Implementation and Verification of a Realistic Battery Model in the DEMKit Simulation Software

Bart Homan, Victor M.J.J. Reijnders, Gerwin Hoogsteen, Johann L. Hurink, Gerard J.M. Smit
Department of EEMCS, University of Twente, Enschede, the Netherlands
Corresponding author: b.homan@utwente.nl

***Abstract*—In this work we consider the implementation of the DiBu-model, a realistic model for battery State of Charge prediction, in DEMKit, a smart grid energy management toolkit. The implementation is discussed in detail and the differences between the ideal battery model previously used, and the new realistic model are shown. Lastly the model and implementation are validated by comparing the State of Charge predicted using DEMKit to the State of Charge derived from measurements on an actual battery. The predicted battery behavior matches reality quite well, the difference between the predicted and measured state of charge is generally less than 2%.**

***Index Terms*—Storage, Predictive model, Smart grid, Energy management, State of Charge**

I. Introduction

Simulations of (smart) grids are used, for example, to predict and investigate the behavior of grids that have yet to be built [1], or to find weaknesses in existing grids. Grid simulation frameworks, such as TRIANA [2], GridSpice [3], and DEMKit [4] use models of the relevant devices, controllers and cables in a (smart) grid to perform these simulations, with and without smart grid control. The accuracy of (the results of) a simulation crucially depends on the accuracy of the models that are used in the simulation.

Batteries are an important part of smart grids. They are used, for instance, as a backup power source, the main power source for an electric car, or to (temporarily) store energy generated by e.g. PV-panels or wind turbines. Hence to accurately simulate a smart grid, accurate models for batteries, e.g. the KiBaM model [5], the Schiffer Model [6] or the Dualfoil model [7] can be considered as valid options. The current DEMKit environment, however, does not use these accurate models but make use of a too ideal battery model, e.g. the actual behavior and tolerances of batteries are not considered. In the simulations, performed using the DEMKit environment, hundreds or even thousands of households are considered. If an accurate but complex model, like one of the aforementioned models, is used in these simulations the computation power needed would be very large, and the computation time would be extremely long. Hence, a simple yet accurate model for the prediction of the battery *State of Charge* (SoC) is needed. The DiBu-model for battery State of Charge prediction, [8], [9] has been developed to allow to predict a more realistic behavior of a battery in simulation tools, while being simple. In this work[*], this model is integrated and implemented using the optimization and simulation software for energy management (DEMKit) and combined with the algorithm from [12].

[*]Note that comparisons between our work (models and simulator) and other possible solutions are beyond the scope of this work. These are covered in previous publications [4], [8], [9], [10], [11].

This work is part of the *GridFlex Heeten* project, which aims to realize a local energy market in which sharing and usage of local energy is stimulated [13]. A neighborhood of 50 households participates in this project, 24 of which will be equipped with a 5 kWh battery. These batteries are controlled in a way to relieve stress in the network and to use generated energy as local as possible. As batteries are an important ingredient in this project, it is important to have accurate and realistic models of batteries.

The main contributions of this paper are:

- Improved accuracy of simulations in DEMKit.
- Cyber-physical interaction between a battery model and the control of the battery.

In Section II the background of the DiBU-model (Section II-A) and of the DEMKit software (Section II-B) are discussed. The implementation of the DiBu-model in the DEMKit simulation environment is discussed in Section III. In Section IV the implementation of the model is tested in a simulation of (part of) the neighborhood in Heeten. Furthermore, in Subsection IV-C the validity of the model and its implementation are tested by comparing the predicted SoC and power of one of the simulated batteries, to the SoC and power calculated from measurements on the corresponding actual battery. In Section V the conclusions and possible future work are discussed.

