse von openBatLib mit denen von PerMod verglichen. Zusätzlich findet ein Vergleich der Performance zu beiden Tools statt. Abgeschlossen wird die Evaluierung in dem die Simulationsergebnisse von openBatLib mit Messwerten eines realen Batteriespeichersystems verglichen werden. Hierzu simuliert openBatLib das reale System über den Simulationszeitraum von einer Woche. Mit den gleichen Eingangsdaten als Sollwertvorgaben findet eine Messreihe zur Bestimmung der Leistungsflüsse des realen Systems über den gleichen Zeitraum statt.

Den Abschluss dieser Arbeit bildet die Zusammenfassung. Sie setzt sich aus einem Fazit und einem Ausblick zusammen.

2 Grundlagen zu PerMod

Im Fokus der Grundlagen steht das in dieser Arbeit verwendete MATLAB-Tool PerMod. Es dient als Ausgangspunkt zur Entwicklung der Funktionsbibliothek openBatLib. Um einen Überblick über das MATLAB Tool PerMod zu geben, führen die folgenden Unterkapitel die Eigenschaften des Tools auf. Das erste Unterkapitel widmet sich in einer Einführung der Entwicklung von PerMod. Darauf folgt eine Beschreibung, für welche Anwendungsbereiche PerMod konzipiert ist. Es geht zusätzlich auf die durch PerMod abgedeckten Systemtopologien von Photovoltaik (PV)-Batteriespeichersystemen ein. Der nächste Abschnitt beschreibt den Umfang der Simulationen der enthaltenen Modelle. Dies vermittelt einen ersten Eindruck über die möglichen Funktionalitäten von openBatLib. Den Schluss bildet eine Beschreibung der Referenzfälle. Sie dienen mit Erzeuger- und Lastprofilen als Grundlage der Simulationen. Die hier aufgeführten Informationen entstammen der Dokumentation zu PerMod in der Version 2.1 [7].

2.1 Einführung

PerMod (Performance Simulation Model for PV-Battery Systems) ist ein MATLAB Tool zum Simulieren von Energie- und Leistungsflüssen in PV-Batteriespeichersystemen. Entwickelt wurde es an der Hochschule für Technik und Wirtschaft (HTW) Berlin.

Die Arbeiten zu diesem Tool fanden parallel zu der Entwicklung des Effizienzleitfadens für PV-Speichersysteme statt. Der Leitfaden beschäftigt sich mit der Charakterisierung der Wirkungsgrade, des Standby-Verbrauchs und der Regelungseffizienz von stationären PV-Batteriespeichersystemen [8] [9]. PerMod wurde als quelloffenes Simulationsmodell unter der MIT-Lizenz veröffentlicht. Dadurch ist es möglich, ohne Einschränkung der Rechte die Software zu nutzen, zu ändern und zu veröffentlichen. Sie bietet durch diese Lizenzierung ebenfalls die Möglichkeit einer Implementierung in andere Softwareanwendungen und einer kommerziellen Nutzung des Tools [10].

2.2 Anwendungsbereiche und abgebildete Systemtopologien

Das Ziel der Entwicklung von PerMod war die Bewertung der Energieeffizienz netzgekoppelter PV-Batteriespeichersysteme für Wohnanwendungen. Das Tool simuliert hierzu die Leistungsflüsse dieser Systeme. Die erstellten Modelle verarbeiten Zeitreihen wie Erzeuger- und Lastprofile in sekündlicher Auflösung über einen Zeitraum von einem Jahr.

Insgesamt bildet das Tool drei Systemtopologien von PV-Batteriespeichersystemen ab. Diese Topologieformen binden den Batteriespeicher jeweils auf verschiedene Weisen ein. Eine schematische Übersicht der Topologien ist in Abbildung 1 dargestellt.

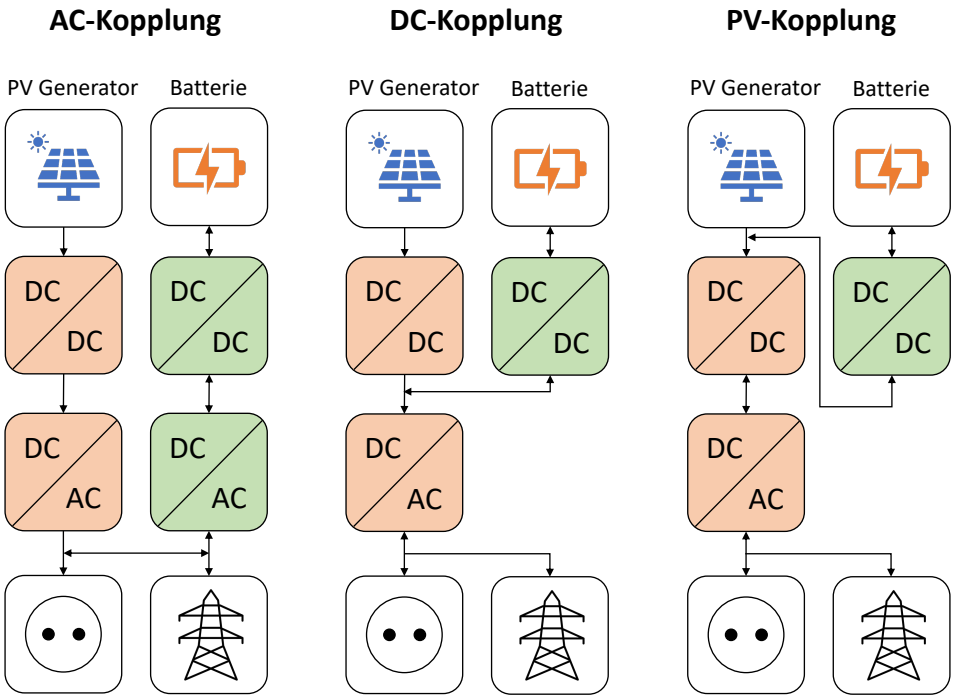

Abbildung 1: Durch PerMod abgebildete Systemtopologien

PerMod unterscheidet die Batteriespeichersysteme in eine (alternating current engl. für Wechselstrom) (AC)-Kopplung, eine (direct current engl. für Gleichstrom) (DC)-Kopplung oder eine PV-Generatorkopplung des Batteriespeichers. Ein PV-Generator ist ein Zusammenschluss einzelner PV-Stränge aus mehreren

PV-Modulen. Die PV-Generatorkopplung ist in dieser Arbeit verkürzt als PV--Kopplung bezeichnet. PerMod beinhaltet neben den Funktionen zur Simulation der Batteriespeicher solche, die PV-Wechselrichter simulieren.

Bei einer AC-Kopplung des Batteriespeichers ist der PV-Generator über einen PV-Wechselrichter an das Hausnetz angeschlossen. Der PV-Wechselrichter wandelt den Gleichstrom des PV-Generators in einen Wechselstrom. Das Laden und Entladen der Batterie erfolgt über einen am selben Knotenpunkt installierten bidirektionalen Batteriekonverter. Der Konverter wandelt den Ladestrom den Batterie in einen Gleichstrom und den Entladestrom in einen Wechselstrom.

Eine DC-Kopplung bindet den Batteriespeicher in den Gleichstromzwischenkreis des PV-Wechselrichters ein. Abhängig von der Einsatzstrategie lädt der PV-Generator die Batterie oder dient direkt der Lastdeckung und Netzeinspeisung. Wechselrichterbrücken teilen sich in unidirektionale und bidirektionale Ausführungen auf. Eine bidirektionale Ausführung der Wechselrichterbrücke bietet die Möglichkeit, die Batterie zusätzlich aus dem Wechselstromnetz zu laden.

Die dritte abgebildete Systemtopologie ist die PV-Kopplung. In dieser Topologieform ist der Batteriespeicher über einen Batteriekonverter zwischen dem PV-Generator und einem PV-Wechselrichter eingebunden. Der PV-Generator lädt die Batterie direkt über den vorgeschalteten Batteriekonverter. Die Einspeisung der PV-Leistung in das Hausnetz, sowie die Entladung der Batterie in das Netz erfolgen über den PV-Wechselrichter.

2.3 Simulationsumfang der Modelle

Dieses Unterkapitel beschäftigt sich mit dem Simulationsumfang der Modelle von PerMod. Es zeigt auf welche Verlustmechanismen die Modelle berücksichtigen, beziehungsweise vernachlässigen.

Die Leistungsflusssimulationen der Modelle stützen sich auf nach dem Effizienzleitfaden ermittelte Parameter [8]. Dieser Leitfaden gibt Messverfahren zur Bestimmung verschiedener Größen von PV-Batteriesystemen vor. Die ermittelten Größen wie Batteriekapazität oder Umwandlungswirkungsgrade bilden die Grundlage der

Simulationen durch die Modelle. Folglich begrenzen die Messverfahren den maximal zu erreichenden Detaillierungsgrad der Modelle [7]. Systemeigenschaften und Verlustfaktoren lassen sich nur in einem begrenzten Rahmen nachbilden [9]. PerMod berücksichtigt Dimensionierungs-, Umwandlungs-, Energiemanagement- und Standby-Verluste [7]. Die Tabelle 1 gibt einen Überblick über die modellierten Verlustmechanismen.

Tabelle 1: Durch PerMod modellierte Verlustmechanismen

Dimensionierungsverluste	- max. Nennleistung der Systemkomponenten
Umwandlungsverluste	- Leistungsabhängigkeit des Umwandlungswirkungsgrades der Leistungselektronik bei Nennspannung - Mittlere Batterieeffizienz
Regelungsverluste	- Stationäre Effizienz der MPPT-Einheit - Hysterese der Ladesteuerung - Mittlere stationäre Regelabweichung der Lade- und Entladeleistung - Mittlere Totzeit und Einschwingzeit der Lade- und Entladeleistung
Energiemanagementverluste	- Einschränkung der PV-Leistung (Begrenzung der Einspeiseleistung auf 70% der PV-Nennleistung)
Standby-Verluste	- Stromverbrauch der Leistungselektronik im Standby-Betrieb - Stromverbrauch peripherer Komponenten (z.B. Wechselstromsensoren, Energiemanager usw.)

Dimensionierungsverluste entstehen durch Leistungsbeschränkungen der Systemkomponenten wie Wechselrichter und Konverter. Die Leistungsfähigkeit eines PV-Batteriesystems ist durch Begrenzungen der PV-Wechselrichter und Batteriekon-

verter beschränkt. Der Einfluss der Leistungsbegrenzungen des Batteriekonverters zeigt sich bei Lastspitzen insofern, dass bei Überschreiten der Nennleistung, sie nur zum Teil durch die Batterie gedeckt werden können. Ähnlich verhält es sich bei Erzeugerleistungen größer der Nennleistung. Hier kann die Batterie nur mit einem Teil der zur Verfügung stehenden Leistung geladen werden. Die Modelle berücksichtigen diese maximalen Nennleistungen der Systemkomponenten.

In der Leistungselektronik und in der Batterie kommt es zu Umwandlungsverlusten. Diese sind in den Modellen als lastabhängige Wirkungsgrade der Leistungselektronik und einer mittleren Batterieeffiziens berücksichtigt. Wirkungsgrade sind bei unterschiedlicher Auslastung nicht konstant. Vielmehr zeigt sich eine Abhängigkeit zur Last.

Hinzu kommen Regelungsverluste die auf Steuergeräte zurückzuführen sind. Diese Verluste umfassen die stationäre Effizienz der MPPT-Einheit, eine Hysterese der Ladesteuerung und mittlere stationäre Regelabweichungen des PV-Batteriespeichersystems. Weitere wichtige Größen dieser Kategorie sind die mittlere Totzeit und Einschwingzeit der Regelsysteme.

Durch das EEG (Erneuerbare Energien Gesetz) ist eine Begrenzung der Netzeinspeisung auf 70 % der PV-Nennleistung vorgeschrieben [9]. Zur Einhaltung dieser Regularien ist diese Begrenzung als Energiemanagementsystemverlust implementiert.

Stromverbräuche der Systemkomponenten im Standby-Modus verursachen Standby-Verluste. Hierzu zählen die Verbräuche der Leistungselektronik und weiterer peripherer Komponenten wie Sensoren oder Displays.

Der Effizienzleitfaden sieht keine Verfahren zur Bestimmung der Batteriealterung vor. Eine Alterung einer Batterie bedeutet eine Verschlechterung ihrer Kenndaten wie der Kapazität oder den Innenwiderstand. Die Alterung einer Batterie unterteilt sich in zwei Kategorien. Eine Batterie kann kalendarisch oder zyklisch altern. Die kalendarische Alterung bezeichnet eine Alterung der Batterie, wenn sie ungenutzt bleibt oder lagert. Die zyklische Alterung bezieht sich auf ein Alterungsprozess durch ständiges Laden und Entladen der Batterie [11]. Verluste durch Alterungsprozesse finden in den Modellen keine Beachtung. PerMod bildet folglich

PV-Batteriespeichersysteme nur im Neuzustand ab [7].

Eine Charakterisierung von PV-Generatoren findet nach dem Effizienzleitfaden nicht statt. Aus diesem Grund greifen die Modelle auf Datenblattangaben und etablierte Modelle zurück [9].

An dieser Stelle ist es wichtig zu erwähnen, dass die entwickelten Modelle die Komponenten eines PV-Batteriespeichersystems nicht auf physikalischer Ebene nachbilden. Vielmehr berechnen sie Verlustleistungen der Umwandlungspfade von Gleich- zu Wechselstrom zwischen den verschiedenen Systemkomponenten wie Batteriekonverter und PV-Wechselrichtern. Insgesamt simuliert PerMod fünf Umwandlungspfade. Die folgenden Absätze erläutern diese in verkürzter Form. Für den weiteren Verlauf sind die Umwandlungspfade mit Abkürzungen versehen. In der Tabelle 2 findet sich eine Übersicht dieser Umwandlungspfade. Sie zeigt welche Abkürzung für welchen Umwandlungspfad steht. Zusätzlich gibt sie Informationen darüber, in welchen Topologien die unterschiedlichen Umwandlungspfade auftreten.

Tabelle 2: Übersicht der Umwandlungspfade

		Topologie		
Wandlungspfad	Abkürzung	AC	DC	PV
PV AC-Ausgangsleistung	PV2AC	Optional	Ja	Ja
PV-Batterieladung	PV2BAT	Optional	Ja	Ja
AC-Batterieentladung	BAT2AC	Ja	Ja	Optional
AC-Batterieladung	AC2BAT	Ja	Optional	Optional
DC-Batterieentladung	BAT2PV	Nein	Nein	Ja

Die Umwandlung der PV-Leistung in eine AC-Leistung ist als PV2AC bezeichnet. Dieser Umwandlungspfad beschreibt den Eigenverbrauch oder die Netzeinspeisung der PV-Erzeugung. Verluste entstehen durch die Wandlung der DC-Leistung des PV-Generators in eine AC-Leistung durch den PV-Wechselrichter. Er findet sich in allen Systemtopologien.

Wird die Batterie durch die PV-Leistung geladen, also die PV-Erzeugung in eine DC-Batterieladeleistung konvertiert, ist dies als PV2BAT bezeichnet. Dieser Pfad findet sich in der DC- und der PV-Kopplung des Batteriespeichers wieder.

Die Ladung der Batterie aus dem Netz ist als AC2BAT bezeichnet. Er beschreibt die Umwandlung einer AC-Leistung in eine DC-Ladeleistung durch den Wechselrichter (Batteriekonverter) des Batteriesystems. Bei einer DC- oder PV-Kopplung der Batterie muss der PV-Wechselrichter dazu in einer bidirektionalen Ausführung ausgelegt sein.

