Adaptive Energy-Harvest Profiling to Enhance Depletion-Safe Operation and Efficient Task Scheduling[☆]

Christian Renner[*], Volker Turau

Hamburg University of Technology, Institute of Telematics, Hamburg, Germany

Abstract

Forecasting the expected energy harvest enables small-sized energy-harvesting sensor nodes to schedule tasks or adapt the radio duty cycle. This ability ensures depletion-safe and efficient operation. Most energy sources exhibit cyclic patterns of intensity, e.g., the sun. These patterns show periods with unequal—low versus high and stable versus varying—energy production and heavily depend on a node's location as well as seasonal and environmental changes. Existing forecast algorithms do not exploit these patterns, but create and update forecasts at static and arbitrary points in time, the main knob being the number of updates per cycle. We present a method enabling sensor nodes to adapt to harvesting patterns at runtime. It is designed for seamlessly replacing the static scheme to improve the accuracy of a wide range of existing forecast algorithms. In our evaluation, we show that (i) the adaptive method traces the energy pattern in real-world deployments accurately, (ii) reacts to seasonal and environmental changes, (iii) increases forecast accuracy, and (iv) reduces the number of prediction updates. These achievements enhance depletion-safe operation and efficient task scheduling with fewer recalculations and adjustments of the duty cycle. They also facilitate the exchange of harvesting forecasts for collaborative node tasks, since less information has to be shared.

Keywords: energy harvesting, energy-intake prediction, sustainable sensor networks

1. Introduction

Energy-efficient operation of wireless sensor network deployments has drawn considerable attention with emphasis on MAC [1] and routing protocols [2]. While this approach can merely prolong a network's lifetime, energy-harvesting sensor nodes [3, 4, 5] can virtually guarantee unlimited and uninterrupted operation. Among the resulting benefits of such a platform are

- avoiding gaps in collection data, which may reduce the expressiveness of measurements; and
- preventing manual intervention for battery replacement.

The latter usually infers large costs or logistic problems, particularly in harsh, difficult-to-reach, or large-scale setups [6, 7]. It may also cause severe intrusion into and an interference with the phenomena under investigation [8].

Energy-harvesting sensor nodes make perpetually operating and sustainable sensor networks possible. Yet, achieving non-intrusive monitoring of phenomena requires devices of tiny size. This demand is clearly opposed to using components that guarantee a sufficient amount of energy even under harsh conditions lasting for long periods of time [9].

Tiny components yet come at a non-negligible cost: Harvesting potential—i.e, the amount of energy that can be drawn from the environment—is decreased. The harvester has to comply with the average power demand of the node's hardware w.r.t. the running application and algorithms. However, the actual extent of the harvest may not be known precisely prior to deployment and is not accounted for by global weather forecasts. The extent often depends on the exact positioning of the harvester, since local or seasonal effects dominate energy production—e.g., solar-powered nodes close to the sun-averted side of a building suffer from shadows. Aging effects, dirt, and environmental changes reduce the potential of the harvester. Hence, a sensor node must be able to adapt its power consumption profile to these conditions. Important research on this topic has already been carried out [10, 11, 12, 13, 14, 15, 16]. In conclusion, nodes have to adjust their schedule to achieve energy-neutral operation [17], i.e., they must not exceed the amount of harvested energy within a given period of time. For periods of low or no harvest, a node depends on the capacity of its energy-buffer. Supercaps

[☆]Research supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG), contract number TU 221/4-1

[*]Corresponding author at: Hamburg University of Technology, Institute of Telematics, Schwarzenbergstrasse 95, 21073 Hamburg, Germany, Tel.: +49 40 428 78 3746; Fax: +49 40 428 78 2581

Email addresses: christian.renner@tu-harburg.de (Christian Renner), turau@tu-harburg.de (Volker Turau)

URL: http://www.ti5.tu-harburg.de/staff/renner (Christian Renner), http://www.ti5.tu-harburg.de/staff/turau (Volker Turau)

have been frequently used, since they combine small size and cheap prices while providing enough capacity to back up a node's operation for several days [18] and offer virtually unlimited charge cycles.

Employing a pattern of energy production for the harvester output with respect to the environment during runtime improves the operation efficiency of sensor nodes and their robustness against temporary energy depletion, e.g.:

- Throttling the radio duty cycle is unnecessary, if the energy buffer is close to depletion, but energy can be harvested in the near future
- Short periods of high harvesting potential can be used for fast and high-volume data exchange or collection, because there is no need for radio duty-cycling
- More stable routing paths can be established, if perspective energy harvest is integrated into path establishment
- Delay-tolerant tasks can be deferred and sampling rates reduced, if residual energy and expected harvest are low

Therefore, prediction techniques (focusing on solar harvesters) have been introduced to wireless sensor nodes [19, 20, 21, 17] that exploit quasi-cyclic behavior of harvesters. While these approaches improve energy-awareness with respect to future energy resources, they have two major drawbacks. Firstly, they incorporate many influencing factors and parameters with partly uncertain impact. A detailed analysis on the individual influences has not been carried out in detail. Secondly, the prediction schemes rely on static update intervals of equal lengths. Times of low dynamics thus get the same attention as do more dynamic phases, although more concentrated attention on the latter is required to efficiently update task schedules or detect imminent threats of energy depletion—e.g., there is no need to reassess the harvester output during night times for a solar harvester. In contrast, short phases with high harvester output but strong dynamical behavior among cycles need prompt reassessment in order to react to over- or underestimations. Non-equally distributed, but predefined prediction points are no remedy, if the harvesters deliver distinct and unusual patterns of energy—e.g., a solar-harvesting node under a tree may harvest energy in the morning and afternoon, but not at noon, when the tree shades the harvester. In the winter however, when the tree bears no leafs, harvesting is possible without interruption.

In this paper we address these issues and make the following contributions. Firstly, we define a theoretic foundation for identifying the pattern of energy-harvester output, enabling an improved timing for prediction and a node's task schedule adaptation. Secondly, we derive a practical, adaptive algorithm with small memory-footprint and low computational power achieving a considerable improvement over static time slot distributions. Thirdly, we evaluate the accuracy of existing prediction algorithms using our new slotting scheme and present a detailed comparison. In this context, we also explore the influence of individual parameters and discard those which have low or no substantial influence on prediction accuracy. We finally give practical advice on tailoring harvester prediction schemes to real-world applications and present considerations about future research directions regarding more elaborate prediction schemes.

2. Overview

This section outlines the fundamentals of forecasting the behavior of an energy harvester for sensor nodes. The subsequent sections provide formal definitions of the concepts.

2.1. Objective

Efficient utilization of available energy resources is mandatory for tiny energy-harvesting sensor nodes. To meet this end, the following situations must be avoided:

1. *Empty energy buffer at times of low harvesting potential*: A node cannot continue operation until energy can be harvested again
2. *Full energy buffer during times of high harvesting potential*: Additional energy produced by the harvester cannot be stored and has to be consumed directly or is lost otherwise

Knowing the performance pattern of its energy harvester, a sensor node is enabled to avoid these situations by scheduling its tasks appropriately. For example, a node can adjust sensor sampling rates, adapt its radio duty cycle, or alter the level of data aggregation for sent radio messages.

In the following, we define a set of assumptions for forecasting the harvester performance. We sketch the basic idea of forecasting techniques and introduce our novel approach.

2.2. Model Assumptions

To achieve a comprehensible explanation of our pattern-detection scheme, we make the following assumptions:

1. Time is divided into equidistant, discrete time steps $t = 0, 1, 2, \ldots$
2. Harvester output—e.g., current or power—is sampled at these time steps. The samples are denoted $h[t]$
3. Harvester output is cyclic with a cycle length of T time steps. The term cyclic means that the autocorrelation of the series $h[.]$ has a strong local maximum at time T, so that forecasting future values is feasible. The cycle length is inherent to the type of harvester and is derived empirically, e.g., it is 24 h for outdoor solar harvesters

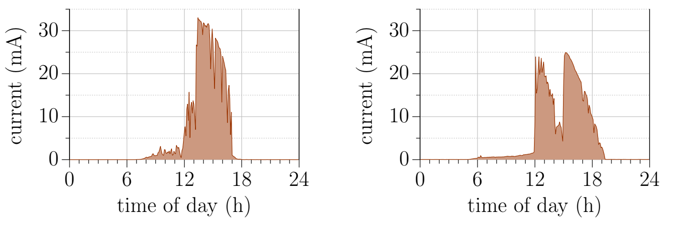


Figure 2: Solar harvester output on two sunny days in March (left) and July (right), recorded by a sensor node placed on a window sill facing westwards

Figure 1a depicts a schematic example of a harvester output meeting these assumptions. Two different days (cycles) of a real solar harvester are displayed in Fig. 2: A sensor node samples the produced current of the solar cell every 30 s. The two days exhibit the required correlation, but there is a seasonal influence on harvester output.

2.3. General Approach

Our goal is to identify the pattern of harvester output and create a forecast of the future performance of the harvester output at a time of prediction denoted $\hat{t}$ utilizing a memory of a cycle length T. The prediction horizon is of length Z time steps, i.e., we want to forecast the unknown future harvesting potentials $h[t]$, $t = \hat{t}+1, \hat{t}+2, \ldots, \hat{t}+Z$. An example forecast with prediction horizon of $Z = 6$ for a harvester output with cycle $T = 12$ is shown in Fig. 1b.

This approach is memory-demanding, since it requires storing the latest T past values $h[t]$ while producing Z new values. For a solar harvester with a sampling period of 30 s (cf. Fig. 2), the cycle length is $T = 2\,880$. If samples could be stored as 10 bit values (the resolution provided by many sensor node ADCs), memory consumption for a prediction horizon of a complete cycle (i.e., $Z = T = 2\,880$) would be as large as 7.2 kB. Even modern sensor nodes rarely offer more than 8 or 16 kB, leaving almost no room for application memory. Reducing the sampling rate—i.e., skipping samples—is not desired, as it decreases the accuracy and reliability: Fig 2 shows a large sample variation within short time.

Many approaches therefore divide a cycle into a constant number of equally-sized slots [19, 14, 22]. We call this a static slot distribution. Instead of storing all harvester samples, only the mean value of samples within a slot is stored. For a slot length of, e.g., 30 min, cycles are divided into 48 slots each embracing 60 samples. Memory consumption for a prediction horizon of one cycle, i.e., 48 slots, is reduced to 96 B (for a realistic implementation with 16 bit integers).