II. Background

A. The DiBu-model

The Diffusion Buffer model for battery State of Charge prediction (*DiBu-model*), has been introduced in [8] and has been improved and further verified in [9]. The state of charge, expressed as a percentage of the maximum SoC, is predicted by predicting the battery voltage during a charge, discharge or idle step in a future time interval. More precisely, the state of charge (SoC_t), at a time (t) in the future is calculated using the following equation:

$$SoC_t = SoC_{t-1} + \frac{U_t \cdot I_t \cdot \Delta t}{E_{max}}, \qquad (1)$$

where SoC_{t-1} is the SoC at a time before t, (so this is either a starting condition or a previously estimated value for the SoC), Δt is the time difference between t and $t-1$ (typically in the range of 1 second to 1 minute), I_t is the current at time t, which is known because it is supplied to or demanded from the battery, E_{max} is the battery capacity given by the manufacturer and, lastly, U_t is the voltage at time t. To predict this voltage,

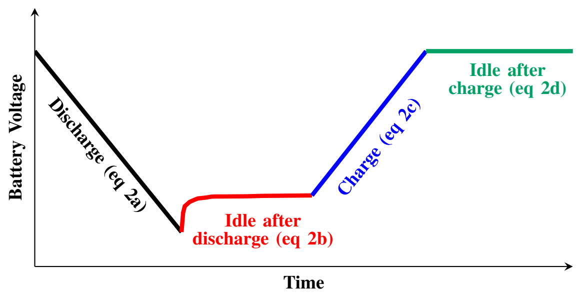

Fig. 1. Schematic representation of the voltage behavior over time during the various states of battery operation.

four states of battery operation are used in the model: charging, discharging, idle after charging and idle after discharging, see Fig. 1. The separation between idle periods after charging and discharging is to account for the *capacity recovery effect* and the *rate capacity effect*, which are explained in [9]. For each of the four states an equation for U_t is derived, see (2a) - (2d). These equations are used to model the battery voltage during the mentioned different states of operation; discharging (2a), idle periods after discharging (2b), charging (2c) and idle periods after charging (2d).

$$U_t = U_{t-1} + \frac{\alpha \cdot I_{t-1}}{SoC_{S0}} \tag{2a}$$

$$U_t = U_{t_0^*} + (U_{max} - U_{t_0^*}) \cdot \left(1 - e^{-\frac{t-t_0^*}{\beta \cdot (t-t_0^*)+\gamma}}\right) \tag{2b}$$

$$U_t = U_{t-1} + \frac{I_{t-1}}{\delta} \tag{2c}$$

$$U_t = U_{t-1} \tag{2d}$$

In Equations (2a), (2c) and (2d), U_{t-1} is the voltage at a time $t-1$ (before t), I_{t-1} is the current at $t-1$ which is known, and SoC_{S0} is the state of charge of the battery at the start of discharging. In (2b), t_0^* is the time at which the idle step starts, $U_{t_0^*}$ is the voltage at the start of the idle step and U_{max} is the voltage at the start of the discharge step, directly before the idle step. The parameters α, β, γ, and δ are given constants and battery-specific. These parameters can be determined from measurements at the battery and this process has been thoroughly discussed in [8] and [9].

B. DEMKit overview

DEMKit (short for Decentralized Energy Management toolKit) is a smart grid optimization and discrete time simulation software developed at the University of Twente [4][†]. The simulation framework follows a cyber-physical systems approach, wherein the interaction between control (optimization methods) and the physical devices and grids can be analyzed. DEMKit provides a library with components that model devices and grid assets. Furthermore, components containing control and optimization algorithms are available. By combining these components, a complete model of a smart grid of many households, equipped with e.g. batteries and electric vehicles, can be created and its total energy profile can be controlled and optimized.

Within DEMKit, the device components model the behavior of a device, such as a battery or the stochastic behavior of a load. Each device determines, based on the current state, its power consumption in the next time interval. Furthermore, at the start of a time interval, each device has the option to change the internal state based on the power consumption of the last interval. For example, a battery can update its SoC according to the energy consumed/produced since the previous time interval. All components are configurable through parameters, such as e.g. the capacity of a battery or the load profile of a device, such as e.g. for a washing machine. A controller can be connected to a device component to influence its behavior. Herein, the device model is leading to ensure that the device is always operated in a valid state. The battery models currently implemented in DEMKit are generally ideal and loss free.