Wird die Batterie in das Netz entladen, geschieht dies über den BAT2AC Umwandlungspfad. Über diesen Umwandlungspfad wird die gespeicherte Energie der Batterie zur Deckung der Last verwendet oder in das Netz eingespeist.

Ist eine Batterie direkt mit dem PV-Generator gekoppelt, entlädt sich die Batterie in den gleichen Knotenpunkt wie die PV-Leistung. Dieser Pfad ist als BAT2PV bezeichnet. Die PV- und DC-Kopplung sehen eine DC-seitige Einbindung der Batterie vor. In diesem Pfad steuert der Konverter des Batteriesystems die Lade- und Entladeleistung der Batterie. Durch die Anpassung der PV-Erzeugung auf ein zur Nennelistung des PV-Wechselrichters kompatibles Leistungsniveau entstehen Verluste. Diese Verluste sind in diesem Pfad berücksichtigt.

2.4 Struktureller Ablauf der Modelle

Der Simulationsablauf der Modelle teilt sich in zwei Hauptbestandteile. Den ersten Hauptteil bildet die Modellierung des PV-Systems, den zweiten Teil des Batteriesystem. Dieser Abschnitt beschreibt stellvertretend für die übrigen Modelle das AC-gekoppelte Batteriesystem. Eine Abbildung zum strukturellen Ablauf findet sich in der Abbildung 2 auf Seite 13.

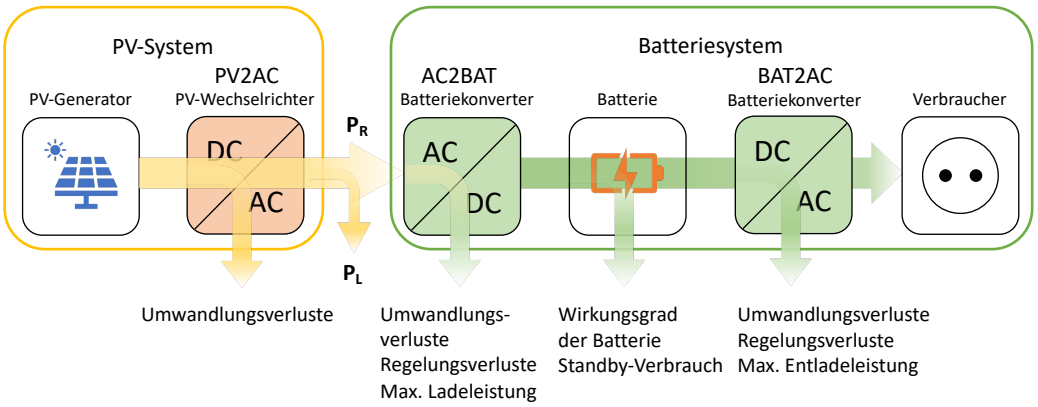

Abbildung 2: Struktureller Ablauf des AC-gekoppelten PV-Batteriesystems mit P_{L} als Last und P_{R} als Residualleistung.

Zunächst berechnet PerMod die Umwandlungsverluste des PV-Systems. Sie entstehen durch die Wandlung einer DC-Größe in eine AC-Größe durch den PV-Wechselrichter.

Durch seine maximale Eingangsleistung $P_{\mathrm{PV2AC,\,max}}$ begrenzt der PV-Wechselrichter die erzeugte Leistung P_{PV} des PV-Generators auf $P_{\mathrm{PV,\,lim}}$ nach Gleichung (1).

$$P_{\mathrm{PV,\,lim}} = \min\left(P_{\mathrm{PV}}, P_{\mathrm{PV2AC,\,max}}\right) \tag{1}$$

Der Effizienzleitfaden gibt ein Messverfahren zur Ermittlung der Verlustleistungen der Umwandlungspfade vor. Die Wirkungsgrade einer Umwandlung schwanken über den Leistungsbereich eines Wechselrichters. Aus diesem Grund definiert der Leitfaden Messungen an bestimmten Betriebspunkten. Die Betriebspunkte liegen in einem Bereich von 5 bis 100 % der Nennleistung des Wechselrichters verteilt [8]. Über diese Verlustleistungen lässt sich die Verlustleistungskurve der Wechselrichter über eine quadratische Gleichung annähern. Die Annäherung liefert die drei Koeffizienten a, b und c. Mit diesen Koeffizienten lassen sich die Verlustleistungen der Wechselrichter annähernd bestimmen.

Zur Berechnung der Verlustleistung $P_{\mathrm{PV2AC,\,loss}}$ durch die Wandlung der PV-Leistung in eine AC-Ausgangsleistung durch den PV-Wechselrichter, wird nach der

Gleichung (2) die begrenzte PV-Leistung $P_{\mathrm{PV,\,lim}}$ auf die maximale Eingangsleistung $P_{\mathrm{PV2AC,\,in,\,max}}$ des Wechselrichters auf die Größe p_{PV2AC} normiert.

$$p_{\mathrm{PV2AC}} = \frac{P_{\mathrm{PV,\,lim}}}{P_{\mathrm{PV2AC,\,in,\,max}}} \tag{2}$$

Die Verlustleistung $P_{\mathrm{PV2AC,\,loss}}$ ergibt sich nach der Gleichung (3). Die Gleichung greift dazu auf die aus der Annäherung der Verlustleistungskurve ermittelten Koeffizienten $a_{\mathrm{PV2AC,\,in}}$, $b_{\mathrm{PV2AC,\,in}}$ und $c_{\mathrm{PV2AC,\,in}}$ zurück.

$$P_{\mathrm{PV2AC,\,loss}} = a_{\mathrm{PV2AC,\,in}} \cdot {p_{\mathrm{PV2AC}}}^2 + b_{\mathrm{PV2AC,\,in}} \cdot p_{\mathrm{PV2AC}} + c_{\mathrm{PV2AC,\,in}} \tag{3}$$

Aus der Verlustleistung $P_{\mathrm{PV2AC,\,loss}}$ und der begrenzten PV-Leistung $P_{\mathrm{PV,\,lim}}$ ergibt sich nach Gleichung (4) die potenzielle Ausgangsleistung $P_{\mathrm{PV,\,out}}$ des PV-Wechselrichters.

$$P_{\mathrm{PV,\,out}} = \max\left(0, \left(P_{\mathrm{PV,\,lim}} - P_{\mathrm{PV2AC,\,loss}}\right)\right) \tag{4}$$

Die maximale Ausgangsleistung $P_{\mathrm{PV2AC,\,out,\,max}}$ des PV-Wechselrichters begrenzt die potenzielle Ausgangsleistung $P_{\mathrm{PV,\,out}}$ auf die Größe P_{PVS} nach Gleichung (5).

$$P_{\mathrm{PVS}} = \min\left(P_{\mathrm{PV,\,out}}, P_{\mathrm{PV2AC,\,max}}\right) \tag{5}$$

Ist die Ausgangsleistung des PV-Wechselrichters P_{PVS} gleich null, befindet sich dieser im Standby-Modus. Zu diesen Zeitpunkten gilt es den Standby-Verlust $P_{\mathrm{SB,\,loss}}$ des PV-Wechselrichters zu berücksichtigen. Standby-Verluste sind in PerMod den Verlusten durch die Peripherie $P_{\mathrm{Peri,\,loss}}$ zugeordnet, siehe Gleichung (6). Sie beinhalten neben den Standby-Verlusten alle weiteren Verlustleistungen P_{Peri}, die durch periphere Bauteile wie Displays oder Sensoren entstehen.

$$P_{\mathrm{Peri,\,loss}}(t) = \begin{cases} P_{\mathrm{SB,\,loss}} & \text{für } P_{\mathrm{PVS}}(t) = 0 \\ P_{\mathrm{Peri}}(t) & \text{für } P_{\mathrm{PVS}}(t) \neq 0 \end{cases} \tag{6}$$

Die Differenz aus Ausgangsleistung P_{PVS} des PV-Wechselrichters, der Last P_{L} und den Peripherieverlusten $P_{\mathrm{Peri,\,loss}}$ ergibt eine Residualleistung P_{R} nach Gleichung (7).

$$P_{\mathrm{R}} = P_{\mathrm{PVS}} - P_{\mathrm{L}} - P_{\mathrm{Peri,\,loss}} \tag{7}$$

Die Residualleistung P_{R} dient somit als Sollwertvorgabe für die Regelung des Batteriesystems. Die Regelung des Batteriesystems reagiert auf diese Sollwertvorgabe. Das Modell berücksichtigt verschiedene Eigenschaften einer Regelung. Dies sind Tot- und Einschwingzeiten, sowie stationäre Regelabweichungen.

Die Totzeit t_{tot} der Regelung nimmt nach Gleichung (8) Einfluss auf die AC-seitige Ladeleistung $P_{\mathrm{BS,\,tot}}$ des Batteriesystems.

$$P_{\mathrm{BS,\,tot}} = P_{\mathrm{R}}(t - t_{\mathrm{tot}}) \tag{8}$$

Die mittlere stationäre Regelabweichung $P_{\mathrm{AC2BAT,\,dev}}$, unter Berücksichtigung der minimalen Lade ($P_{\mathrm{AC2BAT,\,min}}$)- und Entladeleistung $P_{\mathrm{BAT2AC,\,min}}$ des Batteriekonverters, begrenzt die Ladeleistung auf $P_{\mathrm{BS,\,abw}}$ nach Gleichung (9).

$$P_{\mathrm{BS,\,abw}} = \begin{cases} \max\left(P_{\mathrm{AC2BAT,\,min}}, (P_{\mathrm{BS,\,tot}} + P_{\mathrm{AC2BAT,\,dev}})\right) & \text{für } P_{\mathrm{BS,\,tot}} \\ & > P_{\mathrm{AC2BAT,\,min}} \\ \min\left(-P_{\mathrm{BAT2AC,\,min}}, (P_{\mathrm{BS,\,tot}} - P_{\mathrm{BAT2AC,\,dev}})\right) & \text{für } P_{\mathrm{BS,\,tot}} \\ & < -P_{\mathrm{BAT2AC\,min}} \\ 0 & \text{sonst} \end{cases} \tag{9}$$

Weiterhin begrenzen die maximale Eingangsleistung $P_{\mathrm{AC2BAT,\,max}}$ und Ausgangsleistung $P_{\mathrm{BAT2AC,\,max}}$ des Batteriekonverters die AC-Ladeleistung $P_{\mathrm{BS,\,in}}$ auf die

Größe $P_{\mathrm{BS,\,lim}}$, siehe dazu die Gleichungen (10) und (11).

$$P_{\mathrm{BS,\,in}} = \begin{cases} P_{\mathrm{AC2BAT,\,max}} & \text{für } P_{\mathrm{BS,\,abw}} > P_{\mathrm{AC2BAT,\,max}} \\ P_{\mathrm{BS,\,abw}} & \text{für } P_{\mathrm{BS,\,abw}} < P_{\mathrm{AC2BAT,\,max}} \end{cases} \tag{10}$$

$$P_{\mathrm{BS,\,lim}} = \begin{cases} -P_{\mathrm{BAT2AC,\,max}} & \text{für } -P_{\mathrm{BS,\,in}} < -P_{\mathrm{BAT2AC,\,max}} \\ P_{\mathrm{BS,\,in}} & \text{für } -P_{\mathrm{BS,\,in}} > -P_{\mathrm{BAT2AC,\,max}} \end{cases} \tag{11}$$

Die Regelung des Batteriesystems benötigt eine Einschwingzeit zum Einstellen der DC-Ladeleistung P_{Bat} des Batteriekonverters. Durch die Gleichung (12) passt das Modell die Größe $P_{\mathrm{BS,\,lim}}$ auf die Einschwingzeit der Regelung des Batteriesystems zu der Leistung $P_{\mathrm{BS},\,\tau}$ an.

$$P_{\mathrm{BS},\,\tau} = P_{\mathrm{BS}}(t - 1\,\mathrm{s}) + (P_{\mathrm{BS,\,lim}} - P_{\mathrm{BS}}(t - 1\,\mathrm{s})) \cdot 1 - e^{-\frac{1\,\mathrm{s}}{\tau}} \tag{12}$$

Die Einschwingzeit ist durch ein PT1-Glied mit einer Zeitkonstante τ modelliert. Die Leistung $P_{\mathrm{BS},\,\tau}$ ergibt sich aus der Zeitkonstante τ, der Zeitschrittweite und der Leistung $P_{\mathrm{BS}}(t-1)$. Da das Modell mit einer festen Zeitschrittweite $\Delta t = 1\,\mathrm{s}$ arbeitet, wird von t eine Sekunde subtrahiert.

Ob das Batteriesystem die Batterie lädt oder entlädt, ist von mehreren Faktoren abhängig. Ein bestimmender Faktor ist der Ladezustand soc_0 der Batterie. Des Weiteren sind es der Schwellwert soc_{H}, die boolesche Variable th und das Vorzeichen der Residualleistung P_{R}. Die Einführung des Schwellwerts soc_{H} unterbindet abrupte Wechsel zwischen dem Lade- und Entladebetrieb. Er verhindert nach kurzzeitiger Entladung der Batterie eine sofortige Nachladung. Der Schwellwert soc_{H} ist auf einen Ladezustand von 98 % festgelegt. Erst bei Unterschreitung lädt das Batteriesystem die Batterie. Ob der Schwellwert soc_{H} in Betracht gezogen wird, steuert die boolesche Variable th. Sie kann die Zustände '1' oder '0' annehmen. Nur wenn sie den Zustand '1' hat, findet der Schwellwert soc_{H} Beachtung.

Zusätzliche Umwandlungsverluste entstehen in Abhängigkeit vom Leistungsdurchsatz durch die Wandlung von Wechselstrom zu Gleichstrom des Batteriewandlers.

Zur Bestimmung der DC-seitigen Batterieladeleistung P_{Bat} berechnet das Batteriemodell zunächst die normierte Größe p_{BS}, siehe Gleichung (13).

$$p_{\mathrm{BS}} = \begin{cases} \frac{P_{\mathrm{BS},\,\tau}}{P_{\mathrm{AC2BAT,\,max}}} & \text{für } P_{\mathrm{BS},\,\tau} > 0 \\ & \quad \wedge\ soc_0 < 1 - th \cdot (1 - soc_{\mathrm{H}}) \\ \frac{|P_{\mathrm{BS},\,\tau}|}{P_{\mathrm{BAT2AC,\,max}}} & \text{für } P_{\mathrm{BS},\,\tau} < 0 \wedge\ soc_0 > 0 \end{cases} \tag{13}$$

Sie normiert $P_{\mathrm{BS},\,\tau}$ auf die maximale Ladeleistung $P_{\mathrm{AC2BAT,\,max}}$, beziehungsweise Entladeleistung $P_{\mathrm{BAT2AC\,max}}$ des Batteriesystems.