While this approach conserves memory, it reduces the accuracy of the harvester output pattern and the forecast. Available evaluation results of the various approaches suggest that in many cases slot length correlates with prediction error [20]. This implies that an increased representation error—i.e., the error induced by averaging samples—leads to an increased forecasting error. Yet, increasing the number of slots in a static distribution is inefficient: Fig. 2 reveals that many slots are virtually useless, because the harvesting output is zero. In the example above, roughly half of the slots are situated at times in which there is no sunlight. If slots during those periods would be longer, shorter slots could be used at times of high dynamics, e.g., in the afternoon. For this purpose, we suggest using variable-length slots. This implies that slots have different lengths, which can also be changed (adapted) so as to react to local conditions or seasonal changes. The underlying idea is to effectively decrease the representation error where it is particularly large, to achieve a reduced forecasting error. Although slot lengths have to be stored, they can be efficiently fitted into the remaining bits of the variables used for storing the slots' mean values: For ADCs with a resolution of 10 bit, most programmers will use 16 bit integers for storing each value, leaving 6 otherwise unused bits for holding slot lengths.

We divide a cycle c into a series $\mathcal{S}_c$ of S slots, not necessarily of equal length. The number of slots is constant in all cycles, the reason being that (i) most sensor node operating systems do not offer dynamic memory, and (ii) leaving slots unused will decrease accuracy, implying that using the maximum number of available slots would be the result. A slot $\mathbf{s}$ has a start time $\tau_\mathbf{s}$ and a length of $l_\mathbf{s}$ time steps, plus it maps a representative value $\mu_\mathbf{s}$ to all harvesting potentials embraced. This value could be the mean value as in existing approaches. Sect. 3 provides additional details. The lengths of slots are variable and chosen so as to decrease the representation error w.r.t. an error metric—e.g, the mean absolute error (MAE) or the mean square error (MSE). This also reduces the forecast error; forecasting techniques are discussed in Sect. 5.

The static and variable slotting techniques are shown in Fig. 1c and 1d. Slot lengths in Fig. 1c have a (fixed) length of 3 time steps, whereas slots have different lengths in Fig. 1d. The representation error per cycle is notably decreased. This is evident from the times $t = 18, 19, 20$.

3. Optimal Slot Distribution

In this section, we give a formal definition of the slot distribution concept illustrated in Sect. 2. We also show how to efficiently determine an optimal slot distribution for a cycle with respect to a given error metric.

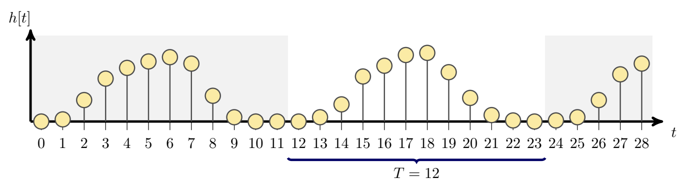
3.1. Time and Harvesting Potential

The series of harvesting potentials $h[t] \in \mathbb{R}^+$ sampled in the interval represented by time step $t \in \mathbb{N}$ is defined as

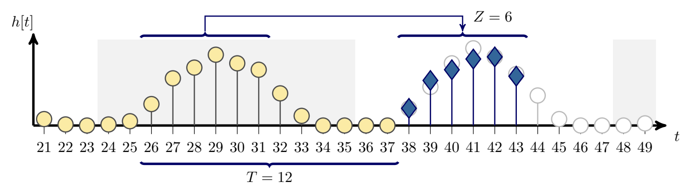
$$\mathcal{H} = \langle\, h[0], h[1], \ldots \,\rangle \tag{1}$$

The series $\mathcal{H}$ is cyclic with length T, and w.l.o.g. we assume that the first cycle of the observation time starts at $t = 0$. We can therefore uniquely decompose any global time step t into a cycle number $c \in \mathbb{N}$ and a cycle time τ:

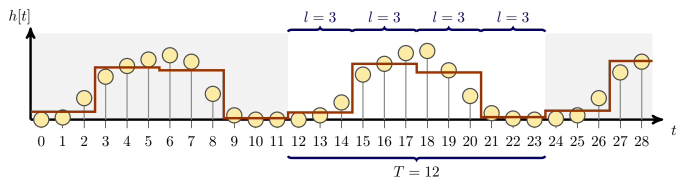
$$t = c \cdot T + \tau \qquad (0 \leq \tau < T)\ . \tag{2}$$



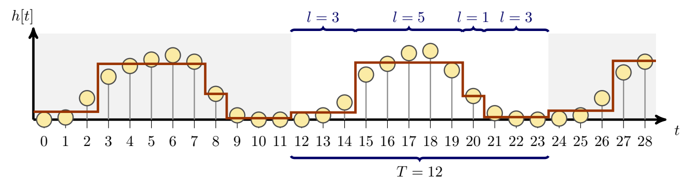
(a) Cyclic harvester output h sampled at time steps t. The output has an observable cycle length of $T = 12$.



(b) Forecast at time step $\hat{t} = 37$: light circles indicate the a-posteriori real samples, diamond marks represent the predictions for $Z = 6$



(c) Harvester output represented by constant-length slots



(d) Harvester output represented by variable-length slots

Figure 1: An example of cyclic harvester output with possible slot distributions and an example forecast

With this definition, we introduce two shortcut notations:

$$h_c[\tau] \;=\; h[c \cdot T + \tau] \qquad (0 \le \tau < T) \tag{3}$$
$$\mathcal{H}_c \;=\; \langle\, h_c[0], \ldots, h_c[T-1] \,\rangle \tag{4}$$

3.2. Variable-Length Slot Distribution

Each series $\mathcal{H}_c$ is represented by a series $\mathcal{S}_c$ with a constant number of S variable-length slots $\mathbf{s} \in \mathcal{S}_c$ with start time $\tau_{\mathbf{s}}$, length $l_{\mathbf{s}}$, and value $\mu_{\mathbf{s}}$ that represents the harvesting samples embraced. Slots are non-overlapping, so that $\sum_{\mathbf{s}\in\mathcal{S}_c} l_{\mathbf{s}} = T$. The start time of a slot $\mathbf{s}$ hence evaluates to

$$\tau_{\mathbf{s}} = \sum_{\mathbf{r}\in\mathcal{S}_c, \tau_{\mathbf{r}}<\tau_{\mathbf{s}}} l_{\mathbf{r}} \,. \tag{5}$$

From these definitions we derive the slot representation series of cycle c

$$\tilde{\mathcal{H}}_c = \langle\, \tilde{h}_c[0], \ldots, \tilde{h}_c[T-1] \,\rangle \tag{6}$$

with its individual values

$$\tilde{h}_c\,[\tau] = \mu_{\mathbf{s}} \quad (\mathbf{s} \in \mathcal{S}_c,\ \tau = \tau_{\mathbf{s}}, \ldots, \tau_{\mathbf{s}} + l_{\mathbf{s}} - 1) \tag{7}$$

Figure 1d serves as an illustration of these definitions. The circles show the series $\mathcal{H}_c$, while the horizontal bars track $\tilde{\mathcal{H}}_c$ (defined by the slot lengths and values).

3.3. Optimal Slot Distribution

The choice of the parameters $\mu_{\mathbf{s}}$ and $l_{\mathbf{s}}$ is subject to minimizing the representation error of actual harvesting potential $\mathcal{H}_c$ and slot representation $\tilde{\mathcal{H}}_c$. Given an error metric f_e—e.g., the absolute error $f_e(x, y) = |x - y|$ or the squared error $f_e(x, y) = (x - y)^2$—we define the representation error of a single slot $\mathbf{s}$ in cycle c as

$$e_c\,(\mathbf{s}) = \left| \sum_{\tau=\tau_{\mathbf{s}}}^{\tau_{\mathbf{s}}+l_{\mathbf{s}}-1} f_e\,(h_c[\tau], \mu_{\mathbf{s}}) \right| \quad (\mathbf{s} \in \mathcal{S}_c)\,. \tag{8}$$

The representation error of a series of slots $\mathcal{S}$ with respect to $\mathcal{H}_c$ is defined as

$$E_c\,(\mathcal{S}) = \sum_{\mathbf{s}\in\mathcal{S}} e_c\,(\mathbf{s}) \tag{9}$$

The slot value $\mu_{\mathbf{s}}$ yielding the minimum error of a single slot $\mathbf{s}$ with given length $l_{\mathbf{s}}$ is obtained by solving

$$\frac{\mathrm{d}E_c}{\mathrm{d}\mu_{\mathbf{s}}} = \frac{\mathrm{d}e_c(\mathbf{s})}{\mathrm{d}\mu_{\mathbf{s}}} \overset{!}{=} 0\,. \tag{10}$$

For the example of the squared error, the minimum error is achieved by choosing $\mu_{\mathbf{s}}$ to be the mean of the harvesting samples embraced by slot $\mathbf{s}$.

Slot lengths $l_{\mathbf{s}}$ are chosen to minimize the error E_c within a cycle c:

$$E^*_c\,(S) = \min_{\mathcal{S}} E_c\,(\mathcal{S})\,. \tag{11}$$

Deriving (9) by the S parameters $l_{\mathbf{s}}$ yields S equations. The variables $l_{\mathbf{s}}$ appear as parameters in the summation of the corresponding slot and in the equation for $\mu_{\mathbf{s}}$. Due to the summation constraint $\sum_{\mathbf{s}\in\mathcal{S}_c} l_{\mathbf{s}} = T$, slot lengths are not independent. An analytical solution of the minimization problem does therefore not exist in general—this is, e.g., the case for the quadratic and absolute error metrics (see above). The naive approach for finding the minimum requires inspecting all admissible solutions. With the first slot always starting at the beginning of a cycle and no slot having zero length, this is equivalent to choosing $S - 1$ starting times from $T - 1$ possibilities. This gives a search space of $T - 1$ choose $S - 1$, rendering this approach infeasible. Yet, the following observations lead to an efficient solution algorithm using dynamic programming (DP).