Controllers within DEMKit have two main purposes, namely offline optimization for e.g. a day-ahead market and performing control on runtime to resolve deviations and disturbances. For the offline optimization the iterative profile steering heuristic [14] is used. This algorithm schedules a cluster of devices for N future time intervals using a sliding window. Hereby, the heuristic makes use of predictions and the aforementioned device models. Within profile steering, each device controller m receives a steering signal $\vec{d} = [d_1, d_2, \ldots, d_N]^T$ containing the desired energy consumption for each of the N intervals. The device controllers respond with an optimized energy profile $\vec{x}_m = [x_{m,1}, x_{m,2}, \ldots, x_{m,N}]^T$. An efficient algorithm to schedule batteries under these steering signals is presented by Van der Klauw et al. in [12]. For more background on the profile steering algorithm, we refer the reader to [14] and [4].

While running, the device controller tries to realize the energy consumption/production planned for each time interval as specified in the offline optimization. Disturbances in predictions or model inaccuracies often result in situations where the scheduled energy consumption cannot be followed exactly. Hence, a strategy to minimize deviations from the schedule is required. To this end, the asynchronous event-driven profile steering mechanism is developed [11], which leverages the predictive nature of profile steering to balance short-term disturbances and long-term flexibility of devices. In this event-driven method, only one device is rescheduled at a time when necessary, e.g. when a large deviation occurs. This reduces the complexity and required computation time, such that a new schedule is available within a few seconds. The same steering signals and device optimization algorithms are used within event-driven profile steering.

III. Implementation

To use the presented DiBu battery model (Subsection II-A) in a smart grid simulation and optimization framework, such as DEMKit, two steps need to be taken. The first step is to implement a discrete time model of the battery model, which is the behavioral description of the battery. And, secondly, an optimization and control strategy suitable for the battery model needs to be implemented. This section covers the implementation of these two aspects in the DEMKit framework.

[†]DEMKit is the successor of the TRIANA smart grid simulation software, also developed at the University of Twente. In recent publications "DEMKit" is used to refer to the simulation software, while in earlier publications "TRIANA" is used.

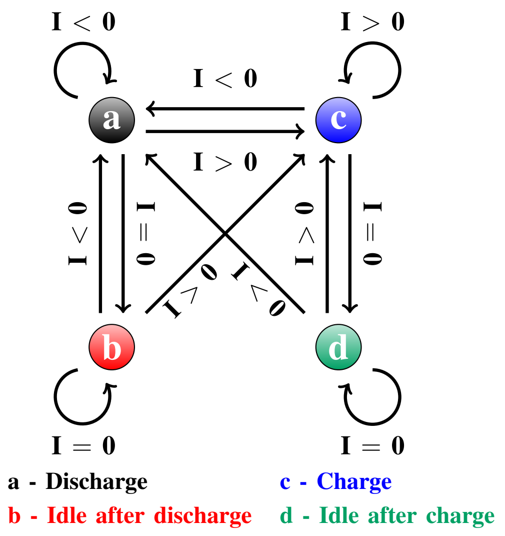

Fig. 2. Schematic representation of the battery state transitions.

A. Battery model

We model the battery by a state machine. Each state corresponds to a different state of operation (as described in the previous section). We determine the state transitions using the requested current from the battery and the previous state of the battery. In Fig. 2 the four states and the different state transitions are given. The equations corresponding to state x is given in Equation $(2x)$, for $x \in \{a, b, c, d\}$.

At each time step of the simulator, the voltage of the battery is updated based on the state of the battery. The following assumptions are made in this process. As DEMKit works with power, only the requested power from the battery is available, not the current. To overcome this, we use the calculated voltage of the battery to determine the charging/discharging current from teh requested power (though voltage measurements can be integrated easily).

At each time step the minimum and maximum possible charging powers are updated according to the voltage of the battery. If the power requested from the battery in the optimization is either below the minimum or above the maximum charging powers, we set the requested power to the minimum or maximum possible charging power. If the voltage of the battery reaches its maximum (6.9 volts for the battery used in the simulations), we assure that the battery is no longer charged, as this would ruin the battery. Also, if the battery is empty (so if the SoC falls below a certain threshold), we reset the voltage to its base value (5.5 volts for the battery used in the simulations). Doing this allows for a re-calibration of the model. At this point, we do not consider the battery losses during (dis)charging, or when storing energy over a longer period of time, beyond the losses that have already been considered in the DiBu-model itself.