Mit der normierten Größe p_{BS} lässt sich nun die Verlustleistung $P_{\mathrm{AC2BAT,\,loss}}$ der AC2BAT-Wandlung, als auch die Verlustleistung $P_{\mathrm{BAT2AC,\,loss}}$ der BAT2AC-Wandlung durch den Batteriekonverter bestimmen, siehe Gleichung (14) und (15).

$$P_{\mathrm{AC2BAT,\,loss}} = a_{\mathrm{AC2BAT,\,in}} \cdot {p_{\mathrm{BS}}}^2 + b_{\mathrm{AC2BAT,\,in}} \cdot p_{\mathrm{BS}} + c_{\mathrm{AC2BAT,\,in}} \tag{14}$$

$$P_{\mathrm{BAT2AC,\,loss}} = a_{\mathrm{BAT2AC,\,out}} \cdot {p_{\mathrm{BS}}}^2 + b_{\mathrm{BAT2AC,\,out}} \cdot p_{\mathrm{BS}} + c_{\mathrm{BAT2AC,\,out}} \tag{15}$$

Nach der Gleichung (16) ergibt sich die DC-seitige Ladeleistung P_{Bat} der Batterie.

$$P_{\mathrm{Bat}} = \begin{cases} \max\left(0, (P_{\mathrm{BS},\,\tau} - P_{\mathrm{AC2BAT,\,loss}})\right) & \text{für } P_{\mathrm{BS},\,\tau} > 0 \wedge soc_0 < 1 - th \cdot (1 - soc_{\mathrm{H}}) \\ P_{\mathrm{BS}} - P_{\mathrm{BAT2AC,\,loss}} & \text{für } P_{\mathrm{BS},\,\tau} < 0 \wedge soc_0 > 0 \\ 0 & \text{sonst} \end{cases} \tag{16}$$

Ist P_{Bat} bestimmt, entscheidet das Modell, ob der Standby-Modus des Batteriesystems aktiv ist. Ist der Standby-Modus aktiv, hat $P_{\mathrm{Bat}}(t)$ zu diesen Zeitpunkten den Wert null. Des Weiteren unterscheidet PerMod zwischen einer geladenen und entladenen Batterie. Ist die Batterie geladen, deckt das Batteriesystem seinen Standby-Bedarf aus der Batterie. Im entladenen Zustand deckt es seinen Bedarf aus dem Netz. So entsteht entweder die DC-Standby-Leistung $P_{\mathrm{Bat,\,SB}}$ nach Glei-

chung 17 oder AC-Leistung $P_{\mathrm{BS,\,SB}}$.

$$P_{\mathrm{Bat,\,SB}} = \begin{cases} P_{\mathrm{Bat}} = -\max\left(0, P_{\mathrm{sys,\,soc0,\,DC}}\right) & \text{für } P_{\mathrm{Bat}} = 0 \wedge soc_0 \leq 0 \\ P_{\mathrm{Bat}} = -\max\left(0, P_{\mathrm{sys,\,soc1,\,DC}}\right) & \text{für } P_{\mathrm{Bat}} = 0 \wedge soc_0 > 0 \end{cases} \tag{17}$$

$$P_{\mathrm{BS,\,SB}} = \begin{cases} P_{\mathrm{sys,\,soc0,\,AC}} & \text{für } P_{\mathrm{Bat}} = 0 \wedge soc_0 \leq 0 \\ P_{\mathrm{sys,\,soc1,\,AC}} & \text{für } P_{\mathrm{Bat}} = 0 \wedge soc_0 > 0 \end{cases} \tag{18}$$

Die DC-Ladeleistung P_{Bat} der Batterie aktualisiert nun abhängig vom Vorzeichen nach der Gleichung (19) den Energiegehalt E_{B} der Batterie. Die Einheit der Batteriekapazität ist kWh. Dementsprechend erfolgt eine Anpassung der Einheit Ws zu kWh.

$$E_{\mathrm{B}} = \begin{cases} E_{\mathrm{B,\,0}} + \dfrac{P_{\mathrm{Bat}} \cdot 1\,\mathrm{s} \cdot \sqrt{\eta_{\mathrm{Bat}}}}{3600} & \text{für } P_{\mathrm{Bat}} > 0 \\ E_{\mathrm{B,\,0}} + \dfrac{P_{\mathrm{Bat}} \cdot 1\,\mathrm{s}}{3600 \cdot \sqrt{\eta_{\mathrm{Bat}}}} & \text{für } P_{\mathrm{Bat}} < 0 \\ E_{\mathrm{B,\,0}} & \text{sonst} \end{cases} \tag{19}$$

Ist der Energiegehalt E_{B} der Batterie aktualisiert, ergibt sich der Ladezustand soc_0 nach Gleichung (20). Dieser ergibt sich aus dem Verhältnis des aktuellen Energiegehalts E_{B} zu der nutzbaren Gesamtkapazität E_{Bat} der Batterie.

$$soc_0 = \frac{E_{\mathrm{B}}}{E_{\mathrm{Bat}}} \tag{20}$$

Nach Gleichung (21) wird die boolesche Variable th an die neue Situation angepasst. Dies vermeidet einen Wechsel zwischen Lade- und Standby-Modus aufgrund des Gleichstromverbrauchs des Batteriesystems.

$$th = \begin{cases} 1 & \text{für } soc_0 > soc_{\mathrm{H}} \wedge soc_0 > 1 \\ 0 & \text{sonst} \end{cases} \tag{21}$$

2.5 Referenzfälle und Parameter

Zusammen mit PerMod stellt die HTW Berlin zwei Referenzfälle bestehend aus Erzeuger- und Lastprofilen und Parameter realer PV-Batteriesysteme zu Verfügung. Die Referenzfälle enthalten Last- und Erzeugerprofile aus dem Jahr 2016. Sie dienen dazu auf dem Markt erhältliche PV-Batteriesysteme hinsichtlich ihrer Batteriekapazität und Nennleistung der Leistungselektronik abzubilden. Im weiteren Verlauf dieser Arbeit werden sie zur Validierung der Modelle von openBatLib herangezogen. Dieses Unterkapitel beschreibt diese Referenzfälle und Parameter.

1. Referenzfall für PV-Batteriesysteme mit einer 5 kWp PV-Nennleistung.
2. Referenzfall für PV-Batteriesysteme mit einer 10 kWp PV-Nennleistung.

Der erste Referenzfall repräsentiert einen Haushalt mit einem jährlichen Bedarf von $E_a = 5010\,\mathrm{kW/a}$ und einer PV-Nennleistung von $P_{\mathrm{PV,\,nenn}} = 5\,\mathrm{kW}p$.

Der zweite Referenzfall erweitert das Lastprofil zusätzlich um eine Wärmepumpe und ein Elektrofahrzeug. Infolgedessen steigt der jährliche Energiebedarf auf $\mathrm{E_a} = 9363\,\mathrm{kW/a}$. Die PV-Nennleistung ist auf $\mathrm{P_{PV\ nenn}} = 10\,\mathrm{kWp}$ erweitert.

Zusätzlich stellt die HTW Berlin Parameter für vier PV-Speichersysteme und zwei PV-Wechselrichter zur Verfügung. Diese Parameter stammen aus Messungen nach dem Effizienzleitfaden. Eine Übersicht der vermessenen Systeme findet sich in Tabelle 3. Sie zeigt neben den Herstellerangaben ebenfalls die Systemtopologien.

Tabelle 3: Übersicht der verfügbaren Systeme

Hersteller	Produkt	Typ	Topologie
Siemens	Junelight Smart Battery 9,9	PV-Batteriesystem	AC
KOSTAL	PLENTICORE plus 5.5 mit BYD Battery-Box H6.4	PV-Batteriesystem	DC
KOSTAL	PLENTICORE plus 10 mit BYD Battery-Box H11.5	PV-Batteriesystem	DC
Fronius	Symo GEN24 10.0 Plus mit BYD Battery-Box H11.5	PV-Batteriesystem	DC
SMA	Sunny Boy 5.0	PV-Wechselrichter	-
SMA	Sunny Tripower 10.0	PV-Wechselrichter	-

3 Konzeption von openBatLib

Dieses Kapitel beschäftigt sich mit der Konzeption der Funktionsbibliothek. Zunächst werden die Anforderungen an die zu entwickelnde Funktionsbibliothek definiert. Im Anschluss dazu findet ein Vergleich von PerMod mit diesen Anforderungen statt. Aus den Ergebnissen des Vergleichs werden schließlich Lösungsansätze erörtert. Für den weiteren Verlauf dieser Arbeit erhält die Funktionsbibliothek den Eigennamen openBatLib.

3.1 Anforderungen an openBatLib

Die Anforderungen an openBatLib ergeben sich aus der Aufgabenbeschreibung nach Kapitel 1.3.

Eine wichtige Anforderung ist eine hohe Modularität ihrer Funktionen. Sie sollen mit einem minimalen Aufwand adaptierbar und wartbar sein. Darüber hinaus soll eine Erweiterung um weitere Funktionen möglich sein, ohne weitergehende Anpassungen an zuvor implementierten Schnittstellen vorzunehmen.

Für einen breiten Anwendungsbereich als Open-Source Software ist zudem eine quelloffene Auslegung des Quellcodes von openBatLib ein großer Vorteil.

Der Einsatz von PV-Batteriespeichersystemen erfolgt unter verschiedenen Strategien. Eine dieser Strategien stellt die Maximierung des Eigenverbrauchs der erzeugten PV-Energie dar. Nach dieser Strategie werden durch die Sollwertvorgabe $P_{\text{max, soll}}$ Überschüsse aus Last und Erzeugung in die Batterie geladen und alle Defizite entladen, siehe Gleichung (22). Diese Betriebsstrategie soll openBatLib abbilden können. Dies erfordert eine Umsetzung der Maximierung als eigenständige Funktion Bibliothek.

$$P_{\text{max soll}} = \max\left((P_{\text{PV}} - P_{\text{Last}}), 0\right) \tag{22}$$

Neben den Simulationsmodellen der Batteriespeichersysteme, wie beschrieben in Kapitel 2.2, soll openBatLib reale Speichersysteme ansprechen können. Dies ermöglicht einen Einsatz der Bibliothek in Kombination mit realen PV-Batterie-

speichersystemen. So können Simulationsergebnisse mit Messungen eines realen PV-Batteriespeichersystems verglichen werden. Ebenso fungiert openBatLib als digitaler Zwilling eines realen Systems. Hierzu bietet es Schnittstellen zur Kommunikation mit realen PV-Batteriespeichersystemen. Diese Schnittstellen ermöglichen es, Ladevorgaben an ein System zu senden und Größen wie den Ladezustand der Batterie abzufragen.

Last- und Erzeugerprofile existieren in unterschiedlichen zeitlichen Auflösungen. Daher sollten die Modelle von openBatLib verschieden aufgelöste Eingangsdaten verarbeiten können. Folglich bietet openBatLib die Eigenschaft, zusätzlich Last- und Erzeugerprofile mit zeitlichen Auflösungen größer einer Sekunde zu interpretieren.

Eine weitere Anforderung besteht darin, die Modelle der Batteriespeichersysteme und der PV-Wechselrichter unabhängig voneinander verwenden zu können. So lassen sich ein Batteriespeichersystem oder PV-Wechselrichter getrennt voneinander simulieren.

openBatLib soll die Möglichkeit bieten, die Modelle mit einzelnen Leistungswerten aufzurufen. Im Gegensatz zu PerMod kann openBatLib so neben der Simulation eines ganzen Jahres, eine Simulation jedes einzelnen Zeitschritts der Eingangsreihe durchführen.

3.2 Vergleich der Anforderung mit PerMod

Da das Tool PerMod als Basis der Entwicklung dient, empfiehlt sich ein Vergleich mit den zuvor gestellten Anforderungen. Dieser Vergleich liefert erste Ansätze über nötige Erweiterungen an den Funktionen PerMods. Eine Gegenüberstellung der Eigenschaften von PerMod und den Anforderungen findet sich in der Tabelle 4.

Bei Betrachtung der Tabelle 4 zeigen sich mehrere Punkte, in denen PerMod nicht den Anforderungen an openBatLib entspricht. Der Vergleich zeigt, dass eine Übernahme der Funktionen aus PerMod nicht ohne Anpassungen möglich ist.

Der Quellcode von PerMod ist für die Software MATLAB verfasst. Dies ist eine proprietäre Software. Der Quellcode ist somit nur in Verbindung mit dieser Soft-

Tabelle 4: Vergleich der Eigenschaften von PerMod zu den Anforderungen an openBatLib.

	PerMod	Anforderungen
Programmiersprache	Matlab	Python
Programmstruktur	seriell	modular
Verbindung zu realen Speichern möglich?	nein	ja
Zeitschrittweite	$\Delta t = 1\,s$	$\Delta t \geq 1\,s$
Aufruf mit Einzelwerten möglich?	nein	ja
Eingangszeitreihen	max. 1 Jahr	beliebig
Batteriespeichermodelle unabhängig nutzbar?	nein	ja

wareumgebung ausführbar. Eine Übersetzung des Quellcodes in eine quelloffene Programmiersprache ist notwendig, um ihn als quelloffen und nicht proprietär zu veröffentlichen.

Die Programmstruktur von PerMod ist seriell ausgelegt. Nach Auswahl des Systems und dem Einlesen der Eingangsdaten, berechnet es eine Jahressimulation in einem Durchlauf. Dies hat den Vorteil einer einfachen Programmstruktur, grenzt jedoch die Einsatzmöglichkeiten ein. Programmabläufe wie der Simulation des PV-Wechselrichters lassen sich nicht getrennt durchführen. An dieser Stelle ist eine Auftrennung des Programmablaufs nötig. Eine modular auslegte Programmstruktur ist aufwändiger, bietet jedoch einen flexibleren Einsatz von openBatLib. Somit lassen sich die Simulationen der Batteriespeichersysteme und PV-Wechselrichter unabhängig voneinander durchführen und untersuchen.

Zur Simulation der Leistungsflüsse akzeptieren die in PerMod enthaltenen Modelle der Batteriespeichersysteme und PV-Wechselrichter lediglich fixe Zeitschrittweiten der Eingangsdaten von $\Delta t = 1\,s$. Eine Anpassung der Modelle für die Verarbeitung von Zeitschrittweiten mit $\Delta t \geq 1\,s$ ist notwendig, um die zuvor definierten Anforderungen dahingehend zu erfüllen.

3.3 Lösungsansätze

Nach dem Vergleich der Anforderungen mit PerMod ergeben sich mehrere Lösungsansätze. Diese werden in diesem Unterkapitel erläutert.

Um openBatLib als quelloffen zu veröffentlichen, wird der Quellcode in der Programmiersprache Python verfasst. Neben MATLAB hat sich Python zunehmend als Standard in Entwicklung und Forschung etabliert [12]. Neben der Entwicklung im Desktop-Bereich, hält Python zunehmend Einzug in die Entwicklung von Web-Tools. Beispiele hierfür sind unter anderen die Web-Frameworks Flask [13] und Django [14]. Somit besteht die Option, die Funktionen aus openBatLib in Web-Anwendung zu integrieren. Python bietet die Möglichkeit auf viele, ebenfalls quelloffene, Module zurückzugreifen. Hervorzuheben sind an dieser Stelle die Module *NumPy* und *pandas*. Durch optimierte Funktionen für vektorielle Operationen bieten diese Module eine Grundlage für wissenschaftliche und technische Anwendungen [12]. Durch eine einfache Syntax und Strukturen fördert sie den Gedanken einer quelloffenen Auslegung. Folglich wird der Quellcode von PerMod zu Python übersetzt.

Um eine hohe Modularität der Funktionen gemäß den Anforderungen an openBatLib zu gewährleisten, werden die Funktionen von PerMod in eine modulare Softwarearchitektur überführt. Die Programmstruktur von openBatLib wird nach der MODEL-VIEW-CONTROL Architektur ausgerichtet. Folgend werden das zugrunde liegende Konzept und die Komponenten dieser Architektur beschrieben. Darüber hinaus wird auf die Funktionen und Rollen der Komponenten im Programmablauf eingegangen.

Die MVC-Architektur kommt vor allem in Web-Applikationen zur Anwendung. Inzwischen hält sie überdies zunehmend Einzug in die Softwareentwicklung für den Desktopbereich. Auch im Bereich von mobilen Apps ist sie üblich. Beispielsweise setzte Apple bei der Entwicklung von iOS-Apps explizit auf das MVC-Konzept [15].