If S' slots have already been distributed until time τ, and if the representation error of this partial distribution is minimal, the additional error of the slot distribution for the remainder of the cycle with the available $S - S'$ slots is

```
// Initialization
best(*Smallest error with $S'$ slots at time $\tau$*);
prev(*Predecessor for $S'$ slots at time $\tau$*);
best = $(b_{\tau,S'})^{T\times S}$, $b_{\tau,S'} = \infty$;
prev = $(p_{\tau,S'})^{T\times S}$, $p_{\tau,S'} = \varnothing$;

for $\tau = 0,\ldots,T-S$ do
    best $[\tau, 0] = e_c$(slot from 0 to $\tau$);
    prev $[\tau, 0] = 0$;

// DP over all times $\tau$ in cycle
for $\tau = 1,\ldots,T-1$ do
    // and all possible starting times $\tau^*$
    for $\tau^* = 1,\ldots,\tau$ do
        // get error once for any $S'$
        err = $e_c$(slot from $\tau^*$ to $\tau$);
        // process possible number of used slots
        for $S' = \max(0, S-1-T+\tau)),\ldots,S-3+\left\lfloor\frac{\tau}{T-1}\right\rfloor$ do
            b = best [ t0-1,s ] + err;
            if b ¡ best [$\tau$, S'+1] then
                best [$\tau$, S'+1] = b;
                prev ($\tau$, S'+1] = $\tau^* - 1$ ;

// obtain slot lengths of best distribution
p = $T - 1$;
for $S'$ = $S-1$,...,0 do
    $l_{S'}$ = p − prev [p,$S'$];
    p = prev [p,$S'$];
```

Algorithm 1: Dynamic Programming approach to compute the optimal slot distribution for a single cycle

independent from the former distribution. This follows directly from (8). For a given optimal slot distribution with intermediate point τ, the induced slot representation $\mathcal{S}_1$ on $[0,\tau]$ and $\mathcal{S}_2$ on $[\tau+1, T-1]$ are optimal. In contrast, the number of slots consumed at intermediate time τ influences the additional error of the possible distributions on $[\tau+1, T-1]$, e.g., a lower error at τ using one additional slot may lead to a larger overall error.

Thus, we can perform dynamic programming over time τ while considering for each τ the number S' of already used slots. The formal algorithm is displayed in Alg. 1 and has an upper-bound complexity of $S \cdot T^2/2$; yet, overall runtime also depends on the complexity of computing e_c. The complexity can be further reduced by using lower and upper bounds for slot lengths. However, finding the optimal slot distribution is only discussed for the comparison (see Sect. 7) with the adaptive slotting scheme presented in the following section.

4. Adaptive Slot Distribution

An optimal slot distribution as defined in Sect. 3 can only be achieved with memory resources and computing power beyond the scope of even modern wireless sensor nodes. For the solar harvesting example in Sect. 2.3, the DP requires roughly $2\,880 \cdot 1\,440 \cdot 48 \approx 2 \cdot 10^8$ iterations and a storage space of up to $2\,880 \cdot 48$ values.

Thus arises the need for a resource-efficient algorithm that improves, or adapts, the current slot distribution to decrease the error. To preserve memory and computing resources, we suggest a scheme based on local decisions—i.e., making decisions based on the evaluation of a single slot at a time—that has the additional advantage of decreasing the number of stored samples h_c. In this section we provide the relaxed problem statement, define operations for adapting an existing distribution and discuss their practical application.

4.1. Problem Statement

Given the slot representation $\tilde{\mathcal{H}}_{c-1}$: At the end of the following cycle c, an adapted representation $\tilde{\mathcal{H}}_c$ has to be determined that reduces the representation error w.r.t. $\mathcal{H}_c$. We thus seek to improve $\mathcal{S}_{c-1}$ yielding the distribution $\mathcal{S}_c$ with

$$E_c\left(\mathcal{S}_{c-1}\right) \;>\; E_c\left(\mathcal{S}_c\right)\ . \tag{12}$$

This requirement is an relaxation of (11), since we are no longer searching for the best possible slot distribution for $\mathcal{H}_c$. In contrast, the idea is to approach this distribution by adapting $\tilde{\mathcal{H}}$ cycle-wise. However, $\tilde{\mathcal{H}}$ will not converge to a static distribution, since $\mathcal{H}_c$ changes from cycle to cycle; but $\tilde{\mathcal{H}}$ will trace these changes and improve forecasting accuracy. Note that changes between cycles imply that the representation error of the new cycle may be larger than in the previous one, even after the slot distribution has been adapted. This is particularly the case, if $\mathcal{H}_c$ is largely different from $\mathcal{H}_{c-1}$.

4.2. Operations for Adapting the Distribution

A fast and resource-efficient adaptation of an existing slot distribution is obtained by two separate procedures: splitting existing and merging adjacent slots.

4.2.1. Splitting Slots

Splitting a slot $\mathbf{s} \in \mathcal{S}_{c-1}$ into S' slots $\mathbf{r}_i$ with values $\mu_{\mathbf{r}_i}$ and lengths $l_{\mathbf{r}_i}$ $(0 \le i < S')$ is performed with the objective of minimizing the representation error, i.e, finding $E_c^*\left(\langle\,\mathbf{r}_0,\ldots,\mathbf{r}_{S'-1}\,\rangle\right)$ with the constraint $l_{\mathbf{r}_0}+\ldots+l_{\mathbf{r}_{S'-1}} = l_\mathbf{s}$ and $\tau_\mathbf{s} = \tau_{\mathbf{r}_0}$. We call this a S'-split operation. The optimal solution for the splitting problem of a single slot can be obtained by solving (10) with Alg. 1. The maximum reduction of the representation error of a S'-split operation on slot $\mathbf{s}$ is

$$C_\cap(\mathbf{s}, S') = e_c\left(\mathbf{s}\right) - E_c^*\left(\langle\,\mathbf{r}_0,\ldots,\mathbf{r}_{S'-1}\,\rangle\right)\ . \tag{13}$$

4.2.2. Merging Slots

A subseries of adjacent slots $\mathcal{S}' \subset \mathcal{S}_{c-1}$ $(|\mathcal{S}'| \ge 2)$ can be merged into a new single slot $\mathbf{r}$ with value $\mu_\mathbf{r}$ and length $l_\mathbf{r}$ by adding the lengths $l_\mathbf{s}$ $(\mathbf{s} \in \mathcal{S}')$ and applying

(10) to the merged slot $\mathbf{r}$. We call this a S'-merge operation. Merging slots yields an representation error increase of

$$C_\cup(\mathcal{S}') = e_c(\mathbf{r}) - E_c(\mathcal{S}') \ . \tag{14}$$

4.3. Admissible Adaptations

An adaptation of a slot distribution with split and merge operations must be admissible, i.e, condition (12) must be satisfied and the number of slots has to be constant (cf. Sect. 2.3). The memory footprint and computing power for such an operation must be low for practical execution on sensor nodes. We thus suggest the following procedure:

1. Calculate $C_\cup(\mathcal{S}')$ for all $\mathcal{S}' \subset \mathcal{S}_{c-1}$ for a constant $S' = |\mathcal{S}'|$ and store the results
2. Calculate and store $C_\cap(\mathbf{s}, S')$ for all $\mathbf{s} \in \mathcal{S}_{c-1}$ for the same constant S' and store the results
3. While the minimum $C_\cup$ and maximum $C_\cap$ satisfy the condition $C_\cup < C_\cap$—i.e., (12) holds—perform the corresponding split and merge operations, and remove these $C_\cup$, $C_\cap$ pairs

It is possible to perform steps 1 and 2 with a range of S' values. However, the storage requirement is $2S$ for each value of S', so that using more than one value for S' may not be feasible. Furthermore, such an approach requires to expand step 3 to find the optimum among all possible combinations, e.g., it would be possible that the combination of two 2-merge operations and only one 3-split operation gives the best result.

4.4. Practical Adaptations

To keep computing and storage complexity at a low level, the number of possible adaptation combinations has to be small. Hence, we suggest to exclusively consider 2-merge and 2-split operations. Early experiments showed that this restriction has very low influence only. We also reduce the number of stored values $C_\cap$ and $C_\cup$ to the B best ones each, so that at most B split and merge operations are performed after each cycle. The quality of this approach is discussed in Sect. 7. Besides complexity, there is another reason for such an approach: In many applications (e.g., solar harvesting) the average harvesting pattern of cycles changes slowly, so that adaptations of the slot distribution become rare, once a good representation is found. However, there may be outliers—e.g., a rainy day in the summer—in which the representation could be poor. If an adaptation scheme allows for massive changes after each cycle, such an outlier may produce an adapted slot distribution not suitable for the average case. This problem is closely related to overfitting.

We also introduce minimum $L_{\min}$ and maximum $L_{\max}$ slot lengths for two reasons. Defining a minimum slot length has a practical cause: There may be energy intake fluctuations within a cycle, that are not cyclic. Solar harvesters suffer from short phases of clouds within a day that lead to significantly lower harvesting values. Since these phases are not likely to occur exactly at the same time on the following day, assigning a slot to these phases should be prevented. Restricting the maximum length of slots allows for storing slot lengths without additional storage space (cf. Sect. 2.3).

To reduce the likelihood of high deviation of harvesting values within a slot, we choose the squared error metric $f_e(a,b) = (a-b)^2$, yielding

$$\mu_\mathbf{s} = \frac{1}{l_\mathbf{s}} \sum_{\tau=\tau_\mathbf{s}}^{\tau_\mathbf{s}+l_\mathbf{s}-1} h_c[\tau] \tag{15}$$

according to (10). For 2-merge and 2-split operations, the following equation is relevant:

$$l_\mathbf{r} \cdot \mu_\mathbf{r} = l_{\mathbf{s}_1} \cdot \mu_{\mathbf{s}_1} + l_{\mathbf{s}_2} \cdot \mu_{\mathbf{s}_2} \ , \tag{16}$$

where $\langle \mathbf{s}_1, \mathbf{s}_2 \rangle$ are two adjacent slots that are embraced by the new slot $\mathbf{r}$. The error of a single slot $\mathbf{s}$ in (8) simplifies to

$$e_c(\mathbf{s}) = \sum_{\tau=\tau_\mathbf{s}}^{\tau_\mathbf{s}+l_\mathbf{s}-1} (h_c[\tau] - \mu_\mathbf{s})^2 = \sum_{\tau=\tau_\mathbf{s}}^{\tau_\mathbf{s}+l_\mathbf{s}-1} h_c^2[\tau] - l_\mathbf{s} \cdot \mu_\mathbf{s}^2 \ , \tag{17}$$

so that there is no need to store the individual values $h_c[\tau]$ for obtaining the errors.

4.4.1. 2-Split Operations

Splitting a slot $\mathbf{s}$ into two separate slots $\mathbf{r}_0$ and $\mathbf{r}_1$ produces the accuracy gain

$$\begin{aligned} C_\cap(\mathbf{s}, 2) &= e_c(\mathbf{s}) - (e_c(\mathbf{r}_0) + e_c(\mathbf{r}_1)) \\ &= \frac{l_\mathbf{s} \cdot l_{\mathbf{r}_0}}{l_\mathbf{s} - l_{\mathbf{r}_0}} (\mu_\mathbf{s} - \mu_{\mathbf{r}_0})^2 \ . \end{aligned} \tag{18}$$

To obtain the largest accuracy gain, we need to find

$$\max_{1 \le l_{\mathbf{r}_0} < l_\mathbf{s}} \{C_\cap(\mathbf{s}, 2)\}. \tag{19}$$

This optimum can only be determined at the end of the slot, when $\mu_\mathbf{s}$ is known, and if all value pairs $(l_{\mathbf{r}_0}, \mu_{\mathbf{r}_0})$ are known. The storage demand therefore depends linearly on the number of harvesting values per slot. For this reason, we introduce the constant C that defines the number of candidate split points analyzed per slot. These candidate points are equally distributed over the analyzed slots (in terms of multiples of the minimum slot length $L_{\min}$).