B. Battery controller

The aforementioned battery scheduling algorithm [12] is not directly applicable to the improved, more realistic battery model, introduced in this work, as this method is not able to cope with the changing power constraints. With the realistic battery model, the battery voltage varies based on the state, and therefore the maximum power consumption/production changes over time. However, initial simulation studies show that the difference in SoC between the ideal and realistic battery model only drifts slightly over time. Therefore we continue using the scheduling algorithm from [12], that was based on an ideal battery model. However, we extend it with functionality to frequently retrieve the current SoC of the battery to track the mismatch between the current and the scheduled state. If this mismatch becomes to large, synchronization and rescheduling can be performed through signaling an event.

In order to signal events for rescheduling, we determine the the mismatch based on the SoC. For this, we use the planned battery power profile $\vec{x}_b = [x_{b,t+1}, x_{b,t+2}, \ldots, x_{b,t+N}]^T$, which is created at a time t, to determine the planned SoC in the future time intervals. Together with the initial SoC ($SoC_{b,t}$) at the time of the planning (t) the planned SoC ($SoC_{b,\tau}$) at time interval $\tau > t$ is then given by:

$$SoC_{b,\tau} = SoC_{b,t} + \sum_{i=t}^{\tau-1} \frac{x_{b,i+1}}{E_{max}}, \qquad (3)$$

When the current state of charge of the battery (SoC_b) differs too much from the scheduled SoC at a time interval τ, i.e. $SoC_b < SoC_{b,\tau} - d$ or $SoC_b > SoC_{b,\tau} + d$, an event is signaled to perform rescheduling. Hereby, parameter d can be chosen arbitrarily to define the maximum allowed mismatch. In our simulations, we use $d = 0.05$. Subsequently, the battery controller receives an updated desired profile $\vec{d}$ from its parent controller running profile steering. Furthermore, the controller synchronizes the maximum power consumption/production values with the battery model, which depend on the current battery voltage. Subsequently, the controller re-plans the operation of the battery based on the updated constraints and the new steering signal.

TABLE I
CHARACTERISTICS AND PARAMETERS OF THE CONRAD BATTERY USED FOR THE SIMULATIONS.

Conrad Battery	
Type	Lead-acid
Nominal voltage (V)	6,0
Rated capacity (Ah)	7,2
α (V/A)	3,030 x 10^{-5}
β (-)	0,268
γ (s)	2,684
δ (A/V)	2,438 x 10^{4}

IV. Results

To validate the implemented battery model and control strategy, simulations are used. For this, one of the (optimized) battery charging profiles from the simulation with DEMKit is programmed in the battery test equipment to compare and validate the simulated profile with measurements from a real battery.

A. Simulation setup

Simulations are performed within the DEMKit simulation toolkit. A small neighborhood, representing a part of the test area in Heeten, (10 households) was modeled using the artificial load profile generator presented in [10]. This neighborhood consists of 5 families with children, 4 young couples and 1 retired couple, all without flexible loads, such as electric vehicles or smart appliances. Each of the 10 households was assigned one battery with the specifications as presented in Table I. Furthermore, to test the energy sharing concept, 5 households got a PV setup, with the intention that a surplus of renewable energy generation of these PV installations is allowed to flow into all 10 batteries. Each household uses a Home Energy Management System (HEMS) running the profile steering algorithm to optimize the household power consumption. The load, PV and battery controllers communicate with the local HEMS. A cluster controller (also using profile steering) coordinates the energy profiles of each household to allow energy sharing among the households. The desired profile for the neighborhood is a zero-profile, $\vec{p} = [0, 0, \ldots, 0]^T$, which implies that the goal is to achieve a balanced cluster where the sum of power consumption/production among households is zero for each interval. Furthermore, to avoid extreme peaks if total balance is not possible, the optimization objective is to minimize the Euclidean distance between this desired profile and the aggregated profile of all devices ($\vec{x}$), i.e. to

$$\text{minimize } ||\vec{p} - \vec{x}||_2. \tag{4}$$

We refer the reader to [14] for more details on the optimization method and profiles.

The battery available for verification has a capacity of 43.2 Wh (6 V $\times$ 7.2 Ah), which is significantly lower then the batteries to be used in the Heeten project, which will have a capacity of 5 kWh. To take this difference into account, we scale down the household load by dividing the load profiles by 115.7 (43.2 Wh / 5 kWh).