Das Konzept dieser Architektur sieht eine Unterteilung der Softwaresegmente in jeweils eigene Verantwortlichkeiten vor. Dies geschieht durch eine Zuweisung zu klar definierten Komponenten. Eine Veranschaulichung dieser Unterteilung findet

sich in Abbildung 3. Aus dieser Aufteilung des Quellcodes nach Rollen ergeben sich mehrere Vorteile im Vergleich zu einer seriellen Ausrichtung des Programmablaufs. Die Komponenten der MVC-Architektur konzentrieren sich nur auf ihre jeweilige Funktion im Programmablauf. Dazu bieten sie Schnittstellen zur Ein- und Ausgabe von Informationen. Dies vermeidet eine Vermischung von Rollen beziehungsweise Logiken innerhalb des Programmablaufs. Des Weiteren bietet sich so die Möglichkeit, die Komponenten und deren Funktionen unabhängig voneinander zu warten, zu pflegen und zu adaptieren [15]. Die Komponenten können beliebig um weitere Funktionen erweitert werden. So lassen sich Funktionen erweitern oder austauschen, ohne weitergehende Anpassung an den übrigen Komponenten vorzunehmen.

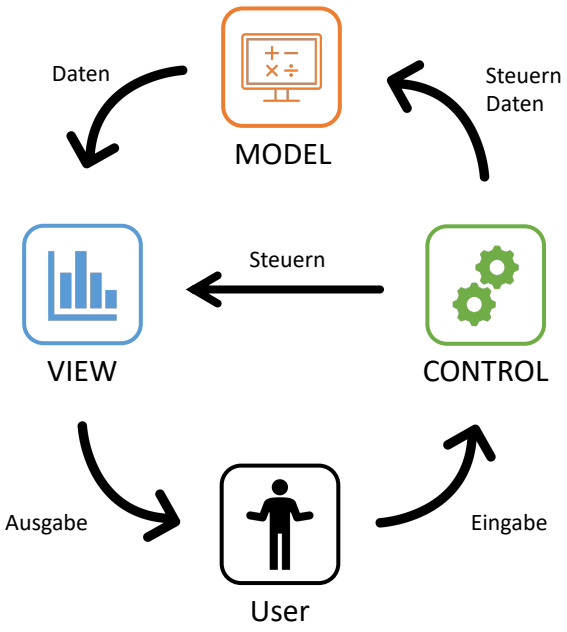

Abbildung 3: Übersicht über den Programmablauf der MVC-Architektur.

Die Modellkomponente (MODEL) ist für Abläufe verantwortlich, die das Verarbeiten und Manipulieren von Daten erfordern. Sie führt die nötigen Operationen und Berechnungen durch und stellt deren Ergebnisse den übrigen Komponenten zur Verfügung. Die Steuerkomponente CONTROL bietet Funktionen, um die Ergebnis-

se der Darstellungskomponente VIEW bereitzustellen. An dieser Stelle gilt es zu erwähnen, dass mit der Begrifflichkeit MODEL die Komponente der Architektur zu verstehen ist. Die Modelle wie die der Batteriesysteme und PV-Wechselrichter sind neben weiteren Funktionen lediglich ein Bestandteil der MODEL Komponente.

Die Darstellungskomponente (VIEW) ist für die Darstellung der Daten aus der MODEL Komponente zuständig. Sie dient als ausgebende Schnittstelle zum User. Handelt es sich um eine Software mit grafischen Eingabemethoden, informiert sie zudem die control Komponente über die Eingaben des Users. Die Form der Darstellungen ist nicht klar definiert. Eine Ausgabe der Daten kann beispielsweise als Text in der Konsole oder durch Grafiken erfolgen. Ebenso ist sie dazu in der Lage Daten in Dateien zu speichern. Die Information über die Darstellungsform und der darzustellenden Daten aus der MODEL Komponente erhält sie durch die CONTROL Komponente.

Die Steuerungskomponente (CONTROL) interpretiert die Eingaben des Users. Sie vermittelt zwischen dem MODEL und der VIEW Komponente. Sie versorgt die MODEL Komponente mit allen notwendigen Daten. Überdies ermittelt die CONTROL Komponente Änderungen im Status der MODEL Komponente und übergibt diese Informationen der VIEW Komponente. Dazu stellt sie Methoden bereit, über welche die VIEW Komponente die benötigten Daten aus der MODEL Komponente abruft. Darüber hinaus gibt die CONTROL Komponente der VIEW Komponente vor, in welcher Form sie die Daten der MODEL Komponente darstellt.

4 Entwicklung von openBatLib

Dieses Kapitel dokumentiert die Entwicklung von openBatLib. Das erste Unterkapitel beschreibt das entwickelte Konzept von openBatLib. Das darauf folgende Unterkapitel beschreibt die Erweiterung der Funktionen, die gemäß der Anforderung aus Kapitel 3.1 nötig waren. Der im Rahmen dieser Arbeit erstellte Quellcode wird durch ein GitHub Repository zur Verfügung gestellt [16].

4.1 Konzeptentwicklung nach der MVC-Architektur

Um die Entwicklung von openBatLib zu erleichtern, ist ein Konzept entwickelt worden. Das Konzept beschreibt den Aufbau der Programmstruktur gemäß den Anforderungen. Wie in den Lösungsansätzen beschrieben, ist die serielle Programmstruktur von PerMod in eine modulare Programmstruktur überführt worden. Eine systematische Übersicht der Programmstruktur von openBatLib ist in Abbildung 4 dargestellt.

Die MODEL Komponente beinhaltet die Funktionen zum Berechnen der Simulationsergebnisse. Hier finden sich die Modelle aus PerMod wieder. Die Modelle der Batteriesysteme und der PV-Wechselrichter sind in eigenständige Funktionen aufgeteilt. Im Speziellen bilden die drei Batteriesystemtypen (AC-, DC- und PV-Kopplung) jeweils eigene Funktionen. Dies ermöglicht es, die Modelle der Batteriesysteme unabhängig von einem PV-Wechselrichter zu verwenden. Des Weiteren bietet die MODEL Komponente eine Schnittstelle zu realen PV-Batteriespeichersystemen. Sie ermöglicht eine Steuerung und einen Datenaustausch zwischen openBatLib und realer PV-Batteriespeichersysteme. Über diese Schnittstelle kann der User Vorgabewerte an das reale System übergeben und verschiedene Statuswerte wie den Ladezustand der Batterie abfragen.

Um dem User die Simulationsergebnisse oder ausgelesene Werte aus realen Systemen zugänglich zu machen, enthält die VIEW Komponente Funktionen zum Darstellen der Ergebnisse und Messwerte. Für diese Komponente ist die Entwicklung von Ausgabefunktionen erforderlich, da PerMod keine Methoden zur grafischen Darstellung bietet. Die Funktionen der VIEW Komponente ermöglichen verschie-

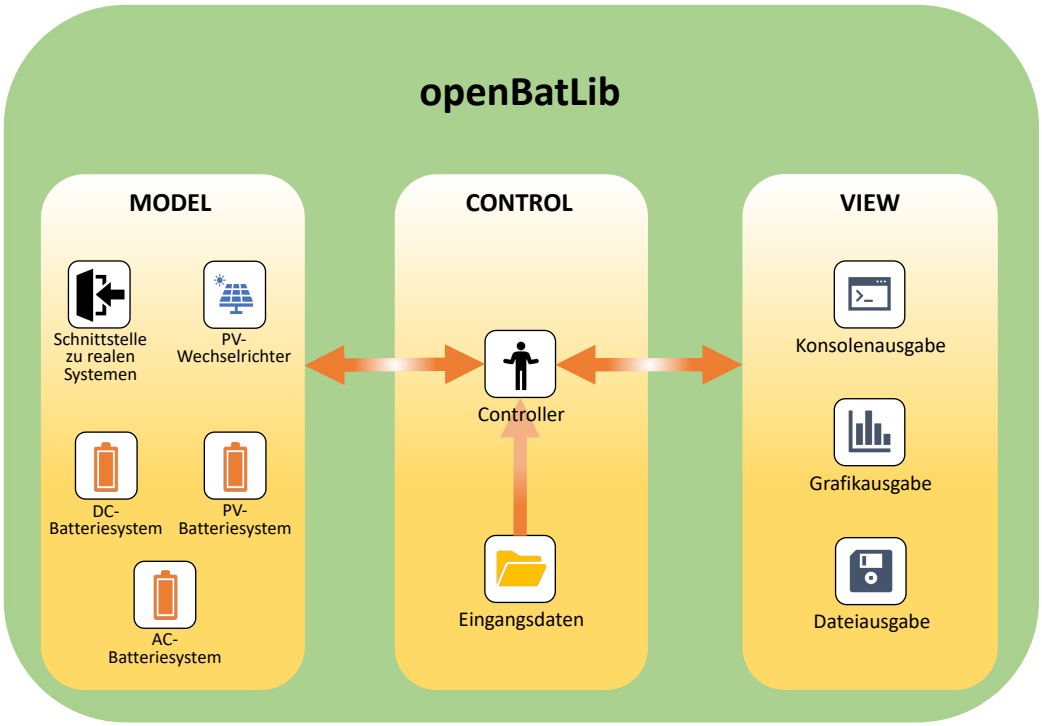

Abbildung 4: Systematische Übersicht des Konzepts nach der MVC Architektur von openBatLib.

dene Ausgabeformen der Simulations- und Messergebnisse. Eine Ausgabe der Daten kann durch die Konsole oder das Erstellen von Grafiken erfolgen. Es besteht zudem die Möglichkeit Grafiken in einer Datei zu sichern. Die VIEW Komponente kann die Ergebnisse in Textform als auch in maschinenverständlicher Form in eine Ausgabedatei sichern. So lassen sich für spätere Untersuchungen und Analysen die Simulationsergebnisse oder Messwerte erneut heranziehen.

Die CONTROL Komponente enthält Funktionen zur Steuerung des Programmablaufs. Sie entscheiden nach der Interpretation der Usereingaben und Eingangsdaten, welche Funktionen aus der MODEL Komponente benötigt werden. Diesbezüglich lädt sie die benötigten Eingangsdaten wie die Parameter des zu simulierenden PV-Batteriespeichersystems, sowie Erzeuger- und Lastprofile. Diese übergibt sie an die Funktionen der PV-Wechselrichter und Batteriesysteme der MODEL Komponente, siehe Abbildung 4. Soll ein reales PV-Batteriespeichersystem angesprochen werden, übergibt sie die benötigten Informationen wie IP-Adressen und Sollwert-

vorgaben an die Schnittstellenfunktion der MODEL Komponente zu realen Systemen. Sind die Simulationen und Messungen abgeschlossen, teilt die CONTROL Komponente der VIEW Komponente mit, welche Daten sie darstellen soll und gibt die Form der Ausgabe vor. Für diesen Zugriff stellt die CONTROL Komponente der VIEW Komponente eine entsprechende Schnittstelle zur Verfügung.

4.2 Erweiterung und Übersetzung der PerMod Modelle

Wie sich nach dem Vergleich von PerMod und den Anforderungen aus Kapitel 3.1 zeigte, können die Modelle aus PerMod nicht ohne Erweiterungen übernommen werden. Dieser Unterabschnitt beschreibt daher die Erweiterungen der Modelle von openBatLib gegenüber der ursprünglichen PerMod Modelle. Die umgesetzten Erweiterungen finden sich in sehr ähnlichen Ausführung in allen drei Batteriesystemmodellen wieder. Aus diesem Grund fasst dieser Abschnitt sie daher zusammen. Exemplarisch beschreibt es die Erweiterung der Funktionen des AC-gekoppelten Batteriespeichermodells.

Die Modelle der Batteriespeichersysteme und der PV-Wechselrichter von PerMod simulieren mit einer fixen Simulationsschrittweite von $\Delta t = 1\,\mathrm{s}$. Die Anforderungen an openBatLib sehen jedoch Simulationen mit Simulationsschrittweiten von $\Delta t \geq 1\,\mathrm{s}$ vor. Dementsprechend ist die Funktion zur Berücksichtigung der Einschwingzeit um Δt erweitert worden, siehe Gleichung (23). An dieser Stelle ist es wichtig zu erwähnen, dass durch den User die Simulationsschrittweite Δt in Sekunden angegeben wird.

$$P_{\mathrm{BS},\,\tau} = P_{\mathrm{BS}}(t - \Delta t) + (P_{\mathrm{BS},\,\mathrm{lim}} - P_{\mathrm{BS}}(t - \Delta t)) \cdot 1 - e^{-\frac{\Delta t}{\tau}} \tag{23}$$

Ebenfalls ist die Bestimmung des Energiegehalts E_{B} der Batterie nach Gleichung (24) um Δt erweitert worden.

$$E_{\mathrm{B}} = \begin{cases} E_{\mathrm{B},\,0} + \dfrac{P_{\mathrm{Bat}} \cdot \Delta t \cdot \sqrt{\eta_{\mathrm{Bat}}}}{3600} & \text{für } P_{\mathrm{Bat}} > 0 \\ E_{\mathrm{B},\,0} + \dfrac{P_{\mathrm{Bat}} \cdot \Delta t}{3600 \cdot \sqrt{\eta_{\mathrm{Bat}}}} & \text{für } P_{\mathrm{Bat}} < 0 \\ E_{\mathrm{B},\,0} & \text{sonst} \end{cases} \tag{24}$$

Durch eine größere Simulationsschrittweite Δt wird in einem Simulationszeitschritt eine größere Energie erzeugt. So kann die Batterie in einem Simulationsschritt über ihre eigentliche Kapazität be- oder entladen werden. Um ein Über- oder Unterladen der Batterie zu verhindern, berechnet das Modell des Batteriesystems die für den jeweiligen Zeitschritt erzeugte Energie $E_{\mathrm{B,\,est}}$ in kWh nach Gleichung (25).

$$E_{\mathrm{B,\,est}} = \frac{P_{\mathrm{BS}} \cdot \Delta t}{3600} \tag{25}$$

Von der ermittelten Energie $E_{\mathrm{B,\,est}}$ ausgehend, prüft das Modell, ob die Batterie mit dieser Energie im Rahmen ihrer nutzbaren Kapazität E_{Bat} und des aktuellen Energiegehalts $E_{\mathrm{B},\,0}$ geladen oder entladen werden kann. Darauf aufbauend berechnet es die Leistung $P_{\mathrm{BS,\,est}}$ nach Gleichung (26).

$$P_{\mathrm{BS,\,est}} = \begin{cases} \dfrac{(E_{\mathrm{Bat}} - E_{\mathrm{B},\,0})}{\Delta t} \cdot 3600 & \text{für } E_{\mathrm{BS,\,est}} > 0 \wedge E_{\mathrm{BS,\,est}} > (E_{\mathrm{Bat}} - E_{\mathrm{B},\,0}) \\ \dfrac{E_{\mathrm{B},\,0}}{\Delta t} \cdot (1 - k_{\mathrm{korr}}) \cdot 3600 & \text{für } E_{\mathrm{BS,\,est}} < 0 \wedge |E_{\mathrm{BS,\,est}}| > E_{\mathrm{B},\,0} \end{cases} \tag{26}$$

Eine schematische Übersicht der Erweiterungen findet in Abbildung 5 auf Seite 30. Im Gegensatz zum ursprünglichen Simulationsablauf, findet sich hier zusätzlich die Begrenzung auf die Nennkapazität der Batterie wieder.