4.4.2. 2-Merge Operations

A 2-merge operation at the end of cycle c on the adjacent slots $\langle \mathbf{s}_1, \mathbf{s}_2 \rangle \subset \mathcal{S}_{c-1}$ yields the merged slot $\mathbf{r}$ (cf. Sect. 4.2.2). The accuracy loss induced is

$$\begin{aligned} C_\cup(\langle \mathbf{s}_1, \mathbf{s}_2 \rangle) &= e_c(\mathbf{s}_\mathbf{r}) - (e_c(\mathbf{s}_{\mathbf{s}_1}) + e_c(\mathbf{s}_{\mathbf{s}_2})) \\ &= l_{\mathbf{s}_1} \cdot \mu_{\mathbf{s}_1}^2 + l_{\mathbf{s}_2} \cdot \mu_{\mathbf{s}_2}^2 - l_\mathbf{r} \cdot \mu_\mathbf{r}^2 \\ &= \frac{l_{\mathbf{s}_1} \cdot l_{\mathbf{s}_2}}{l_{\mathbf{s}_1} + l_{\mathbf{s}_2}} (\mu_{\mathbf{s}_1} - \mu_{\mathbf{s}_2})^2 \ . \end{aligned} \tag{20}$$

The calculation of $C_\cup$ thus only requires the already stored lengths and values of the involved slots.

4.4.3. *Initial slot distribution*

Upon boot-time of a node, an initial slot distribution must be available. Here, using expected slot distributions derived from empirical data or simple models are advisable—e.g., long slots at the beginning and end of a cycle plus equally distributed ones in between for outdoor solar harvesters. Otherwise, we recommend lengths to provide an unbiased home position, i.e, slots of equal or close-to-equal lengths.

4.4.4. *Implementation*

We evaluated the memory consumption of our adaptive scheme versus static slots using an implementation for TinyOS 2.1 that we currently employ in a real-world installation. Our implementation analyzes and temporarily stores $C = 3$ equidistant split points of the current slot and stores the best $B = 1$ slot for splitting and merging each. Harvester output is the average of 12 samples taken every 30 s, yielding an effective sample period of 5 min. We chose $L_{\min} = 1$ and $L_{\max} = 64$ (w.r.t. 5 min intervals), so that slot lengths can be stored along with slot values in a 16 bit unsigned integer (the ADC has a 10 bit resolution). The choice of parameters complies with the results presented in Sect. 7 and 8.

Using the same number of slots, RAM overhead is 16 byte compared to static slots. For example, 12 adaptive slots consume 42 bytes, whereas 12 static slots need 26 bytes. Memory consumption for 24 slots is 66 byte for adaptive and 50 byte for static slots. For each slot of a cycle, computational overhead is restricted to storing the current mean values of the harvester output at the possible split points, finding the optimum per slot and possibly replacing the stored optimal split point with the optimal one of the current slot. At the end of each cycle, merging costs are calculated and at most $B = 1$ split-and-merge operation executed. The current implementation requires less than 2.1 kB additional program memory compared to an implementation without slot adaptations. Floating-point operations are completely avoided, and the main portion of program memory is due to multiplications and divisions, which are supported on the used IRIS platform in software only.

5. Harvesting Forecast

This section introduces and explains the forecasting methods used to compare the prediction accuracy of static and adaptive slot distributions.

5.1. *Tracing Trends along Cycles*

In Sect. 2.3 we have introduced the slot value $\mu_{\mathbf{s}}$, which represents the harvester output in slot $\mathbf{s}$ of the previous cycle. This value may be used for prediction directly, but most existing forecasting schemes calculate a smoothed or average value $\bar{\mu}_{\mathbf{s}}$ based on recent cycles.

The authors of [19] store the values $\mu_{\mathbf{s}}^i$ of D cycles and use the per-slot average for prediction:

$$\bar{\mu}_{\mathbf{s}} \leftarrow \frac{1}{D} \sum_{i=0}^{D-1} \mu_{\mathbf{s}}^i \tag{21}$$

This method has a memory footprint of $S \cdot D$ slots and requires calculation of the mean for each slot and cycle.

A more resource-efficient method is to calculate an exponentially weighted moving average as in [17]:

$$\bar{\mu}_{\mathbf{s}} \leftarrow \alpha \cdot \bar{\mu}_{\mathbf{s}} + (1-\alpha) \cdot \mu_{\mathbf{s}} \tag{22}$$

Here, the memory footprint is reduced to S slots.

These smoothing approaches are compared in Sect. 8.2. Both are compatible with the adaptive slot distribution presented in this paper: The slot values of previous cycles have to be mapped to a (possibly) adapted slot distribution according to the split and merge operations explained in Sect. 4.2.1.

5.2. *Short-Term Prediction*

The weather-conditioned moving average (WCMA) prediction algorithm has drawn much attention [19, 20, 21]. Its underlying idea is to calculate a prediction $\hat{\mu}_{\mathbf{s}}$ for the following slot $\mathbf{s}$ based on (i) the smoothed slot value $\bar{\mu}_{\mathbf{s}}$, (ii) the value $\mu_{\mathbf{r}_0}$ of the current slot $\mathbf{r}_0$, (iii) a scaling factor Λ obtained from the ratio of the values and corresponding smoothings of the most recent K slots $\mathbf{r}_0, \ldots, \mathbf{r}_{K-1}$, and (iv) a trend weight $\omega \in [0,1]$:

$$\hat{\mu}_{\mathbf{s}} = \underbrace{\omega \cdot \Lambda \cdot \bar{\mu}_{\mathbf{s}}}_{\text{trend portion}} + \underbrace{(1-\omega) \cdot \mu_{\mathbf{r}_0}}_{\text{current portion}} . \tag{23}$$

The trend scale Λ is calculated using

$$\Lambda = \sum_{k=0}^{K-1} \lambda(k) \cdot \frac{\mu_{\mathbf{r}_k}}{\bar{\mu}_{\mathbf{r}_k}} , \quad \lambda(k) = \frac{2 \cdot (K-k)}{K \cdot (K+1)} . \tag{24}$$

The quotient in the calculation of Λ bears the problem of possible divisions by zero. We avoid this by replacing the quotient with the value 1, if $\bar{\mu}_{\mathbf{r}_k} = 0$.

This prediction scheme uses the two additional parameters ω and K, while also introducing the weights λ. We will address the influence of these parameters in Sect. 8.4. An important, implicit factor of the prediction scheme is its limited prediction horizon, which is coupled with slot lengths, i.e., $Z = l_{\mathbf{s}}$. We will also analyze this detail.

5.3. *Long-Term Prediction*

A resource-efficient method of generating long-term predictions is to use the smoothed slot values $\bar{\mu}_{\mathbf{s}}$ as predictions $\hat{\mu}_{\mathbf{s}}$ directly [17]. We are not aware of any improved version of this approach. The prediction horizon of such an approach is $Z = T$, because $\hat{\mu}_{\mathbf{s}} = \bar{\mu}_{\mathbf{s}}$ is only updated at the end of slot $\mathbf{s}$.

dataset	days	year	type / unit	location
ECSU	365	2010	solar / irradiance	Elizabeth City, NC
LOLH	365	2010	solar / irradiance	La Ola Lanai, HI
SPMD	365	2009	solar / irradiance	South Park, CO
TUHH	194	2010	solar / current	Hamburg, Germany

Table 1: Datasets used for evaluation

6. Data Basis and Metrics

This section introduces the datasets and metrics used for the evaluation in the subsequent sections.

6.1. Data Basis

For the evaluation of the adaptive and static slotting distribution, we used four solar datasets (cf. Table 1). The first three were taken from the Measurement and Instrumentation Data Center (MIDC) [23]. The fourth one (TUHH) was recorded by the authors with the solar harvester presented in [18]. It was placed outside on a window sill facing westwards on the fourth floor of the main University building. We used the TUHH dataset to evaluate the performance of the slotting and prediction schemes for a sensor node placed in an arbitrary but realistic position. To achieve a comparable data basis, we converted all datasets (if necessary) to 288 values per day, each representing the average harvester output within 5 min base intervals.

The datasets reveal different characteristics regarding the shape of the measured values on a day's cycle. Figure 3 shows a descriptive statistic for the harvester output within each base interval of the observation time. The distinctive pattern of the TUHH dataset in Fig. 3a stems from the shade produced by the building in the morning. The noticeable dip between 2 and 3 PM is caused by a roof overhang. In contrast, the ECSU dataset in Fig. 3b is very symmetric and regular, indicating perfect harvesting conditions. LOLH and SPMD (Fig. 3c and 3d) experience lower values in the afternoon, most likely caused by more clouds in the afternoon.

6.2. Metrics

To evaluate the representation and prediction accuracy, we have applied several metrics. Due to similar qualitative results, we exclusively show results of the root mean square error (RMSE). We also calculated the normed cross-correlation function (NCCF), the mean error, and the mean absolute deviation (MAE). All metrics were calculated on a per-day basis, i.e., we first determined the prediction for each day and then computed the metric for that day. We did not calculate relative errors, because we believe that for most applications, absolute values are of higher relevance, e.g., a 50% error at times of almost zero harvester output does not significantly change the charging process of a sensor node's energy buffer, whereas the contrary is the case at times of high harvester output. In addition, using relative errors in the scenario of energy harvesters implies difficulties at times of low harvester output. Here, small absolute deviations may lead to almost infinite relative changes, e.g., harvesting 1 mA as opposed to an expectancy of 0 mA. Using absolute errors yet leads to problems when dealing with slots of different lengths, which is the case for the adaptive slotting scheme. To compare errors on the same statistical basis, we decided to only compute per-day errors.

7. Evaluation of Slot Representation

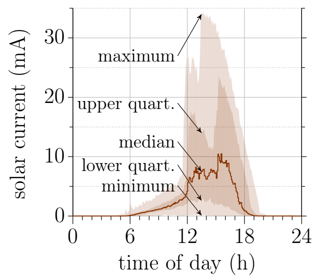
In this section we analyze adaptive slot distributions w.r.t. slot lengths, starting points, and representation errors.