Two simulations are run, one with the original ideal battery model for comparison and one with the realistic battery model presented in this paper. Each simulation considers one week (week 13 from the generated profile) in discrete time intervals of 1 minute. Optimization is performed in intervals of 15 minutes where a sliding window approach is used. Each day, the energy profile for two days into the future (192 intervals) is optimized to avoid boundary effects.

B. Simulation results

Firstly we evaluate the performance of the realistic model, by comparing the results of the two simulations. The two models result in slightly different optimization results, where the realistic case favors charging the battery a bit more on average, which results in a slightly higher average power consumption (P_{avg}) for the whole neighborhood of 10 houses and an higher average SoC (SoC_{avg}). In Table II an overview of the simulation results for the simulation with the ideal battery model (ideal) and the presented model (realistic) is given. Furthermore Fig. 3 shows that the SoC of the realistic battery model is generally higher than that of the ideal battery model, however both profiles follow the same daily pattern. In this figure we also observe that the SoC of the batteries gradually rise, due to a surplus of energy in the neighborhood.

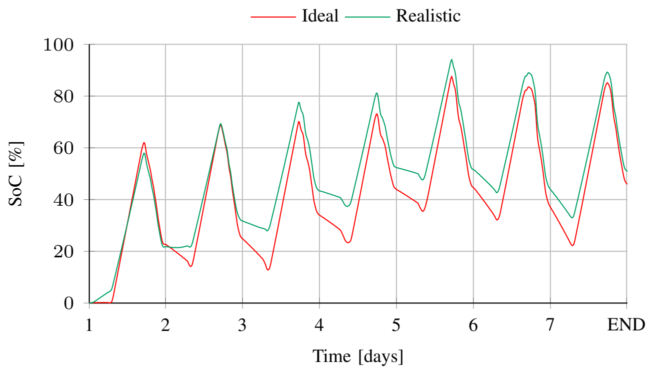

Fig. 3. Average state of charge of the batteries

TABLE II
NUMERICAL RESULTS OF THE SIMULATION CASE STUDIES

	P_{max}	P_{min}	P_{avg}	SoC_{avg}
Ideal	25.0 W	-127.5 W	-5.5 W	42.4 %
Realistic	27.3 W	-132.1 W	-5.3 W	49.2 %

Secondly, we evaluate the performance of the realistic battery model with the planning made by the optimization process. In Fig. 4 the planned power profile (pp$_{planned}$) and the realized power profile with the realistic (pp$_{realistic}$) battery model for the whole cluster are shown. Additionally, the power profile with the ideal battery model pp$_{ideal}$ is also shown for comparison. Larger deviations between pp$_{planned}$ and pp$_{realistic}$ are mainly observed in the first days of the simulation. Over the course of a couple of days, the mismatch is reduced significantly. Moreover, mismatches in the SoC do not affect the cluster performance as the capacity of the batteries is never completely utilized. Thus, despite these mismatches in SoC, the batteries always provide flexibility to resolve planning inaccuracies. As a result, the planning for the neighborhood as a whole can be followed accurately.

A comparison between the power values and the resulting SoC of the planning and the realistic battery model is given in Table III. For both characteristics the mean absolute error (MAE) and the root mean square error (RMSE) is given per day of the complete week. We observe that the error is smallest on the fifth day. Few event-driven replanning operations are required as the SoC of the realistic batterie models do not differ significantly from the planned SoC.

In Fig. 5 the power profile, averaged over all batteries, is shown. It is immediately clear that the realistic battery model is able to closely follow the planned power output. The power constraints of the realistic battery model depend on the SoC and voltage, which match the specification (Table I) at a SoC of 50%. The daily executed optimization process uses the power constraints as communicated by the realistic battery model when creating a new planning. Hence, we observe smaller errors with a balanced planned power profile created when the SoC is close to 50%.

This result can be explained because of the daily optimization performed at the start of each day.