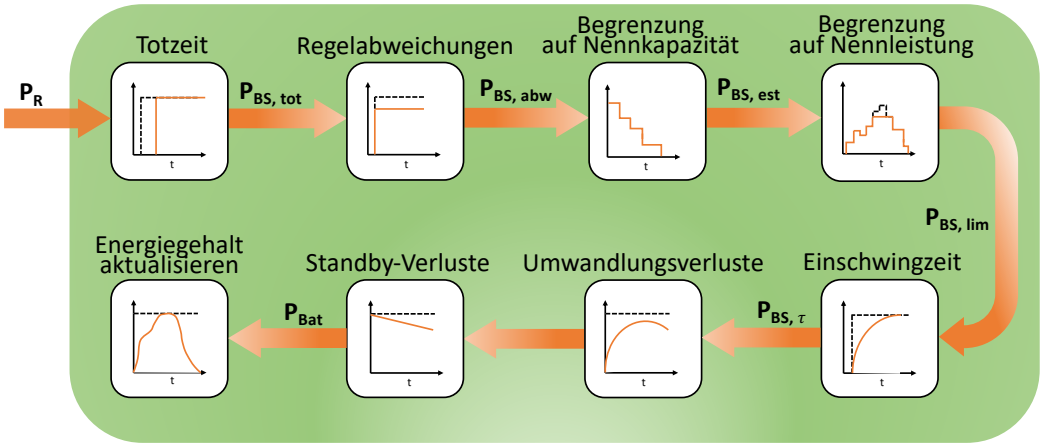

Abbildung 5: Erweiterter Ablauf des AC-gekoppelten Batteriemodells.

Der Einfluss der Tot- und Einschwingzeiten nimmt bei größeren Simulationsschrittweiten deutlich ab. Sie zeigen bei einem Δt in der Größenordnung von mehreren Minuten kaum noch einen Einfluss, auf die in diesem Simulationszeitschritt erzeugte Energie [9]. Ist die Totzeit der Regelung t_{tot} größer als die Summe aus Δt und $3\,\tau$ ($t_{\mathrm{tot}} > (3\,\tau + \Delta t)$), vernachlässigt das Modell daher die Totzeit. Um die Einschwingzeit der Regelung zu berücksichtigen, benötigt das Modell den vorausgegangenen Leistungswert $P_{\mathrm{BS}}(t - \Delta t)$ durch eine Angabe des Users. Ist Δt größer als die Summe aus der Totzeit t_{tot} und $3\,\tau$, vernachlässigt das Modell hingegen die Einschwingzeit.

Zeitgleich zu der Umsetzung der Erweiterung fand eine Übersetzung des Quellcodes von PerMod nach Python statt. Der Quellcode von openBatLib ist in der Python Version 3 verfasst. Im Übersetzungsprozess mussten verschiedene Eigenheiten von MATLAB beachtet werden. Ab der Version 6.5 verwendet MATLAB einen Just-In-Time (JIT) Compiler. Dieser Compiler beschleunigt automatisch das Ausführen von bestimmten Programmteilen, insbesondere Schleifen. Dieser Punkt ist insofern wichtig, da die Modelle der Batteriesysteme in ihrem Ablauf nach einer Schleife ausgerichtet sind. Ein JIT Compiler konvertiert den Quellcode in native Maschinenbefehle [17]. Python bietet in der Version 3 keinen eigenen JIT-Compiler oder eine vergleichbare Eigenschaft. Neben weiteren Partnern hat das Anaconda Projekt einen quelloffenen JIT-Compiler unter dem Namen Numba entwickelt [18].

Funktionen, deren Ablauf durch den Numba Compiler beschleunigt werden konnten, wurden dahingehend angepasst, dass sie zu diesem Compiler kompatibel sind.

4.3 Schnittstelle zu realen Systemen

Um Simulationsergebnisse mit Messungen eines realen PV-Batteriespeichersystems zu vergleichen, oder openBatLib als digitalen Zwilling eines realen Systems einzusetzen, bedarf es hierzu eine Schnittstelle zu einem realen System. Die Entwicklung dieser Schnittstelle ist das Thema dieses Unterkapitels.

Im Labor für regenerative Energien der Hochschule Emden-Leer ist das Batteriespeichersystem Plenticore BI 5.5 des Herstellers Kostal in Kombination mit dem Batteriesystem Battery-Box H6.4 vom Hersteller BYD installiert. Das Plenticore BI 5.5 bietet die Möglichkeit einer Steuerung und einen Datenaustausch über das ModBus/TCP Protokoll. Zum Verbindungsaufbau von openBatLib mit dem realen System, benötigt die entsprechende Funktion verschiedene Parameter wie die IP-Adresse und die Unit-ID des Systems. Diese Angaben stammen aus User-Eingaben. Mit diesen Parametern ist die Funktion dazu der Lage eine Verbindung über das ModBus/TCP Protkoll zum Plenticore BI 5.5 aufzubauen. Über diese Verbindung können Sollwertvorgaben und Daten wie der aktuelle Ladezustand der Batterie abgefragt werden. In einem sekündlichen Intervall schreibt die Funktion über die Schnittstelle Vorgabewerte $P_{\mathrm{PS,\,soll}}$ in das entsprechende ModBus/TCP Register. Der Batteriekonverter des Systems stellt daraufhin die eine AC-Leistung ein. Im selben Intervall fragt die Funktion verschiedene Parameter des Batteriespeichersystems ab. Neben der durch den Batteriekonverter eingestellten AC-Leistung ($\mathrm{P_{BS,ist}}$) und der DC-Leistung ($\mathrm{P_{Bat}}$) gehören dazu noch der Ladezustand der Batterie.

Durch den modularen Aufbau von openBatLib lässt sich die Schnittstelle zu realen Systemen um weitere Funktionen und Protokolle nachträglich erweitern. Auch Lese- und Schreibintervalle lassen sich nachträglich anpassen.

Die VIEW Komponente bietet zudem Funktionen zur Ausgabe und Sicherung der geschriebenen und gelesenen Größen des realen Systems. Sie können in einem sekündlichen Intervall in einer Log-Datei gesichert werden.

4.4 Wrapper-Klassen

Um durch openBatLib auch weiterhin Jahressimulation in einem Durchgang zu simulieren sind Wrapper-Klassen entwickelt worden. Die Wrapper-Klassen übernehmen die Steuerung der Simulationsabläufe auf der MODEL-Ebene. Sie rufen automatisiert alle nötigen Funktionen auf und sichern zur Laufzeit die Simulationsergebnisse. Dies ermöglicht eine komfortablere Nutzung von openBatLib. So sind Simulationen eines PV-Batteriespeichersystems mit einem einzigen Funktionsaufruf möglich. Da weniger Kontextwechsel zwischen den Komponenten nötig sind, beschleunigt dies die Simulation von Zeitreihen mit Leistungswerten für ein Jahr. Zur Simulation eines PV-Batteriespeichersystems stellt der User die Parameter des Systems, sowie das PV-Erzeugerprofil und Lastprofil zur Verfügung. Die zu PerMod bereitgestellten Referenzfälle sind ebenfalls in openBatLib hinterlegt. Es besteht somit zusätzlich die Möglichkeit, Erzeuger- und Lastprofile aus den Referenzfällen zu wählen. Weiterhin hat der User die Option eine Betriebsstrategie wie die Maximierung des Eigenverbrauchs zu wählen.

Aus den eingelesenen Parametern erkennt die CONTROL Komponente die Topologieform des Systems. Soll das PV-Batteriespeichersystem unter einer bestimmten Einsatzstrategie simuliert werden, übergibt die CONTROL Komponente die benötigten Eingangsdaten an diese Funktion der MODEL Komponente. Danach übergibt die CONTROL Komponente die eingelesenen Eingangsdaten der zuständigen Wrapper-Klasse der MODEL Komponente. Dies steuert den Simulationsablauf der Batteriesystemmodelle und berechnet die umgesetzten Energien für den simulierten Zeitraum. Um die Simulationsergebnisse wie die unterschiedlichen Leistungsverläufe oder der Ladezustände der Batterie auszugeben, bietet sie der VIEW Komponente Schnittstellen zum Zugriff auf diese Daten. Der User kann sich daraufhin die Leistungsverläufe oder die umgesetzten Energien über die VIEW Komponente ausgeben lassen. Wie zuvor bereits beschrieben, biete die VIEW Komponente mehrere Möglichkeiten der Ausgabe. Es lassen sich Textausgaben durch die Konsole oder Grafiken erzeugen. Weiterhin besteht die Option die Simulationsergebnisse in einer Ausgabedatei zu sichern.

5 Evaluierung von openBatLib

Durch die Erweiterung der Modelle von openBatLib besitzen sie im Vergleich zu PerMod veränderte Eigenschaften. Um den Einfluss der Umstrukturierung der Programmstruktur und der Erweiterungen zu untersuchen, findet in diesem Kapitel die Evaluierung von openBatLib statt.

Zunächst werden die Simulationsergebnisse von openBatLib und PerMod verglichen. Zeitgleich erfolgt eine Analyse des Simulationsverhaltens von openBatLib bei zunehmenden Simulationsschrittweiten Δt. Nach dem Vergleich der Simulationsergebnisse wird die Performance von openBatLib und PerMod verglichen. Es soll untersucht werden, wie groß der Einfluss einer Umstrukturierung der Programmsturktur und der Erweiterungen der Modelle ist. Diese beinhaltet einen Vergleich der Rechenzeit und des benötigten Arbeitsspeichers beider Tools.

Zum Abschluss der Evaluierung von openBatLib wird über die implementierte Schnittstelle ein reales Batteriespeichersystem angesteuert. Daher simuliert openBatLib im Vorfeld zunächst dieses Batteriespeichersystem über den Simulationszeitraum von einer Woche mit einer Auflösung von $\Delta t = 1\,\mathrm{s}$. Die gleichen Eingangsdaten, die zur Simulation verwendet wurden, stellen die Vorgabewerte für das reale Batteriespeichersystem.

5.1 Vergleich zu PerMod

Neben dem Vergleich der Simulationsergebnisse von openBatLib und PerMod untersucht dieses Unterkapitel die Performance beider Tools.

5.1.1 Vergleich der Simulationsergebnisse

Die Untersuchung der Simulationsergebnisse findet unter definierten Voraussetzungen statt. Alle PV-Batteriespeichermodelle von openBatLib sowie PerMod verwenden den ersten Referenzfall der HTW Berlin. Er repräsentiert einen Haushalt mit einem jährlichen Bedarf von $E_a = 5010\,\mathrm{kW\,h/a}$ und einer PV-Nennleistung von $P_{\mathrm{PV,\,nenn}} = 5\,\mathrm{kWp}$. Die verwendeten Erzeuger- und Lastprofile bestehen aus Datenreihen für ein Jahr in einer sekündlichen Auflösung. Das AC-gekoppelten Batterie-

speichersystem verwendet die Parameter des Siemens Junelight Smart Battery 9,9. Die DC-gekoppelten Batteriespeichersystemmodelle verwenden die Parameter des Plenticore plus 5,5 in Kombination mit der BYD Battery-Box H6,4. Für ein PV-gekoppeltes Batteriesystem lagen im Bearbeitungszeitraum dieser Arbeit keine Parameter vor. Daher greifen die Modelle der PV-gekoppelten Batteriespeichersysteme auf ein fiktives Dummy-System zurück.

Der Vergleich der Simulationsergebnisse erfolgt in Form der umgesetzten Energiesummen über den Simulationszeitraum. Die Tabelle 5 zeigt die umgesetzten Energiesummen und Abweichungen der AC-gekoppelten Batteriespeichermodelle. An dieser Stelle ist es wichtig darauf hinzuweisen, dass die Erläuterungen der einzelnen Formelzeichen zu den Energiesummen in der Nomenklatur aufgelistet sind.

Tabelle 5: Energiesummen und Abweichungen des AC-gekoppelten Systems in kWh.

		openBatLib AC		
	PerMod	$\Delta t = 1\,\mathrm{s}$	$\Delta t = 60\,\mathrm{s}$	$\Delta t = 900\,\mathrm{s}$
$\mathbf{E_L}$	5023,27	0,00	0,00	0,00
$\mathbf{E_{PV}}$	5221,95	-0,01	+10,87	+40,30
$\mathbf{E_{Bat,\ in}}$	1887,63	+4,72	+13,23	+13,97
$\mathbf{E_{Bat,\ out}}$	1828,48	+4,57	+12,82	+13,53
$\mathbf{E_{AC2G}}$	1568,05	-0,05	-64,09	-43,43
$\mathbf{E_{G2AC}}$	1939,68	+0,66	-59,15	-64,19
$\mathbf{E_{G2L}}$	1883,08	-4,18	-70,04	-159,00
$\mathbf{E_{Peri}}$	13,26	0,00	-0,00	-0,01
$\mathbf{E_{CT}}$	39,56	+0,01	-6,67	-28,44
$\mathbf{E_{PVS}}$	5031,88	-0,01	+10,96	+41,58
$\mathbf{E_{AC2BS}}$	2051,64	+4,99	+31,22	+37,16
$\mathbf{E_{BS2AC}}$	1671,40	+4,29	+15,32	+16,32
$\mathbf{E_{PVS2L}}$	1502,43	0,00	+21,08	+109,02
$\mathbf{E_{PVS2BS}}$	1995,04	+0,15	+20,32	-57,66
$\mathbf{E_{G2BS}}$	56,60	+4,84	+10,89	+94,82
$\mathbf{E_{PVS2G}}$	1534,40	-0,15	-30,44	-9,78
$\mathbf{E_{BS2L}}$	1637,75	+4,18	+48,97	+49,97
$\mathbf{E_{BS2G}}$	33,65	+0,11	-33,65	-33,65

In der ersten Spalte sind die durch PerMod berechneten Energiesummen in kWh aufgelistet. Die übrigen Spalten zeigen die Differenzen der Energiesummen von PerMod zu openBatLib in kWh pro Jahr. Die Größe Δt repräsentiert die jeweilige Simulationsschrittweite, mit der openBatLib simuliert hat. PerMod rechnet nach wie vor mit einer Simulationsschrittweite von $\Delta t = 1\,\mathrm{s}$.

Die erweiterten Modelle von openBatLib zeigen bei einer Simulationsschrittweite Δt von einer Sekunde nur leichte Abweichungen zu PerMod. Bei Betrachtung der Simulationsschrittweiten von $\Delta t > 1\,\mathrm{s}$ zeigen sich die auffälligsten Abweichungen. Die zur Bedarfsdeckung aus dem Netz bezogene Energie E_{G2L} nimmt bei steigenden Δt stark ab. Dies deutet auf einen höheren Direktverbrauch der erzeugten PV-Energie E_{PV} und der durch den PV-Wechselrichter umgesetzten Energie E_{PVS} hin. Der höhere Direktverbrauch der PV-Energie E_{PV2L} bestätigt diese Vermutung. Dies weist darauf hin, dass die Modelle bei gröberen Auflösungen einen erhöhten Direktverbrauch berechnen. Folglich sinken die in das Netz eingespeiste Energie E_{AC2G} und die aus dem Netz bezogene Energie E_{G2AC}. Vergleicht man die PV-Ladeenergie der Batterie E_{PV2BS} und Ladeenergie der Batterie aus dem Netz E_{G2BS} zeigt sich, dass sich die Batterie bei $\Delta t = 900\,\mathrm{s}$ überwiegend aus dem Netz lädt. Dennoch erhöht sich der Beitrag zur Lastdeckung durch die Batterie mit E_{BS2L}. Der Energiedurchsatz des Batteriesystems $E_{\mathrm{Bat,\,in}}$ und $E_{\mathrm{Bat,\,out}}$ steigt indes nur leicht.

Auch das openBatLib Modell des DC-gekoppelten Batteriespeichersystems weist Abweichungen auf. Jedoch sind sie insgesamt bei einer Simulationsschrittweite von $\Delta t = 1\,\mathrm{s}$ weniger stark ausgeprägt, siehe Tabelle 6.