7.1. Adaptation Performance

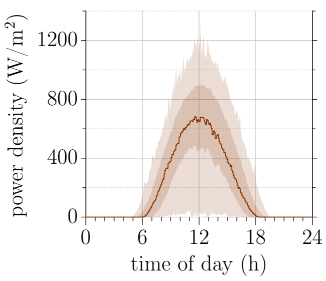
Figures 4a and 4d show a heatmap of the harvester output for the complete observation periods. The profile of the ECSU dataset is widely symmetric, days with high harvesting potential are dominant and the seasons of the year evident. In contrast, there are many bad days at the TUHH site and more dynamics. Harvesting potential is low in the mornings, caused by the shadow of the building, with a sudden increase depending on the seasonal altitude of the sun. The remaining subfigures show the corresponding slot distributions for 6 slots. They indicate that the adaptive scheme reproduces the days' cycle quite accurately by choosing slot lengths according to the harvesting potential. On days with low total potential, the scheme splits the first slot and merges slots in the middle of the day, e.g., starting at day 70 in Fig. 4b. Prominent peaks on consecutive days, e.g., at around day 50, cause a capturing slot rearrangement.

While this behavior is intended, it has the following side effects: As can be inferred from Fig. 4e at around day 270, the scheme shapes short slots in the afternoon, because these days have poor harvesting conditions in the morning but short periods of better conditions in the afternoon. The slot distribution adapts to an unusual pattern. However, the distribution recovers quickly on the following days. The memory-saving version with a restricted number of $C = 3$ split points is displayed in Fig. 4c and 4f. The overall behavior does not change significantly, but adaptations are expectedly less fine-grained and some split-operations with low influence are omitted due to fewer split points.

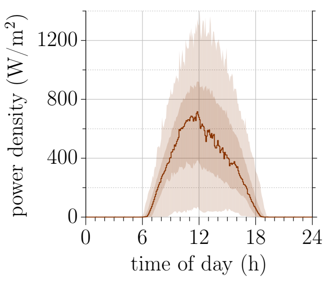
A detailed comparison of an adaptive and optimal distribution is depicted in Fig. 5. The adaptive scheme identifies the two peaks and traces their falling edges accurately. Only few slots are used for the periods of low harvester output in the morning and late night. The slot distribution yields close-to-optimal points for comparing actually harvested energy with the corresponding predictions. This is indicated by the optimal distribution (dashed curve). The latter is sensitive to noise (fourth cycle) and therefore overfitting w.r.t. the previous days. In conclusion, the



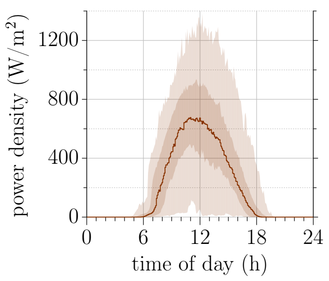
(a) TUHH



(b) ECSU

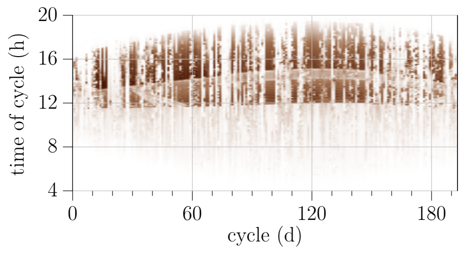


(c) LOLH

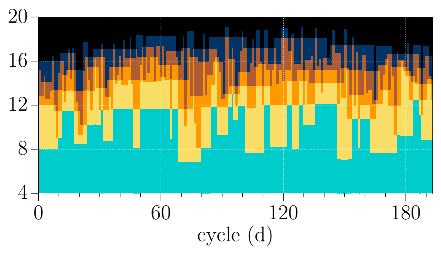


(d) SPMD

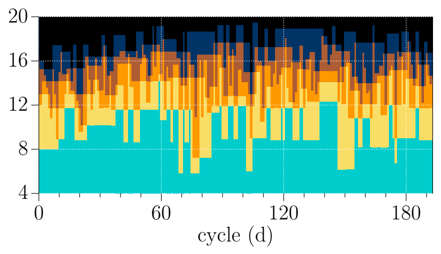
Figure 3: Statistical analysis of potential harvester output within the 5 min base intervals throughout a day



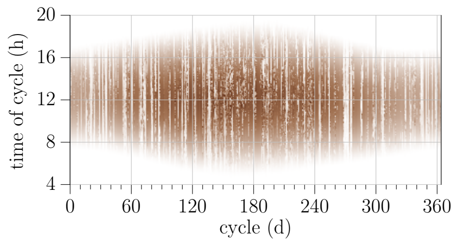
(a) TUHH, harvester output



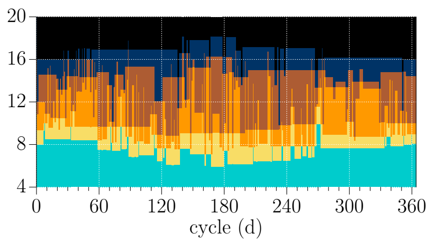
(b) TUHH, no C-constraint



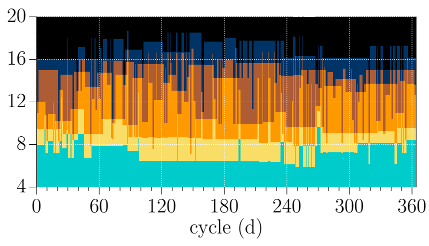
(c) TUHH, $C = 3$



(d) ECSU, harvester output



(e) ECSU, no C-constraint



(f) ECSU, $C = 3$

Figure 4: Adaptive slot distributions with one adaption per day ($B = 1$) and $S = 6$ slots. Left: heatmap of the harvester output per base interval (higher harvest for darker shades). Center/right: Slot distributions indicated by different shades

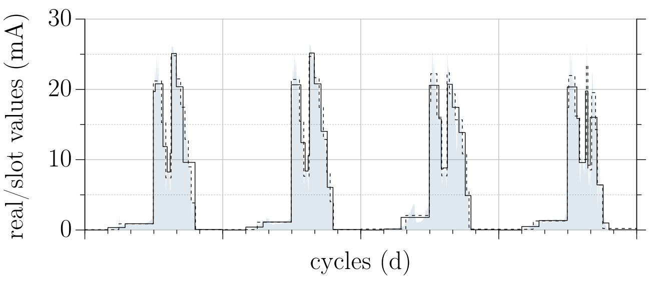
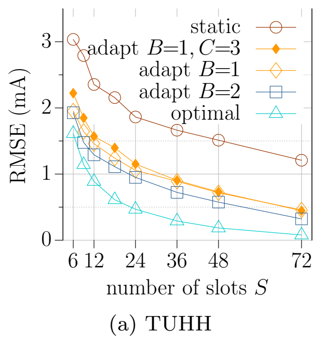


Figure 5: Slot representation of adaptive (solid) vs. optimal (dashed) slots. TUHH dataset with $S = 12$, $L_{\max} = 64$, $B = 1$, $C = 3$. The filled curve indicates the real course of the days

comparison shows that adaptive slots find a distribution that is close to the optimal one, while it is less sensitive to a single day.

7.2. Representation Error

Figures 6a and 6b depict the RMSE for the TUHH and ECSU datasets. The lowest error is naturally produced by the optimal slot distribution (cf. Sect. 3.3). The adaptive scheme performs in between the optimal and static distributions, having about the same distance to each of these. For a larger number of slots, the adaptive scheme largely outperforms the static distribution. The latter spends about half of its slots for periods of low or no harvester output, plus the adaptive scheme is able to adapt to the distinct course of the TUHH dataset. For the ECSU dataset, the performance of the adaptive scheme with few slots is close to the optimal distribution, since the dataset is mostly regular and less changing and can therefore be adapted easier. Both figures indicate that the gain of more than $B = 1$ split operation is marginal; even for the TUHH dataset, for which its impact is slightly higher. Restricting the adaptive scheme to $C = 3$ split points per slot is of low significance for the representation error (the curves are hardly distinguishable).



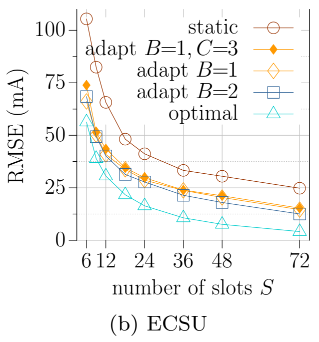
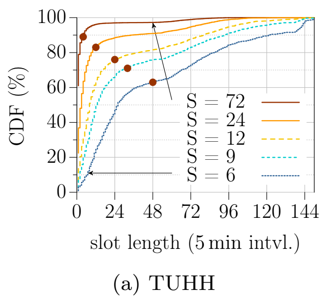


Figure 6: Representation error of slotting schemes



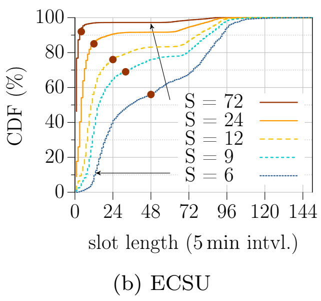


Figure 7: Cumulative distribution function of slot lengths for the optimal distribution. The filled dots indicate the slot lengths of the corresponding static distribution.

7.3. *Distribution of Slot Length*

The cumulative density function (CDF) of slot lengths for the TUHH and ECSU datasets using optimal distributions is portrayed in Fig. 7a and 7b. The majority of the slots is shorter than the ones used in a static setup (filled circles). This is compensated by a few slots with extreme lengths. Due to the long and event-less morning at the TUHH site, there are slots embracing as many as 144 base intervals or 12 h, respectively. There are no such long slots in case of the ECSU dataset. Virtually no slot is longer than 96 base intervals and for $S = 24$, only 5% of the slots exceed 72 base intervals. The reason is the brief period of the year with short days (winter) and the symmetry of the harvesting potential. Note that the CDF of slot lengths depends on the beginning time of the cycles: There could be one very long slots embracing the night. Since this may decrease adaptation performance (for a low C), having a maximum slot length is useful from a practical perspective.

The evaluation of slot lengths indicates that, for solar harvesting, it is sufficient to restrict slot lengths to a few hours in the presence of 12 to 24 slots. If there are too many slots, most of them cover one or two base intervals only, while a few slots cover the remainder of a cycle. Although the representation error is very low, there might be no benefit regarding long-term prediction quality due to the changes in between days (cf. Sect. 6.1).

8. Evaluation of Prediction Quality

In this section, we analyze the influence of adaptive slots on short- and long-term prediction accuracy and additionally evaluate the influence of the parameters of the prediction schemes. Unless otherwise noted, we use $B = 1$, $C = 3$, $L_{\text{min}} = 1$, $L_{\text{max}} = 64$, because these parameters gave satisfying results (cf. Sect. 7) and they guarantee a resource-efficient implementation on sensor nodes (cf. Sect. 4.4.4).