TABLE III
MISMATCHES BETWEEN THE PLANNING AND THE REALISTIC BATTERY

Day	P_{MAE}	P_{RMSE}	SoC_{MAE}	SoC_{start}
1	5.61 W	6.48 W	0.93 %	0.0 %
2	2.80 W	4.09 W	0.95 %	21.8 %
3	2.14 W	3.20 W	0.97 %	31.6 %
4	1.00 W	1.46 W	0.77 %	43.3 %
5	0.62 W	1.02 W	0.87 %	52.3 %
6	0.92 W	1.42 W	0.91 %	51.4 %
7	1.08 W	1.41 W	1.04 %	45.6 %
All	2.03 W	3.30 W	0.92 %	-

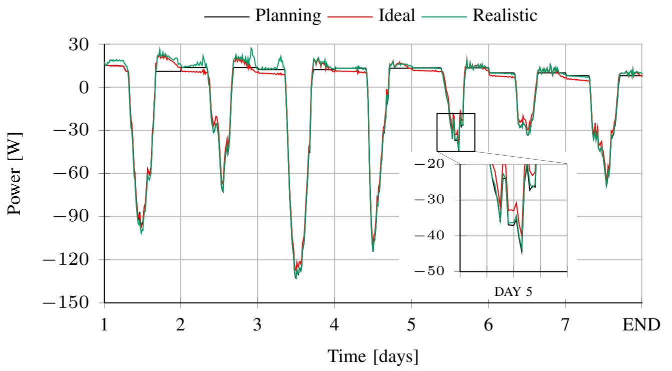

Fig. 4. Power profile of the neighborhood of 10 scaled households including the planning for the realistic battery model case.

C. Validation

Next, the performance of the implemented DiBu-model itself is evaluated. The simulation yields a realized power profile for the batteries (Fig. 5), which can be used to predict the SoC of the battery (Fig. 3). To validate these simulation results, a comparison was made between the predicted SoC and power, and the SoC and power derived from measurements on the actual battery. The battery, which specifications are given in Table I, was subjected to the desired power profile, and the voltages and currents were measured and used to derive the actual power profile and the actual SoC. Note that the *desired power profile* for the battery is the *realized power profile* from the simulation results.

All measurements are done using a *CADEX C8000* battery analyzer [15]. This device has to be programmed manually, therefore the power profile for only one day (day 5) was chosen for this validation. The desired power profile, that was obtained from the simulation in DEMKit and used in this validation is displayed in Fig. 6. However, to accommodate the sensitivity of the battery analyzer the programmed supplied and demanded currents were rounded to the nearest 10 mA. Moreover the sensitivity of the battery analyzer was such that charge or discharge currents below 50 mA could not be achieved accurately. To remedy this, currents of 25 mA and lower were set to 0 A and currents between 25 mA and 50 mA were set to 50 mA.

In Fig. 6 the desired power profile and measured power profile are displayed. Between time index 8:00 and 17:30 and between time index 19:00 and 8:00 the desired power profile and the measured power profile are nearly identical.

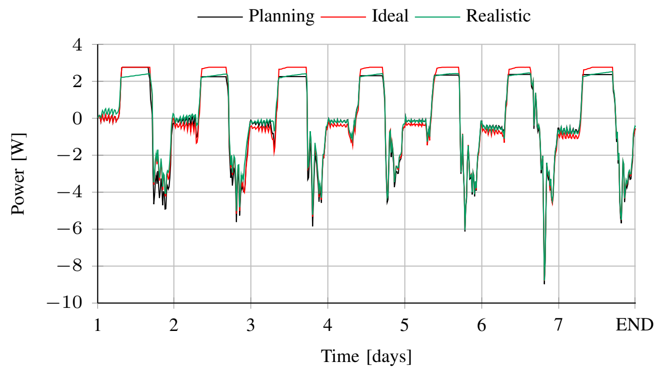

Fig. 5. Average power consumption and production of the batteries, including the planned power profile for the realistic battery model case.

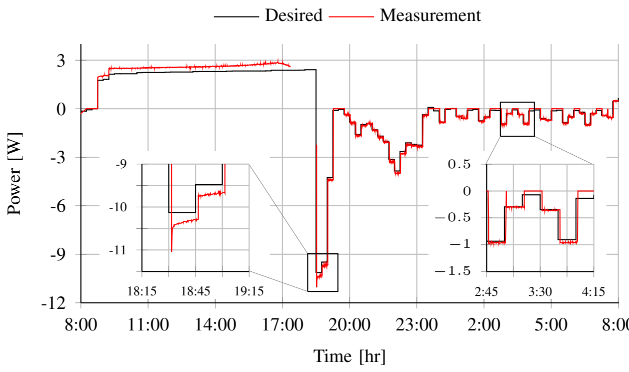

Fig. 6. Desired power profile of one battery in the neighborhood, compared to the power profile for the same battery, derived from the measurements.