Analog zu der Simulation des AC-gekoppelten Systems steigt die erzeugte PV-Energie E_{PV} mit steigender Simulationsschrittweite. Ebenso lässt sich eine erhöhte Bedarfsdeckung durch das PV-Batteriesystem durch E_{PVBS2L} verzeichnen. Zudem sinken die bezogene Energien E_{G2AC} und die zur Lastdeckung bezogene Energie E_{G2L}. Dies ist ebenfalls auf einen erhöhten Direktverbrauch der erzeugten PV-Energie zurückzuführen. Einen deutlichen Einfluss zeigt die Simulationsschrittweite bei dem Energiedurchsatz $E_{\mathrm{Bat,\,in}}$ und $E_{\mathrm{Bat,\,out}}$ der Batterie. Bei steigendem Δt sinken die Lade- und Entladeenergie der Batterie.

Tabelle 6: Energiesummen und Abweichungen des DC-gekoppelten Systems in kWh.

		openBatLib DC		
	PerMod	$\Delta t = 1\,\mathrm{s}$	$\Delta t = 60\,\mathrm{s}$	$\Delta t = 900\,\mathrm{s}$
$\mathbf{E_L}$	5023,58	0,00	0,00	0,00
$\mathbf{E_{PV}}$	5217,17	0,00	+9,07	+41,79
$\mathbf{E_{Bat,\ in}}$	1627,48	-0,05	-12,49	-91,00
$\mathbf{E_{Bat,\ out}}$	1543,30	-0,05	-11,82	-86,23
$\mathbf{E_{AC2G}}$	1877,39	+0,07	-26,67	-2,78
$\mathbf{E_{G2AC}}$	2160,94	+0,07	-36,52	-47,90
$\mathbf{E_{G2L}}$	2148,48	+0,07	-36,38	-46,60
$\mathbf{E_{Peri}}$	13,58	0,00	0,00	0,00
$\mathbf{E_{CT}}$	54,51	0,00	-8,47	-39,93
$\mathbf{E_{G2PVBS}}$	12,46	0,00	-0,14	-1,30
$\mathbf{E_{AC2PVBS}}$	12,46	0,00	-0,14	-1,30
$\mathbf{E_{PVBS2AC}}$	4752,49	0,00	+9,71	+43,83
$\mathbf{E_{PVBS2L}}$	2875,10	-0,07	+36,38	+46,60

Sehr ähnlich zu dem DC-gekoppeltem System verhält sich das PV-gekoppelte Batteriespeichersystem, siehe Tabelle 7. Auch hier lässt sich eine erhöhte Bedarfsdeckung durch das PV-Batteriesystem durch E_{PVBS2L} erkennen.

Die Simulationsschrittweite hat somit einen wesentlichen Einfluss auf die Simulationsergebnisse und deren Detaillierungsgrad. Grundsätzlich stellt bei gröberen Auflösungen ein erhöhter Direktverbrauch der erzeugten PV-Energie ein. Zu diesem Zusammenhang sind bereits mehrere Nachweise erbracht worden [9].

Tabelle 7: Energiesummen und Abweichungen des PV-gekoppelten Systems in kWh.

		openBatLib PV		
	PerMod	$\Delta t = 1\,\mathrm{s}$	$\Delta t = 60\,\mathrm{s}$	$\Delta t = 900\,\mathrm{s}$
$\mathbf{E_L}$	5023,58	0,00	0,00	0,00
$\mathbf{E_{PV}}$	5216,85	+0,21	+13,35	+44,05
$\mathbf{E_{Bat,\ in}}$	1657,98	+3,24	-9,75	-81,85
$\mathbf{E_{Bat,\ out}}$	1572,32	+3,03	-9,36	-77,90
$\mathbf{E_{AC2G}}$	1828,22	-0,21	-53,50	-24,55
$\mathbf{E_{G2AC}}$	2216,26	-2,91	-37,72	-50,26
$\mathbf{E_{G2L}}$	2213,38	-1,78	-36,94	-48,80
$\mathbf{E_{Peri}}$	13,58	0,00	0,00	0,00
$\mathbf{E_{CT}}$	49,00	-0,17	-10,50	-36,45
$\mathbf{E_{G2PVBS}}$	2,89	-1,13	-0,78	-1,46
$\mathbf{E_{AC2PVBS}}$	2,89	-1,13	-0,78	-1,46
$\mathbf{E_{PVBS2AC}}$	4638,42	+1,58	-16,56	+24,25
$\mathbf{E_{PVBS2L}}$	2810,20	+1,78	+36,94	+48,80

Zur Veranschaulichung der Leistungsflüsse zeigt die Abbildung 6 exemplarisch einen Tagesverlauf für den 29. Juni 2016 des AC-gekoppelten Systems. Sie zeigt die Leistungsflüsse in unterschiedlichen Simulationsschrittweiten. Von $\Delta t = 1\,\mathrm{s}$ aufsteigend bis zu einem Δt von einer Stunde.

Bei der Auflösung von $\Delta t = 1\,\mathrm{s}$ zeigen sich stark fluktuierende Werte der PV-Leistung. Ein solcher Verlauf ist typisch für einen Tag mit wechselnder Bewölkung. Die Residualleistung weist indes vereinzelt kurzfristige Lastspitzen auf. Vor allem in den Morgen- und Abendstunden. Bei einer Vergrößerung der Zeitschrittweite Δt nimmt die Höhe der Lastspitzen deutlich ab. Zusätzlich stellt sich eine Glättung des Verlaufs der Residualleistung ein. Ab einem Δt von fünfzehn Minuten ist dies deutlich zu erkennen.

Der Verlauf der Batterieleistung bei $\Delta t = 1\,\mathrm{s}$ zeigt um etwa sieben Uhr einen Entladevorgang. Dieser lässt sich ab einer viertelstündigen Auflösung nicht mehr beobachten. An dem dargestellten Tag erreicht das Batteriesystem ab einer Auflösung von einer Minute weniger häufig die maximale Ladeleistung des Batteriekonverters von 3,4 kW. Eine Begrenzung durch die maximale Entladeleistung von

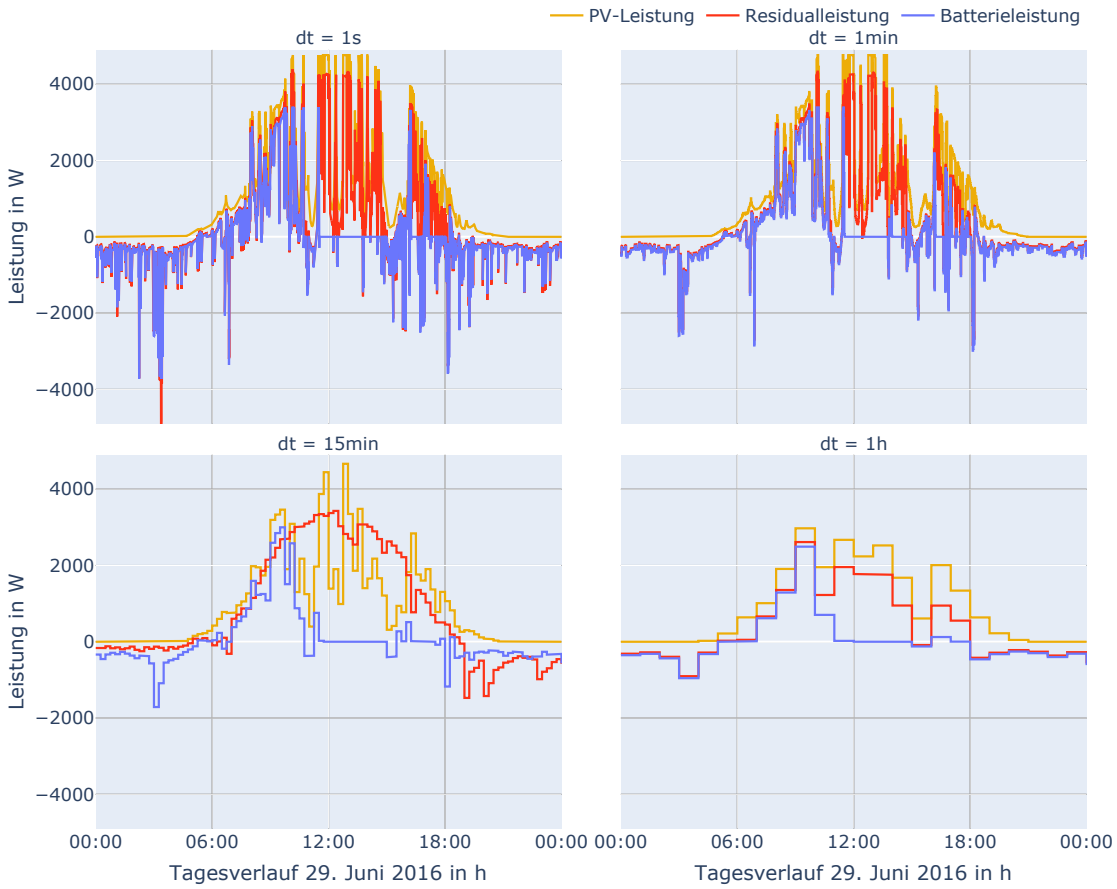

Abbildung 6: Simulierte Leistungsflüsse des AC-gekoppelten Systems am 29. Juni 2016.

3,7 kW tritt gar nicht mehr auf. Besonders deutlich zeigt sich dies um ca. drei Uhr. Die auftretenden Lastspitzen kann der Batteriespeicher nur zum Teil decken. Durch das Mitteln flacht die Lastspitze jedoch stark ab, sie kann so vollständig durch die Batterie gedeckt werden.

Die Leistungsverläufe bei einer Auflösung von Δt gleich einer Stunde zeigen eine zunehmende Gleichzeitigkeit von Erzeugung und Verbrauch. Dies bestätigt eine Zunahme des Direktverbrauchs der erzeugten PV-Leistung.

Allgemein lässt sich ein wesentlicher Einfluss der zeitlichen Auflösung auf die Simulationsergebnisse feststellen. Kurzzeitige Fluktuationen in den Leistungsverläufen sind bei gröberen zeitlichen Auflösungen nicht mehr erfassbar. Ebenso un-

terschätzen die Simulationen infolge dessen die Zyklenzahl der Batterie. Da eine zyklische Alterung der Batterien durch die Modelle nicht abgebildet wird, ist dieser Einfluss jedoch zu vernachlässigen. Bei gröberer Auflösung spielen die Leistungsbegrenzungen des Batterieumrichters kaum noch eine Rolle. Sie werden zudem weniger häufig in ihrem nominalen Leistungsbereich betrieben. Der Wirkungsgrad von Wechselrichtern ist unter dem nominalen Leistungsbereich deutlich kleiner. Mit der Vergrößerung der Simulationsschrittweite von $\Delta t = 1\,\mathrm{s}$ zu einem Δt von einer Stunde verringert sich die Energieaufnahme der Batterie. Dies zeigt sich vor allem in den Ergebnissen der DC- und PV-gekoppelten Batteriesysteme. Bei zeitlichen Auflösungen der Simulationsschrittweite ab fünfzehn Minuten wird der Energiedurchsatz der Batterie systematisch unterschätzt. Dies suggeriert einen niedrigeren Ausnutzungsgrad der Batteriekapazität. Zur korrekten Abbildung von kurzzeitigen Entladevorgängen und anschließenden Ladevorgängen ist daher eine hohe zeitliche Auflösung der Eingangszeitreihen entscheidend. Je höher die Auflösung der Eingangsdaten, desto genauer ist der Detaillierungsgrad der Simulationen.

5.1.2 Vergleich der Performance

Neben dem Vergleich der Simulationsergebnisse wird der Einfluss der Erweiterungen der Modelle und die Umstrukturierung der Programmstruktur auf die Performance von openBatLib untersucht. Hierzu wird die Rechenzeit und der Arbeitsspeicherbedarf beider Tools untersucht. Der zur Simulation verwendete Rechner ist mit einem 2 GHz Quad-Core Intel Core i5 Prozessor und einem 16 GB 3733 MHz LPDDR4X Arbeitsspeicher ausgestattet.

Für eine Integration der Funktionen und Modelle von openBatLib in externe Auslegungstools, kann die Arbeitsspeicherauslastung und die Rechenzeit einen begrenzenden Faktor auf die Gesamtperformance des so erweiterten Auslegungstools darstellen. Vor allem im Bereich von Web-Applikation kann eine hohe Rechenzeit einen begrenzenden Faktor der Bedienerfreundlichkeit darstellen. Das Messen der Rechenzeit bezieht sich auf die benötigte Zeit zum Ausführen der Simulation durch die Batteriespeichermodelle. Die gemessenen Zeiten repräsentieren eine Simulation für einen Simulationszeitraum von einem Jahr mit einer Auflösung von $\Delta t = 1\,\mathrm{s}$.

In der Tabelle 8 auf Seite 41 sind die gemessen Zeiten dargestellt.

Tabelle 8: Rechenzeiten der Batteriespeichermodelle in Sekunden bei einer Simulationsschrittweite von $\Delta t = 1\,\mathrm{s}$ für ein Jahr

	AC	DC	PV
PerMod	2.08	3.97	2.29
openBatLib	759.3	710.5	1025.7
openBatLib(JIT)	2.4	2.95	3.01
Differenz (PerMod-openBatLib(JIT))	0.32	-1.02	0.72

Bei Betrachtung der Tabelle 8 zeigt sich, dass die Rechenzeiten von openBatLib ohne den Einsatz des Numba JIT-Compiler um ein vielfaches länger sind im Vergleich zu PerMod. Unter der Verwendung dieses Compilers für die Batteriespeichermodelle von openBatLib haben sich die Rechenzeiten deutlich verkürzt. Die Modelle der AC- und PV-gekoppelten Batteriespeicher von openBatLib sind mit einem Unterschied der Rechenzeit von weniger als einer Sekunde nur unwesentlich langsamer. Im Fall des DC-gekoppelten Systems liefert openBatLib die Ergebnisse eine Sekunde früher als PerMod.

Der benötigte Arbeitsspeicher beträgt bei PerMod ca. 2,65 GB. openBatLib benötigt ca. 2,24 GB. Der höhere Speicherbedarf von PerMod liegt unter anderem darin begründet, dass es alle hinterlegten Parameter der PV-Batteriespeichersysteme in den Speicher lädt. An dieser Stelle ist openBatLib hingegen ist so ausgelegt, dass sie nur die Parameter des aktuell zu simulierenden Systems lädt.

5.2 Vergleich zum realen Speichersystem

Den Abschluss der Validierung von openBatLib bildet der Vergleich der Simulationsergebnisse mit Messungen eines realen Batteriespeichersystems. Das untersuchte Batteriespeichersystem ist das Plenticore BI 5,5 des Herstellers Kostal in Kombination mit der Battery-Box H6.4 vom Hersteller BYD. Es handelt sich bei diesem Produkt um ein AC-gekoppeltes Batteriespeichersystem. Zur Simulation verwendet das Modell die nach dem Effizienzleitfaden ermittelten Parameter des realen Systems.

Die Simulation sowie die Vorgabewerte des Batteriespeichersystems beziehen sich auf den ersten Referenzfall der HTW-Berlin. Für den Validierungslauf fiel die Wahl des Last- und Erzeugerprofils auf die Daten der Woche vom 18.07.2016 bis zum 24.07.2016. Dieser Zeitraum enthält sowohl klare als auch bewölkte Tage. Somit lassen sich unterschiedliche Wetterbedingungen abdecken. Die Auflösung der Datenreihen beträgt weiterhin eine Sekunde. Eine Übersicht der Erzeugerleistung des PV-Generators mit einer Nennleistung $P_{\mathrm{PV,\,nenn}}$ von 5 kWp und des Lastprofils für diesen Zeitraum findet sich in Abbildung 7.