8.1. *Correlation between Cycles*

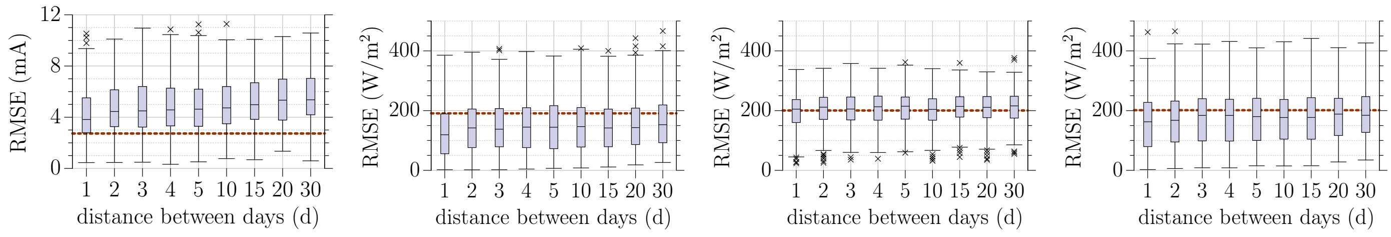
Prior to assessing the prediction quality of the schemes, we analyzed the correlation of days with respect to their distance. The results are shown in Fig. 8. The days of TUHH (Fig. 8a) exhibit the weakest correlation among the datasets. The median RMSE of 3.82 mA is large in comparison with the mean harvester output of only 2.72 mA. Since the RMSE correlates with day distance, more recent days have a higher impact on predicting future days. Figure 8b reveals that days of ECSU are stronger correlated. However, there are a few outliers, which are caused by changes from sunny to cloudy days and days with unstable weather conditions. The ratio of the median RMSE of 119 W/m^2 for two adjacent days and the mean harvesting output of 191 W/m^2 is considerably lower as for TUHH.

8.2. *Cycle Prediction with Static Slots*

Adjacent days have higher correlation and lower RMSE than those being further apart. Many prediction schemes thus calculate (weighted) averages of slot values to increase prediction accuracy. In this section, we compare the two methods presented in Sect. 5.1 and inspect their gain over simply using the slot values of the previous day for prediction.

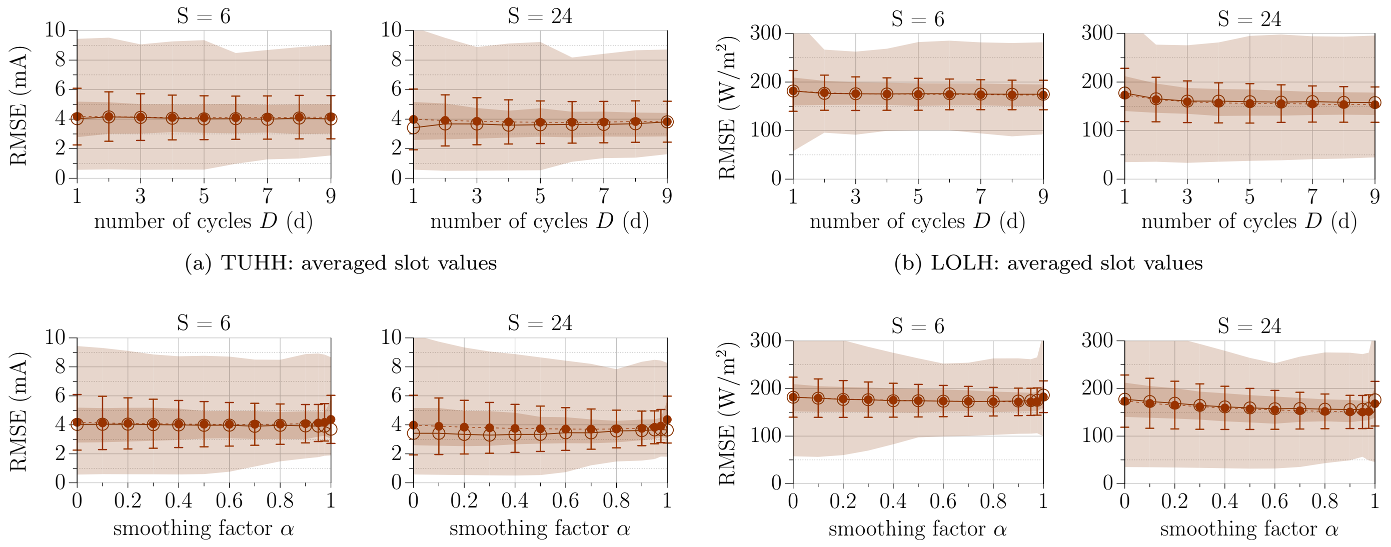
Figures 9a and 9b depict the error of predicting a full cycle by averaging each slot value of the previous D cycles. For the TUHH dataset (Fig. 9a), prediction accuracy is slightly affected by D only. Mean and median stay almost unaffected, but outliers in both directions are reduced. The mean prediction error is slightly above ($S = 6$) or below ($S = 24$) the RMSE of adjacent days. Hence, slotting does not infer a loss of prediction accuracy, and averaging slot values introduces a lower variance of accuracy. The situation is different for the LOLH dataset (Fig. 9b): averaging 3 days decreases the mean prediction error by roughly 20% and lowers its variance. Half of the improvement is due to the smoothing effect of slotting itself ($D = 1$ in the plot), which compensates for the non-cyclic variations in the afternoon.

Prediction accuracy of exponential smoothing is shown in Fig. 9c and 9d. A larger α may decrease the average (median) RMSE and deviation, while the sensitivity to α is low. This implies that choosing α not optimized for an environment or individual node does not have a large negative impact. This has practical relevance, for optimal settings are not likely to be known for every single node



(a) TUHH (b) ECSU (c) LOLH (d) SPMD

Figure 8: Statistical similarity analysis of individual days (days with a distance of 1 are adjacent). The boxplots show quartiles (boxes) and outliers (cross marks). The bold, dashed horizontal lines indicate the average harvester output



(a) TUHH: averaged slot values (b) LOLH: averaged slot values

(c) TUHH: exp. smoothed slot values (d) LOLH: exp. smoothed slot values

Figure 9: Prediction accuracy with static slots: Comparison between predictions obtained by averaging (upper row, cf. (21)) and exponentially smoothing (lower row, cf. (22)) among past cycles. The plots show minimum and maximum values (light shade), quartiles (darker shade), medians (filled circles), and mean values with standard deviation (filled circles with errorbars)

and its environment. Choosing α from the range $[0.6, 0.9]$ gives good results for all datasets (including ECSU and SPMD) considering the complete statistical distribution of RMSE values.

In conclusion, smoothing slot values exponentially is to be favored over averaging. The prediction accuracy is preserved or even improved, while memory consumption and computing power are heavily reduced. Smoothing in general brings an accuracy gain as compared to plainly relying on the slot values of the previous day. Parameter sensitivity is comparably low and the range of good parameter choices is similar for all datasets.

8.3. Cycle Prediction with Adaptive Slotting

Due to the results in Sect. 8.2, we analyzed the benefit of an adaptive slotting scheme using exponentially smoothed slot values. The results for the TUHH and LOLH datasets are shown in Table 2. For $S = 6$ the adaptive scheme yields a higher prediction accuracy: The mean RMSE decreases by up to 0.15 mA. Using 12 adaptive slots achieves the precision of 24 static slots. This observation is valid for all datasets. Having more than 12 adaptive slots does not increase prediction accuracy notably. The range of smoothing factors with best results is roughly the same as stated before.

A more detailed analysis of prediction accuracy regarding the behavior of the different schemes on a per-day basis yields additional insight. Figure 10 portrays the prediction error for an excerpt of days of the ECSU dataset. The plots indicate that the potential of using adaptive slots is higher, if the number of slots is equal (Fig. 10a). The comparison in Fig. 10b supports the finding, that adaptive slotting with 12 slots is on par with 24 static slots: All curves behave similarly, plus the error difference (lower row) is always close to zero. In most cases, both slotting schemes achieve a RMSE close to the corresponding, possible optimum, which is derived by calculating the slot values for the following day based on the slot distribution of the current one. However, there are outliers for both smoothing factors α, but at different times. If the weather changes

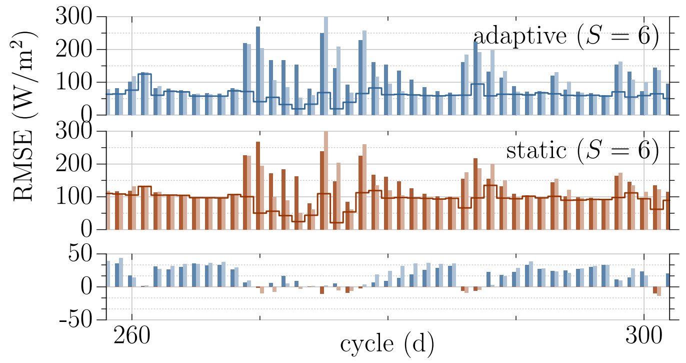
	S static slots				S adaptive slots			
α	6	12	18	24	6	12	18	24
0.0	4.17	4.01	4.02	3.98	4.09	4.01	4.05	4.07
0.2	4.10	3.90	3.89	3.84	3.97	3.85	3.86	3.88
0.4	4.06	3.83	3.81	3.76	3.91	3.76	3.75	3.76
0.6	4.04	3.80	3.76	3.71	3.89	3.71	3.69	3.69
0.8	4.05	3.80	3.77	3.71	3.92	3.72	3.69	3.68
0.9	4.08	3.85	3.80	3.76	4.00	3.77	3.74	3.74

(a) TUHH: mean RMSE (mA)

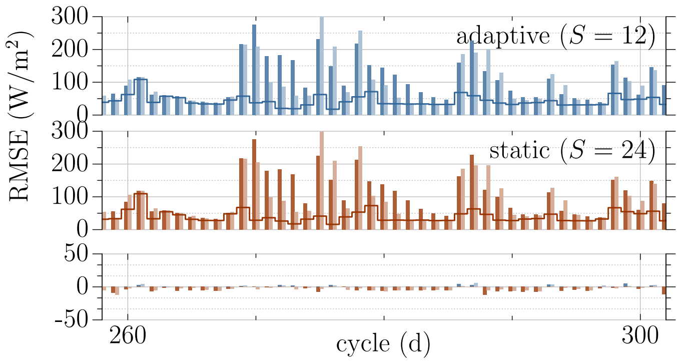
	S static slots				S adaptive slots			
α	6	12	18	24	6	12	18	24
0.0	181.8	172.4	172.6	173.4	174.7	176.2	180.0	182.5
0.2	177.9	166.3	165.0	165.1	169.4	167.8	169.7	171.3
0.4	175.3	162.1	159.7	159.2	166.0	161.9	162.3	163.2
0.6	173.5	159.1	155.9	155.0	163.9	157.8	157.0	157.2
0.8	172.2	157.0	153.2	151.9	163.8	155.5	153.2	152.8
0.9	171.8	156.2	152.1	150.7	165.9	155.8	152.0	151.3

(b) LOLH: mean RMSE (W/m^2)

Table 2: Prediction accuracy of static vs. adaptive slots with exponentially smoothed slot values ($L_{\min}=2$, $L_{\max}=128$ for $S=6$)



(a) ECSU: $S=6,\ L_{\min}=2,\ L_{\max}=128$



(b) ECSU: static ($S=24$) vs. adaptive ($S=12,\ L_{\max}=64$)

Figure 10: Per-day comparison of prediction accuracy using 1-adaptive (top) and static slots (middle). The difference of the scheme is shown in the bottom plot (values larger than 0 indicate a higher accuracy of the 1-adaptive approach). The plots show the performance with $\alpha=0.8$ (dark bars) and $\alpha=0.4$ (light bars). The solid step function indicates the possible optimum of the respective schemes by applying the slot distribution of one day to the following one.