The maximum difference between the desired power profile and the measured power profile is 0.46 W, at 17:00. After 19:00 the maximum difference is about 0.1 W. Between time index 17:30 and 19:00 there is no measured power, this is caused by a problem during the measurement. At the start of the indicated time interval the battery voltage was too high, this prompted the equipment to protect the battery by skipping to the next discharge step. This has been indicated in the graph, by a missing part of the line. It is likely that the high voltage is caused by degradation of the battery, as this battery has been used for many measurements.

The predicted SoC (SoC_{pred}) and the SoC calculated from the measurements (SoC_{meas}) are compared in Fig. 7. Apart from the interval between 17:30 and 19:00 the SoC_{meas} matches the SoC_{pred} very well. From 8:00 to 17:30 the difference between the SoC_{pred} and the SoC_{meas} is negligible. The largest deviation between the SoC_{pred} and the SoC_{meas} (6,6 %) is observed around 19:30. At this moment a power peak of 10 W is drawn from the battery. From 19:30 onward the difference decreases and between 5:00 and 8:00 the difference is no larger than 1.5%.

So the measured power matches the desired power profile quite closely. So the results from the simulation using the realistic battery model give a realistic view on the behavior of the battery.

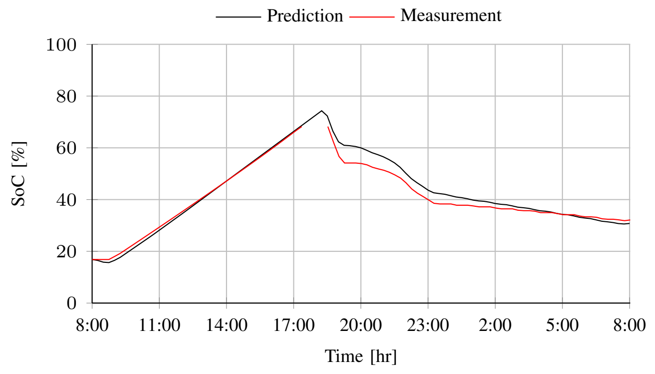

Fig. 7. Predicted SoC of one battery in the neighborhood, compared to the SoC of the same battery, calculated from the measurements.

D. *In conclusion*

The DiBu-model is quite suitable for use with the DEMKit simulation software. It was demonstrated (Fig. 4 and 5) that the power profiles realized with the DiBu-model flow match the planned power profiles quite closely, in some instances even more closely than the power profiles realized with the ideal model. It was also demonstrated (Fig. 6) that the power profiles realized with the DiBu-model match the realistic behavior of the battery quite well. The difference between the planned power profile, and realized power profile using the realistic model is largest when the battery is nearly full or nearly empty. An adaption to the optimization model could be made to improve on this. If stricter minimum and maximum power bounds are used as input to the scheduling algorithm, the scheduling algorithm is prevented from fully using the battery potential. However, this would result in reduced peak shaving performance. On the other hand, the unused capacity and power headroom can be used to resolve differences between the ideal battery used for scheduling and the presented realistic battery model. A minimum mismatch is essential in cases where contracts bind flexibility providers, such as aggregators, to their submitted profile plans on the day-ahead market.

V. Conclusions & Future work

In this work we presented the integration of a realistic battery model within the simulation and optimization software DEMKit. The realistic battery model results in an improved accuracy of the simulation results. Furthermore, the previously developed battery scheduling algorithm is, with minimal changes, able to create operation schedules for a battery in the realistic model. Thereby, the complexity (and required computational time) can be kept low. Deviations from the day-ahead planning and the actual realization on runtime are generally low.

Comparisons between the predicted SoC and the SoC calculated from measurements show that the predictions made using the DiBu model match reality very well. The difference between the DiBu-model prediction and the SoC_{meas} is generally less than 1,5 %. Only in the time interval where the largest amount of power is demanded from the battery, the deviation between the predicted SoC and SoC_{meas} increases.