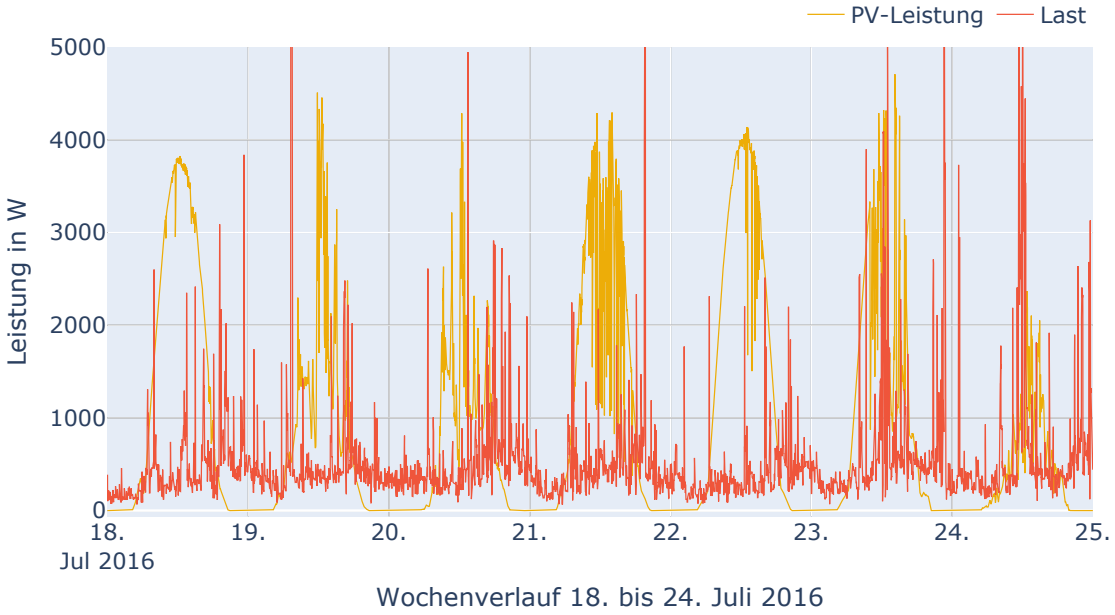

Abbildung 7: Wochenverlauf des Erzeuger- und Lastprofils.

Der Plenticore BI 5,5 ist nicht mit einem PV-Wechselrichter verbunden. Daher stammt die PV-Ausgangsleistung des PV-Wechselrichters aus dem PV-Wechselrichtermodell von openBatLib. Der simulierte PV-Wechselrichter ist der Sunny Boy 5,5 des Herstellers SMA. Zusammen mit dem Lastprofil und einer Maximierung des Eigenverbrauchs durch openBatLib ergibt sich eine Residualleistung P_{R}. Die Residualleistung P_{R} stellt den AC-seitigen Vorgabewert des realen Batteriespeichersystems dar. Das Steuern des Batteriespeichersystems findet durch openBatLib statt. Über eine Schnittstelle ermöglicht der Plenticore BI 5,5 einen Verbindungsaufbau und Datenaustausch über das ModBus/TCP Protokoll.

Während des einwöchigen Testlaufs schreibt openBatLib sekündlich die Vorgabewerte $P_{\mathrm{R}}(t) = P_{\mathrm{soll}}(t)$ in das dafür vorgesehene Register des Batteriespeichersystems. Im gleichen Intervall wird die durch den Batteriekonverter eingestellte AC-Leistung $P_{\mathrm{ist}}(t) = P_{\mathrm{PBS}}(t)$ und die DC-Batterieladeleistung P_{Bat} ausgelesen. Neben den Leistungen loggt openBatLib zusätzlich den aktuellen Ladezustand soc_0 der Batterie mit. An dieser Stelle gilt es zu beachten, dass zum Bearbeitungszeitpunkt dieser Arbeit keine Angaben zu der Genauigkeit der ausgelesen Werte vorliegen. Aus diesem Grund ist die Angabe einer Toleranz der Messwerte nicht möglich. Unter Anbetracht dessen, dass die Modelle neben diesem Vergleich zusätzlich gegen PerMod validiert werden, kann diese Messung dennoch zu der Validierung von openBatLib beitragen.

In der Auswertung der einwöchigen Messreihe zeigen sich Abweichungen zu den vorliegenden Parametern. Die Tot- und Einschwingzeit der Regelung des Plenticore BI 5,5 liegen laut Messungen nach dem Effizienzleitfaden bei $t_{\mathrm{tot}} = 0{,}63\,\mathrm{s}$ und $t_{\tau} = 2{,}79\,\mathrm{s}$. Die Messung im Rahmen dieser Arbeit hat eine Totzeit t_{tot} von circa 3 Sekunden und eine Einschwingzeit t_{τ} von circa 5 Sekunden ergeben. Diese Abweichungen sind auf veränderten Reglereigenschaften durch die externe Sollwertvorgabenregelung des Plenticore BI 5,5 zurückzuführen. Des Weiteren ist es in diesem Betriebsmodus nicht möglich, die Batterie auf 0 % zu entladen. Die fixe untere Grenze liegt bei 5 %. Ein weiterer abweichender Parameter gegenüber der Messung nach dem Effizienzleitfaden ist die nutzbare Kapazität der Batterie. Die Messungen ergeben im Mittel eine nutzbare Kapazität von 5,3 kWh. Sie ist damit um 0,4 kWh kleiner als in den vorliegenden Parametern. Die Simulationsparameter von openBatLib sind dahingehend angepasst worden.

Der Verlauf der Ladezustände der Batterie ist in der Abbildung 8 dargestellt. Eine Wochendauerlinie der Ladezustände findet sich in Abbildung 9. Sie gibt eine Übersicht, an wie vielen Minuten der Woche welche Ladezustände vorgeherrscht haben.

Die Ladezustände der angepassten Simulation decken sich zu großen Teilen mit den Messwerten. Am 22. und 23. Juli lässt sich erkennen, dass die Batterie in der Simulation sich nicht auf das gleiche Niveau entlädt. Das Modell scheint die Entlade-

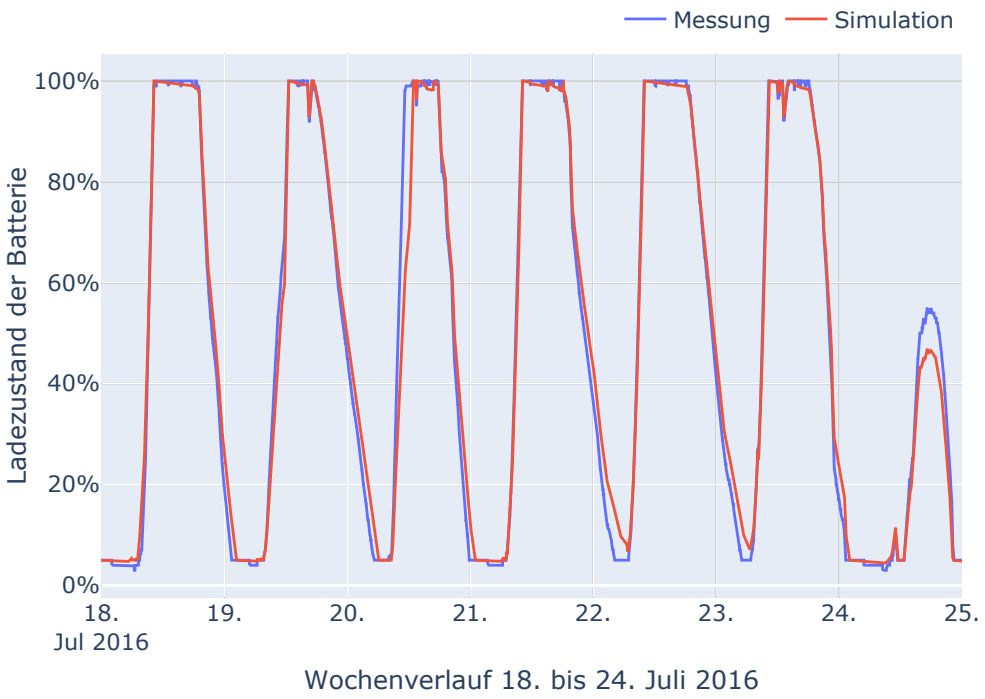

Abbildung 8: Wochenverlauf des Ladezustands der Batterie

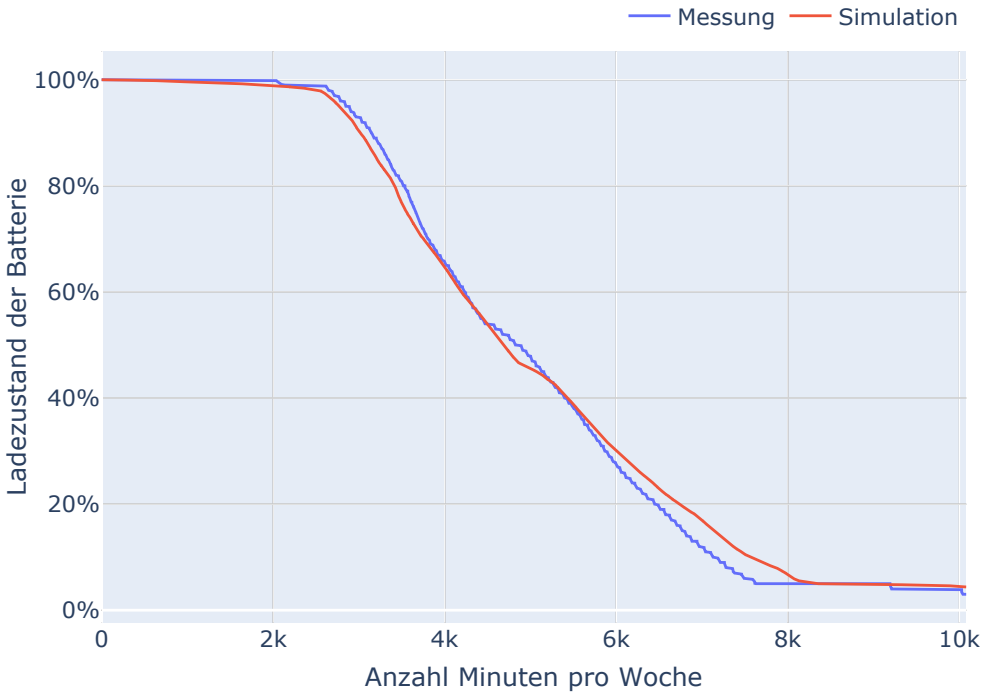

Abbildung 9: Wochendauerlinie der Ladezustände der Batterie.

effizienz der Batterie zu unterschätzen. Dies spiegelt sich in den absoluten Zahlen der Wirkungsgrade wider. Die Abbildung 10 zeigt die gemessenen und simulierten Wirkungsgrade des Plenticore BI 5,5. Der Wirkungsgrad Entladen (BAT2AC) der Batterie berechnet sich nach der Gleichung 28. Die Abweichungen der Entladeeffizienz der Messung zu der angepassten Simulation betragen +3,8 %. Zieht man die umgesetzte DC-Entlademenge aus Abbildung 11 hinzu, ergibt sich für diese Woche eine Differenz von +0,66 kWh. Ein weiterer Faktor in diesem Zusammenhang ist die Überschätzung des Wirkungsgrads der Batterie. Er ist als Speicherung in der Abbildung 10 aufgeführt. Der Wirkungsgrad der Batterie η_{Bat} ergibt sich nach Gleichung 29. Hier zeigt sich eine Differenz von +3,43 %. Dies führt zusätzlich zur Verschlechterung der gesamten Entladeeffizienz.

Die Ladeeffizienz wird im Vergleich zur Messung indes nur leicht unterschätzt. Eine Unterschätzung der Ladeeffizienz lässt sich an den schneller steigenden Flanken der Messung in Abbildung 8 erkennen. In ausgeprägter Form lässt sich dieser Effekt am letzten Tag des Validierungslaufs ablesen. An diesem Tag lädt sich die Batterie der Simulation gegenüber der Messung auf ein niedrigeres Niveau auf. Der Wirkungsgrad zum Laden der Batterie weicht um -1,49 % ab, siehe Abbildung 10. Bei der geladenen Energiemenge zeigt sich eine Differenz zwischen Messung und Simulation der geladenen DC-Energie in dieser Woche beträgt +0,63 kWh.

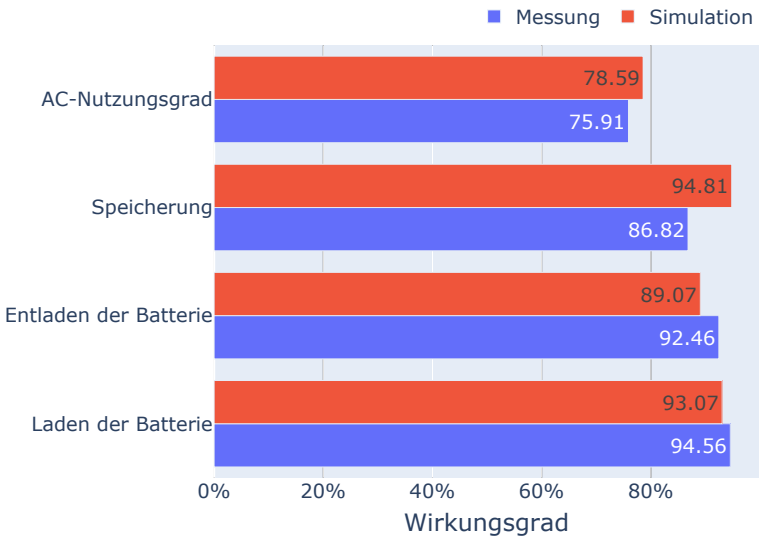

Abbildung 10: Ermittelte Wirkungsgrade des Batteriesystems.

Eine weitere Vergleichsgröße ist der AC-Ausnutzungsgrad η_{AC}. Er berechnet sich nach Gleichung 30. Die Simulationsergebnisse weichen hier um +2,68 % von der Messung ab.

$$\eta_{\mathrm{AC2BAT}} = \frac{\int_a^b P_{\mathrm{Bat\ laden}}\, dt}{\int_a^b P_{\mathrm{BS\ entladen}}\, dt} \tag{27}$$

$$\eta_{\mathrm{BAT2AC}} = \frac{\int_a^b P_{\mathrm{BS\ entladen}}\, dt}{\int_a^b P_{\mathrm{Bat\ laden}}\, dt} \tag{28}$$

$$\eta_{\mathrm{Bat}} = \frac{\int_a^b P_{\mathrm{Bat\ entladen}}\, dt}{\int_a^b P_{\mathrm{Bat\ laden}}\, dt} \tag{29}$$

$$\eta_{\mathrm{AC}} = \frac{\int_a^b P_{\mathrm{BS\ entladen}}\, dt}{\int_a^b P_{\mathrm{BS\ laden}}\, dt} \tag{30}$$

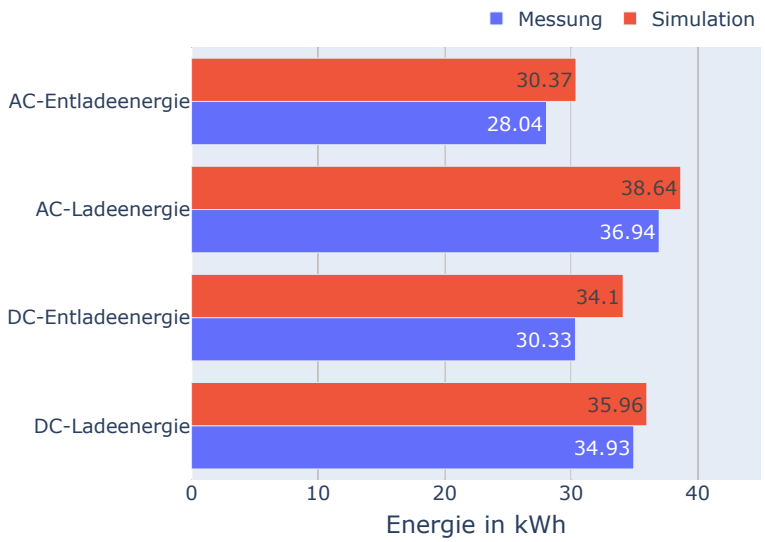

Abbildung 11: Umgesetzte Energiemengen einer Woche.