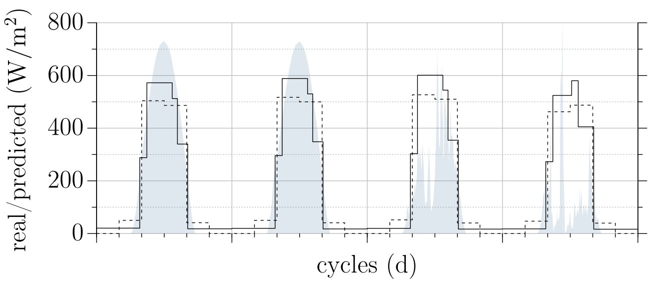
from one stable period to a different stable period—e.g., from sunny to cloudy days—a smaller α proves beneficial, as it allows for a quicker adaptation to the new conditions. If, in contrast, there only is one outlier day—e.g., a cloudy day within a series of sunny days–a large α yields better results, because the outlier has lower impact on the slot values. These large errors are an inherent problem of the prediction scheme, attenuating the benefits of the adaptive slotting scheme in a statistical evaluation by heavy outliers affecting the mean RMSE.

Slot distributions for the ECSU and TUHH dataset are shown in Fig. 11. They clearly illustrate how the adaptive scheme captures the symmetric curve of solar radiation, but also merges and splits slots to react to the changed conditions. The higher prediction accuracy of 6 adaptive vs. 6 static slots can be explained by the improved approximation of the curve, cf. Fig. 11a. Yet, caused by the larger variations in the middle of the day, the scheme arranges the first and last slot to embrace short times of non-zero radiation, leading to a non-negligible error at night. If more slots are used (Fig. 11b), this problem is no longer observed. The results of 24 static versus 12 adaptive slots are similar: Both approaches have (almost) the same resolution during the non-zero radiation phases of the days. The figures pinpoint the major problem of using a simple prediction based on smoothed slot values: There is no remedy for changed conditions, in which a large α prevents quick tracking.

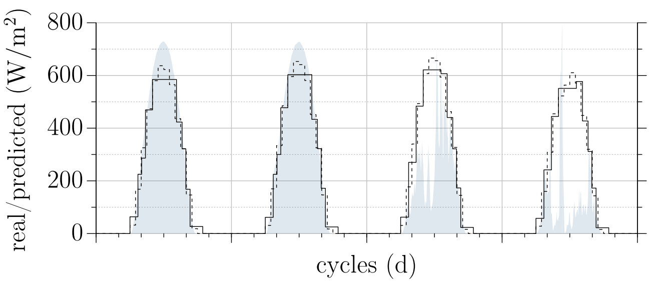
Distinct patterns profit from adaptive slots. The two peaks of the TUHH dataset are correctly identified and tracked (Fig. 11a, third cycle). Additional slots improve the course within the peaks (cf. Fig. 11b). The figures reveal that the adaptive scheme identifies the regions of interest—i.e., those with high but varying values. If applying a more advanced prediction technique than simply using smoothed slot values, this behavior can be turned into a higher return on investment. In particular, changed conditions, that are more likely to have a larger influence in periods of higher and varying harvester output, can be identified directly and with low latency. Thus, reactions on additional or less harvester output than forecasted can be undertaken more timely.

8.4. WCMA Short-Term Prediction

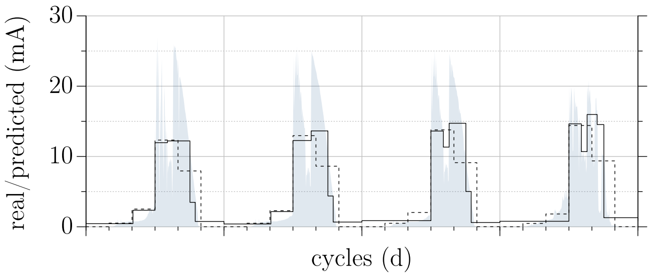
This section compares the accuracy of WCMA-based forecasts (cf. Sect. 5.2) using adaptive versus static slots. We also investigate the impact of the WCMA parameters.



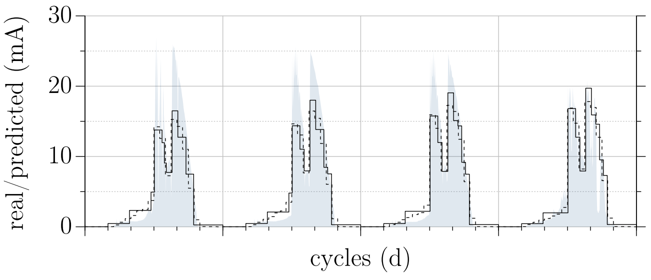
(a) ECSU, $S = 6,\ L_{\min} = 1,\ L_{\max} = 256$



(b) ECSU, $S = 24/12$ (static/adaptive), $L_{\max} = 64$



(c) TUHH, $S = 6,\ L_{\min} = 1,\ L_{\max} = 256$



(d) TUHH, $S = 24/12$ (static/adaptive), $L_{\max} = 64$

Figure 11: Comparison of predicted slot values for static (dashed) vs. adaptive (solid) slottings for $\alpha = 0.8$. The filled curve indicates the real course of the days

8.4.1. Influence of the Trend

The per-day RMSE of static slots for the TUHH dataset is illustrated in Fig. 12. The plots allow for several observations. Firstly, using the value of the previous slot ($\omega < 1$) does not lead to a significantly higher prediction accuracy; in contrast, relying on the trend only is favorable. Secondly, there is a positive influence of the trend scale Λ ($K \geq 1$), but the improvement of $K \geq 2$ is low (the open circle can hardly be distinguished from the filled one). Using $S = 6$ slots is an exception: the low number of slots gives almost no room for reacting to the course of the current day. These observations are valid for all datasets for similar values of α with the following restrictions. The RMSE for $\omega > 0$ increases for lower values of α, and for $\alpha < 0.5$ it is not beneficial to rely on the trend. For LOLH, trend-scaling is only advantageous for more than 12 slots. These results are comprehensible, because there is a limited correlation of the trend in adjacent slots: Closer and shorter slots tend to have a higher correlation. Yet, there are adjacent slots that exhibit almost no trend correlation, e.g., if one of the slots has a very low $\mu_{\mathbf{s}}$. In contrast, using $\bar{\mu}_{\mathbf{s}}$ of the current slot $\mathbf{s}$ to predict $\hat{\mu}_{\mathbf{r}}$ of the directly following slot $\mathbf{r}$ is not feasible, if the values of $\bar{\mu}_{\mathbf{s}}$ and $\hat{\mu}_{\mathbf{r}}$ are not similar. Particularly solar harvesting does not offer a similarity of adjacent slot values for a small S—e.g., cf. Fig. 11, where certain slot values $\mu_{\mathbf{s}}$ vary greatly.

8.4.2. Adaptive Slots

To compare the prediction accuracy of adaptive versus static slots, Table 3 lists mean RMSE values for WCMA based on exponential slot smoothings—the parameters and dataset are the same as before. The general behavior is similar, yet with notable exceptions. The profit of using the trend portion is larger for adaptive slots. Using values of $K > 1$ has a low but notable positive influence; the opposite is the case for static slots. Using a larger value of K generally increases prediction accuracy of adaptive slots, whereas it decreases the accuracy in many setups using static slots. A generic choice of this parameter is therefore simplified with adaptive slots. Moreover, prediction accuracy is improved, e.g., 12 adaptive slots (with $K = 3$) yield almost the same results as 24 static slots, when $\omega \geq 0.8$.

The impact of adaptive slots on other datasets is similar. The correlation of the trends among slots depends on the dataset and time of the day (cf. Sect. 8.4.1). For all datasets, 6 adaptive slots are preferable over the same amount of static slots for low smoothing factors. For $\alpha = 0.5$, adaptive slots outperform their static counterparts by more than 11% for the ECSU dataset. For $\alpha = 0.8$, both have similar RMSE values, but adaptive slots work well only for $K \geq 2$. For all other parameter setups, adaptive slots achieve better results, i.e., fewer adaptive slots are needed to achieve a result accomplished with static slots; thus less forecasts are required.