For future work, the battery scheduling algorithm can be improved further with the presented results in mind. One observation is that the battery schedules are followed more accurately when the initial voltage (and SoC) at the moment of scheduling is close to the nominal value and specifications. It is expected that better initial values for the voltage van SoC, and the use of less than the full battery capacity for scheduling, would result in a planning that can be followed more accurately.

Future work should also be dedicated to a further validation of the DEMKit simulation environment, and the DiBu-model by comparing predictions for the neighborhood in Heeten, to measurements obtained from the actual neighborhood.

Furthermore it was observed that the battery analyzer equipment could not set charge or discharge currents below 0,05 A. It is expected that the *Battery Management Systems* (BMS) used in the Gridflex Heeten project has similar limitations. This could mean that the BMS is unable to carry out the planning made by DEMKit. Future work should be dedicated to investigating this, and adapting the DEMKit software to account for this.

Acknowledgements

The authors thank the Dutch national program TKI Switch2Smartgrids (project SGE), TKI iDeego (project ORTEP), and the Dutch organization RVO (project Grid Flex Heeten) for their support.

References

[1] P. Kriett and M. Salani, "Optimal control of a residential microgrid," *Energy*, vol. 42, no. 1, pp. 321 – 330, 2012.
[2] A. Molderink, "On the three step methodology for smart grids," Ph.D. dissertation, University of Twente, 2011.
[3] K. Anderson, J. Du, A. Naryan, and A. E. Gamal, "Gridspice: A distributed simulation platform for the smart grid," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2354 – 2363, 2014.
[4] G. Hoogsteen, "A cyber-physical systems perspective on decentralized energy management," Ph.D. dissertation, University of Twente, 2017.
[5] J. F. Manwell and J. G. McGowan, "Lead acid battery storage model for hybrid energy systems," *Solar Energy*, vol. 50, no. 5, pp. 399–405, 1993.
[6] J. Schiffer, D. U. Sauer, H. Bindner, T. Cronin, P. Lundsager, and R. Kaiser, "Model prediction for ranking lead-acid batteries according to expected lifetime in renewable energy systems and autonomous power-supply systems," *Journal of Power sources*, vol. 168, pp. 66–78, 2007.
[7] M. Doyle, T. Fuller, and J. Newman, "Modeling of galvanostatic charge and discharge of the lithium/polymer/insertion cell," *J. Electrochem. Soc.*, vol. 140, no. 6, pp. 1526–1533, 1993.
[8] B. Homan, R. P. van Leeuwen, L. Zhu, J. B. de Wit, and G. J. M. Smit, "Validation of a predictive model for smart control of thermal and electrical energy storage," in *IEEE Energycon 2016*, 2016, pp. 1 – 6.
[9] B. Homan, R. P. van Leeuwen, M. V. ten Kortenaar, and G. J. M. Smit, "A comprehensive model for battery state of charge prediction," in *IEEE Powertech 2017*, 2017, pp. 1 – 6.
[10] G. Hoogsteen, A. Molderink, J. L. Hurink, and G. J. M. Smit, "Generation of flexible domestic load profiles to evaluate demand side management approaches," in *IEEE Energycon 2016*, 2016, pp. 1 – 6.
[11] G. Hoogsteen, A. Molderink, J. L. Hurink, and G. J. M. Smit, "Asynchronous event driven distributed energy management using profile steering." in *IEEE Powertech 2017*, 2017, pp. 1 – 6.
[12] T. van der Klauw, M. E. T. Gerards, and J. L. Hurink, "Resource allocation problems in decentralized energy management," *OR Spectrum*, vol. 39, no. 3, pp. 749–773, 2017.
[13] "Gridflex heeten (http://www.gridflex.nl) - referenced 2018/02/20."
[14] M. E. T. Gerards, H. A. Toersche, G. Hoogsteen, T. van der Klauw, J. L. Hurink, and G. J. M. Smit, "Demand side management using profile steering," in *2015 IEEE Eindhoven PowerTech*, 2015, pp. 1–6.
[15] Cadex C8000 battery analyzer, "http://www.cadex.com/en/products/c8000-battery-testing-system," referenced 2018/2/27.