Vergleicht man die Wirkungsgrade in Abhängigkeit zu der Last zeigt sich, dass das Modell die Wirkungsgrade vor allem im niedrigen Leistungsbereich von 0 W bis 1 kW unterschätzt, siehe Abbildung 12 und 13. Dies gilt sowohl für die Lade-, als auch Entladeeffizienz. Im weiteren Verlauf der Kurven gleichen sich die Wirkungsgrade der Simulation und der Messung wieder an.

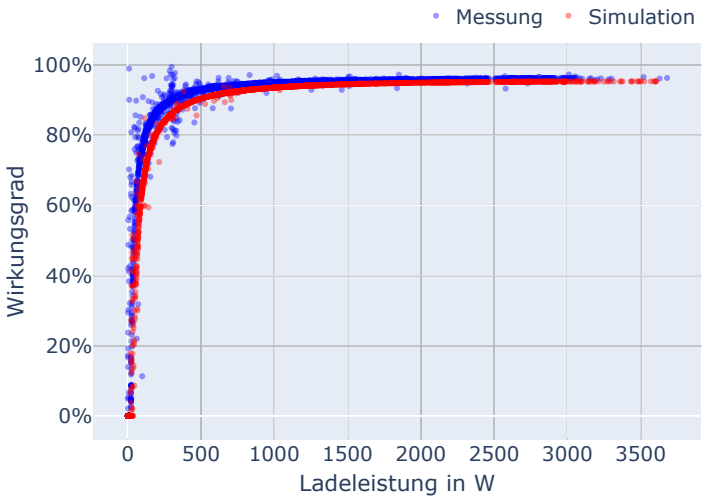

Abbildung 12: Darstellung der lastabhängigen Ladeeffizienz.

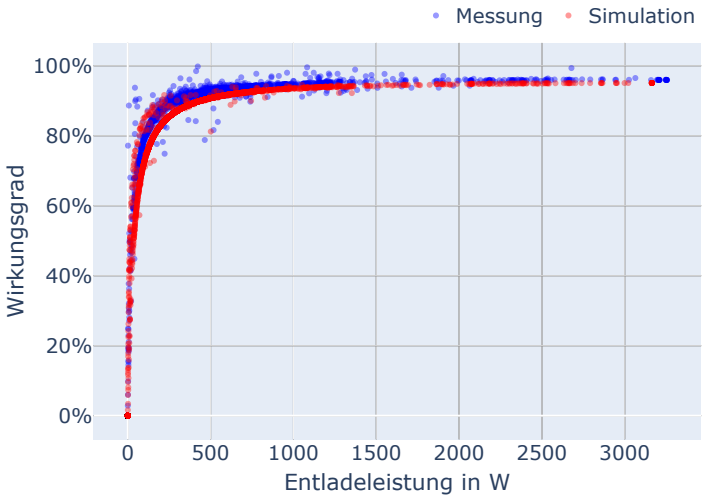

Abbildung 13: Darstellung der lastabhängigen Entladeeffizienz.

6 Zusammenfassung

Dieser Abschnitt hat die Aufgabe den Verlauf der Arbeit in einem Kapitel zusammenzufassen. Gefolgt von einem abschließendem Fazit und einem Ausblick über den Rahmen dieser Arbeit hinaus.

Aus der einleitenden Motivation dieser Arbeit ging die Aufgabe hervor eine quelloffene Funktionsbibliothek zum Simulieren von PV-Batteriespeichern zu entwickeln. Die Basis hierfür stellten das quelloffene MATLAB-Tool PerMod und die veröffentlichten Erzeuger- und Lastprofile der HTW Berlin. Ziel war es die Funktionen und Modelle von PerMod in die MVC-Architektur zu überführen. MATLAB ist eine proprietäre Software des MathWorks Unternehmens. Dies gewährleistet keine vollkommen quelloffene Nutzung der Modelle. Aus diesem Grund fand eine Übersetzung in die Programmiersprache Python der Version 3 statt.

Den Rahmen der Entwicklung bildete die Definition verschiedener Kriterien und Anforderungen an openBatLib. Sie gaben die Grundlage einer Konzeptentwicklung von openBatLib. Aufbauend auf diesem Konzept basiert die Weiterentwicklung der Modelle und Funktion von PerMod.

Nach der Entwicklung von openBatLib folgte eine Bewertung der erweiterten Modelle von openBatLib. Bei der Validierung sind sie zunächst mit den ursprünglichen Modellen aus PerMod verglichen worden. Des Weiteren folgte ein Vergleich zu einem realen Batteriesystem. Das Batteriespeichermodell von openBatLib verwendete zur Simulation die nach dem Leitfaden erhobenen Parameter dieses Speichers. Die Messung des Speichersystems erstreckte sich über den Zeitraum von einer Woche. Die Simulation und das Speichersystem verwendeten dazu identische Eingangsdaten aus Erzeugung und Last.

Ein Performancevergleich in Form von Rechenzeiten und Speicherverbrauch im Arbeitsspeicher des verwendeten Rechners zeigte, dass eine erhöhte Modularität keine Verschlechterung der Performance zeigte.

Eine Auswertung der durch die Simulation umgesetzten Energiesummen brachte mehrere Erkenntnisse. Die erweiterten Modelle besitzen die Eigenschaft gröber aufgelösten Eingangsdaten zu verarbeiten. Diese gewonnene Flexibilität geht auf

Kosten der Genauigkeit und Detaillierung der Simulationsergebnisse. Zwischen dem Detaillierungsgrad und den zeitlichen Auflösungen ließ sich ein Zusammenhang beobachten. Vereinfacht lässt sich zusammenfassen, dass gröbere Auflösungen der Eingangsdaten ungenauere Ergebnisse und Beobachtungen liefert.

Verglichen mit den erhobenen Messwerten des realen Batteriespeichersystems zeigten sich Abweichungen in den Berechnungen des Modells. Grund hierfür waren neben Beschränkung des Batteriesystems in Bezug auf das Betriebsverhalten, die nutzbare Kapazität der Batterie. Sie fiel im Vergleich zu den vorliegenden Parametern kleiner aus. Unter der Fernsteuerung durch openBatLib über die dafür vorgesehene Schnittstelle wich es vom Normalbetrieb ab. Dies äußerte sich neben einer maximalen unteren Entladegrenze von 5 % in einer erhöhten Tot- und Einschwingzeit der Regelung. Eine entsprechende Anpassung der Simulationsparameter resultierte in einer Verkleinerung der Abweichungen. Die Untersuchung der Ergebnisse brachte die Erkenntnis, dass das Modell die Entladeeffizienz und Batteriewirkungsgrad des Systems überschätzt.

6.1 Fazit

Im Rahmen dieser Arbeit ist eine Funktionsbibliothek mit dem Eigennamen openBatLib entwickelt worden, die sich in ihren Eigenschaften mit PerMod vergleichen lässt. Sie enthält neben erweiterten PV-Batteriespeichermodellen zusätzliche Funktionalitäten. So lassen sich die Komponenten fortan als eigenständige Einheiten nutzen. Es besteht ein direkter Zugriff auf die Modelle der PV-Wechselrichter und Batteriespeichermodelle oder den Parametern der hinterlegten Systeme. Auch kann der User auf Betriebsstrategien zugreifen und PV-Wechselrichter einzeln simulieren. openBatLib bietet unterschiedliche Wege, um auf Berechnungsergebnisse zuzugreifen. Es lassen sich Werte in Form von Textausgaben oder Grafiken anzeigen und Dateien sichern.

Nach der Validierung stellt sich heraus, dass openBatLib in Bezug auf die Performance sich mindestens gleichwertig zu PerMod verhält. Dies spiegelt sich in einem leicht verringerten Arbeitsspeicherverbrauch und in ähnlichen Rechenzeiten wider. Werden bei der Auslegung eines Speichers gröber aufgelöste Eingangs-

daten verwendet, verlieren die Simulationen an Genauigkeit und Detaillierungsgrad. Kurzzeitige Fluktuationen in den Leistungsverläufen sind nicht mehr erfassbar. Leistungsspitzen flachen ab. Leistungsbegrenzungen der Systeme verlieren so an Bedeutung. Diese Faktoren können in einer suboptimalen Auslegung eines Speichersystems resultieren. Die so erstellten Berechnungen geben einen zu hohen Eigenverbrauch der PV-Energie vor. Aufgrund dessen kann die Wahl auf einen zu kleinen Speicher fallen.

Während der Fernsteuerung wies das Batteriesystems abweichende Eigenschaften zu den vorliegenden Parametern auf. Eine Reproduktion der Eigenschaften, unter denen die Parameter nach dem Leitfaden erhoben wurden, war nicht möglich. Der Erkenntnisgewinn aus diesem Vergleich findet unter Vorbehalt statt. Ebenso blieb die Frage offen, wie das Batteriesystem die Größen in den Registern generiert. Nach Anpassung der Simulationsparameter ließ sich das System unter den veränderten Bedingungen jedoch genauer nachbilden. Als Fazit dieses Vergleichs lässt sich festhalten, dass das AC-gekoppelte Batteriemodell Wirkungsgrade unterschätzt. Insbesondere gilt dies für das Entladen und den Wirkungsgrad der Batterie.

6.2 Ausblick

Dieser Abschnitt widmet sich einem Ausblick. Es wird erörtert, unter welchen Umständen die gewonnenen Erkenntnisse weiter genutzt werden können und welche Potenziale zum Verbessern der Modelle und deren Abläufe bestehen.

Bei einem Aufruf der Modelle in Einzelschritten lässt sich nach dem aktuellen Stand die Totzeit nicht berücksichtigen. Dies ist dem Aufbau der Modelle nach einer Schleife geschuldet. Damit eine Totzeit in die Berechnung mit einfließen kann, könnte das Modell mit einer Methode zum Messen der Zeit und einem Gedächtnis ausgestattet werden. Bei dem ersten Aufruf würde es die aktuelle Zeit t_0 des Host-Systems auslesen. Bei jedem weiteren Aufruf misst es erneut die Zeit und vergleicht sie mit t_0. Ist die Differenz $t_{\mathrm{dif}} \geq t_{\mathrm{tot}}$, wird der aktuelle Leistungswert umgesetzt. Je nach Δt der Zeitreihe und dem Intervall des Aufrufs wird die Leistung anteilig umgesetzt.

Eine weitere Untersuchung kann die Funktion zur Abschätzung der Lade-, beziehungsweise der Entladeenergie verbessern. Dies könnte die Abweichungen der erweiterten Modelle minimieren.

Zum Zeitpunkt der Erstellung dieser Arbeit verfügt openBatLib über keine Funktionen zur Berücksichtigung von Zellalterungen der Batterie. Eine Möglichkeit eine gealterte Batterie nachzubilden bestünde darin, die Parameter eines alternden Systems in gewissen Abständen nach dem Leitfaden zu ermitteln. Diese dienen dann als Simulationsparameter. Eine weitere Methode wäre die Annahme einer pauschalen Verringerung der Kapazität und Ladeleistung beruhend auf Mess- und Erfahrungswerten.

Eine deutliche Beschleunigung der Simulation kann durch eine Parallelisierung der Berechnungen erreicht werden. Zusätzliche Geschwindigkeit lässt sich durch eine stärkere Vektorisierung des Ablaufs gewinnen. Dazu werden Teile der Schleife durch Vektorfunktionen übernommen.

Python Programmierung hält zunehmend Einzug in die Entwicklung von Web-Tools. Beispiele hierfür sind unter anderen die Frameworks Flask [13] und Django [14]. Es besteht somit die grundsätzliche Möglichkeit openBatLib beziehungsweise Teile dieses Tools in bestehende Web-Anwendung zu integrieren.

Literaturverzeichnis

[1] M. Sterner und I. Stadler, Hrsgg., *Energiespeicher - Bedarf, Technologien, Integration*, 2. Aufl. Wiesbaden: Springer Vieweg, 2017, OCLC: 932683817.

[2] Bundesnetzagentur, „Regelungen zu Stromspeichern im deutschen Strommarkt," Bundesnetzagentur, Tech. Rep., 2020.

[3] BWS, „Statistische Zahlen der deutschen Solarstrombranche (Speicher/Mobilität)," in *Statistische Zahlen der deutschen Solarstrombranche (Speicher/-Mobilität)*, 2020.

[4] „DKE/AK 371.0.9 Kennwerte von stationären Batteriespeichern." [Online]. Verfügbar: https://www.dke.de/de/ueber-uns/dke-organisation-auftrag/dke-fachbereiche/dke-gremium?id=3007512&type=dke%7Cgremium

[5] V. Quaschning, N. Orth, S. Maier, und N. Böhme, „Bewertung und Optimierung der Energieeffizienz von Photovoltaik-Batteriesystemen (EffiBat)," HTW Berlin, Berlin, Tech. Rep., Juli 2020.

[6] J. Figgener, D. Haberschusz, K.-P. Kairies, O. Wessels, S. Zurmühlen, D. U. Sauer, und P. Woerner, „Speichermonitoring BW Jahresbericht," Institut für Stromrichtertechnik und Elektrische Antriebe RWTH Aachen, Aachen, Tech. Rep., 2019.

[7] J. Weniger, T. Tjaden, N. Orth, und S. Maier, „Documentation Performance Simulation Model for PV-Battery Systems (PerMod) Version 2.1," Apr. 2020.

[8] BVES und BWS, „Effizienzleitfaden für PV-Speichersysteme," in *Effizienzleitfaden für PV-Speichersysteme*, Juli 2019.

[9] J. Weniger, „Bewertung der Energieeffizienz von netzgekoppelten Photovoltaik-Batteriesystemen in Wohngebäuden," Dissertation, TU Berlin, Berlin, Apr. 2020.

[10] „The MIT License." [Online]. Verfügbar: https://opensource.org/licenses/MIT

[11] A. Klein, „Experimentelle Untersuchung von Batteriesystemen im simulierten niedrigen Erdorbit," Dissertation, Universität Stuttgart, Stuttgart, Juli 2017.

[12] C. Schäfer, *Schnellstart Python: ein Einstieg ins Programmieren für MINT-Studierende.* Springer-Verlag, 2019.

[13] „Flask." [Online]. Verfügbar: https://palletsprojects.com/p/flask/

[14] „Django." [Online]. Verfügbar: https://www.djangoproject.com/

[15] R. Steyer, *Webanwendungen mit ASP.NET MVC und RAZOR: Ein kompakter und praxisnaher Einstieg.* Morgan Kaufmann, 2017.

[16] „openBatLib." [Online]. Verfügbar: https://github.com/fastrockstar/openBatLib

[17] W. D. Pietruszka, *MATLAB und Simulink in der Ingenieurpraxis: Modellbildung, Berechnung und Simulation*, 4. Aufl., Serie Lehrbuch. Wiesbaden: Springer Vieweg, 2014.

[18] „Numba." [Online]. Verfügbar: http://numba.pydata.org/