Trend scaling does not work well with 6 adaptive slots for the LOLH and SPMD datasets. For 12 slots, there is no accuracy gain using adaptive slots instead of static ones for LOLH, but a slight improvement for SPMD. An increasing number of slots leads to an accuracy gain for adaptive slots. Employing 24 slots with $\alpha = 0.9$ achieves a gain of more than 5% and 11% for 72 slots and the

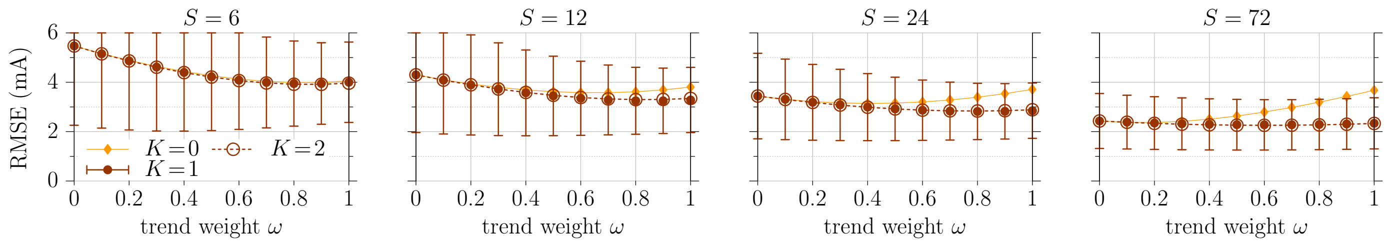


Figure 12: Influence of trend weight ω and K on prediction accuracy of WCMA. TUHH dataset with static slots and $\alpha = 0.8$

		K, static slots				K, adaptive slots			
S	ω	0	1	2	3	0	1	2	3
	0.0	5.47	5.47	5.47	5.47	5.51	5.51	5.51	5.51
	0.4	4.44	4.40	4.39	4.40	4.32	4.33	4.34	4.33
6	0.8	3.99	3.95	3.92	3.92	3.81	3.67	3.65	3.65
	0.9	3.99	3.95	3.92	3.91	3.83	3.62	3.59	3.60
	1.0	4.05	4.00	3.96	3.96	3.92	3.63	3.59	3.60
	0.0	4.30	4.30	4.30	4.30	4.13	4.13	4.13	4.13
	0.4	3.69	3.57	3.59	3.60	3.45	3.38	3.34	3.32
12	0.8	3.61	3.25	3.30	3.34	3.44	3.01	2.93	2.91
	0.9	3.69	3.25	3.30	3.36	3.55	2.99	2.91	2.90
	1.0	3.80	3.28	3.35	3.41	3.72	3.02	2.93	2.92
	0.0	3.44	3.44	3.44	3.44	3.03	3.03	3.03	3.03
	0.4	3.14	2.99	2.99	2.99	2.75	2.61	2.58	2.57
24	0.8	3.40	2.82	2.83	2.87	3.22	2.43	2.38	2.37
	0.9	3.54	2.82	2.85	2.90	3.44	2.43	2.38	2.38
	1.0	3.71	2.85	2.89	2.95	3.68	2.45	2.40	2.41
	0.0	2.43	2.43	2.43	2.43	2.03	2.03	2.03	2.03
	0.4	2.51	2.30	2.28	2.27	2.18	1.91	1.89	1.88
72	0.8	3.19	2.29	2.27	2.29	3.08	1.91	1.90	1.91
	0.9	3.43	2.31	2.30	2.33	3.37	1.93	1.93	1.95
	1.0	3.68	2.34	2.33	2.37	3.67	1.96	1.97	2.00

Table 3: Influence of ω and K on prediction accuracy of WCMA for TUHH dataset with $\alpha = 0.8$

LOLH dataset. The corresponding values for SPMD are 11% and 12%.

8.4.3. Impact of Slot-Value Smoothing

Smoothing slot values among cycles has an important impact on prediction accuracy. The highest impact, of course, unfolds for $\omega = 1$. Figure 13 shows the prediction accuracy for α, also comparing static and adaptive slot distributions. For both datasets, $\alpha \approx 0.9$ gives optimal results for static slots. While the slopes to the left of this point are shallow, a strong increase in RMSE is observable for a large value of α. It is thus advisable to rather choose a too small α than a too large one. The plots also show that an increase of S pays of better for small values of α (e.g., $S = 6, \ldots, 24$), whereas the gap between 24 and 72 slots is small compared to the tripled memory and computing demand.

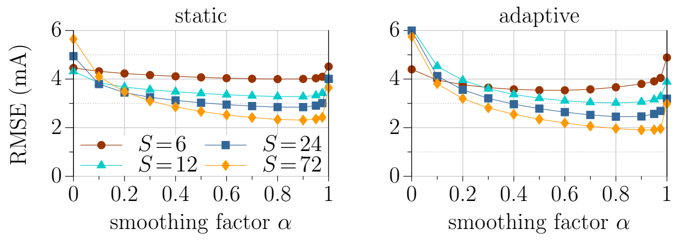
The prediction accuracy with adaptive slots is higher, since the scheme captures the more fluctuating regions of a cycle. Choosing the best smoothing factor α depends on S, while it is generally independent of the dataset. In general, the optimal α is greater for a larger number of slots. For 6 slots, $\alpha = 0.5$ is a good choice, while for 24 slots 0.85 to 0.9 yields the best accuracy. The reason why fewer slots need a smaller α is buried in the fact that split-and-merge operations target larger regions of a cycle, so that a smaller α is required to more quickly adjust to the new mean slot value.

8.5. Additional Observations

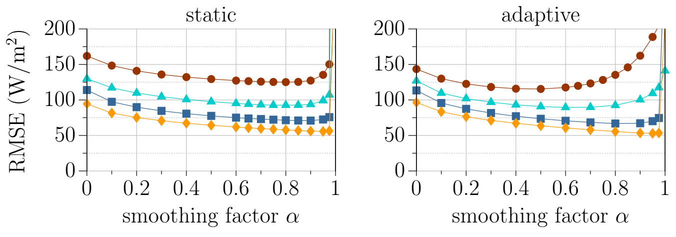
In the following, we provide additional results for possible modifications of WCMA.

Since using the value of the previous slot directly is of low improvement for WCMA prediction accuracy, we analyzed if it is beneficial to use an exponentially smoothing of previous samples $h[t]$ as, e.g., discussed in [17]. We replaced $\mu_{\mathbf{r}_0}$ in (23) with an exponentially smoothed value of recent $h[t]$. The results can be summarized as follows: Firstly, the mean RMSE was lower w.r.t. to using the value of the last slot. Secondly, there was a low improvement over trend-only prediction ($\omega = 1$), suggesting that a possible but uncertain accuracy gain is bought by introducing a new parameter. Thirdly, the benefit was larger for adaptive slots, caused by the (intended) lower inter-slot correlation. Thirdly, the results show a low sensitivity to the smoothing factor with a minimum close to 0.3 for the TUHH dataset.

The plots discussed in Sect. 8.4.1 suggest that prediction accuracy improves with an increasing number of slots S. However, this is only valid for short-term predictions, since an increased S decreases the prediction horizon. For very large values of S ($S \geq 72$) an average or exponential smoothing of recent values $h[t]$ gives the approximately same mean result as a WCMA prediction regardless of the choice of ω and K. Instead of using additional slots (and thus consuming more memory), performing multiple predictions per slot increases (short-term) prediction accuracy—e.g., 24 equidistant predictions using only 6 static slots produce a mean RMSE of roughly 3.2 mA for the TUHH dataset ($\alpha = 0.8$), which is an improvement of 0.8 mA over 6 predictions only. The error of less than 3.0 mA with 12 adaptive slots is yet not achieved. Using adaptive slots is also more efficient due to the lower number of forecasts and schedule updates.



(a) TUHH



(b) ECSU

Figure 13: Influence of smoothing factor α for trend-only WCMA prediction ($\omega = 1$, $K = 1$)

8.6. Recommendations for the Field

The extensive evaluation has enabled us to derive the key parameters of short- and long-term prediction for practical application on solar-energy-harvesting sensor nodes. These results are vastly independent of the node's location. Energy forecasting with a small number of slots is feasible. Using 12 adaptive slots is a good compromise between keeping a low-memory footprint and achieving improved prediction results for short and long prediction horizons. Smoothing slot values among cycles increases the prediction accuracy. An exponential filter has a lower parameter sensitivity than averaging values of recent days. At the same time, the prediction error is reduced in many setups while preserving memory. A filter coefficient of $0.6 < \alpha < 0.95$ gave optimal results with low variation for all datasets and $S = 12$. The short-term WCMA prediction scheme produced optimal prediction results for $\omega \approx 1$. To decrease the complexity of implementation, computation, and parameter-setup, we suggest using $\omega = 1$. Adaptive slots profit from a trend-scale Λ based on a history calculated from the trends of more than one recent slot. The results of our evaluation support that selecting $K = 2$ or $K = 3$ is a good choice in all setups.

9. Conclusion

Achieving energy-efficient, reliable, and sustainable operation of miniature energy-harvesting sensor nodes is coupled tightly with accurate forecasting of their future energy intake. To achieve this goal, it is not sufficient to generate a rough estimate of the cumulative energy per day. Nodes must be aware of their energy budget per time to derive optimal while depletion-safe task schedules. The former largely depends on the positioning of each individual sensor node. Relying on global information, such as forecasts from local weather stations in case of solar harvesters, is hence not sufficient. Determining static energy patterns prior to deployment is hardly feasible in large networks, nor can they adapt to unexpected changes in the environment and performance of the harvester. Forcing a harvester's cycle into a constant number of static slots cannot capture short periods of high harvester output while possibly wasting too many slots for periods of low interest, i.e., zero energy intake.

In this paper we have presented the first adaptive slotting scheme for energy-forecasting sensor nodes that identifies and traces the actual pattern of energy intake. By an extensive evaluation using real-world datasets, we have shown its ability to learn a harvester's pattern of energy production. We have integrated this novel scheme into well-known and established forecasting algorithms for solar-harvesting sensor nodes. Adaptive slots increase accuracy for both short- and long-term predictions while preserving a low memory footprint by taking advantage of unused memory and requiring few additional bytes only. Moreover, slots adapting to the harvester's energy pattern alleviate the sensitivity of prediction quality to the set of parameters of the prediction algorithms, hence enabling generic choices of parameter sets.

The results of this work demonstrate that improving the fundament of known forecasting techniques—the slot distribution describing the harvester's pattern of energy production—is more promising than refining the forecasting techniques based on half-baked slot distributions. This work shows that there is plenty of room for improving prediction quality by analyzing and tracing the energy-intake profile with low overhead. Our evaluation also reinforces that long-term prediction may gain from incorporating global information, such as cloud cover or snow warnings, since a node's view on the future is limited to its present and local perspective.

References

[1] P. Dutta, S. Dawson-Haggerty, Y. Chen, C.-J. M. Liang, A. Terzis, Design and Evaluation of a Versatile and Efficient Receiver-Initiated Link Layer for Low-Power Wireless, in: Proc. 8th Intl. Conf. on Networked Sensing Systems, SenSys, 2010.

[2] H. Lee, M. Wicke, B. Kusy, O. Gnawali, L. Guibas, Data Stashing: Energy-Efficient Information Delivery to Mobile Sinks through Trajectory Prediction, in: Proc. 9th Intl. Conf. on Information Processing in Sensor Networks, IPSN, 2010.

[3] X. Jiang, J. Polastre, D. Culler, Perpetual Environmentally Powered Sensor Networks, in: Proc. Intl. Symp. on Information Processing in Sensor Networks, IPSN, 2005.

[4] F. Simjee, P. H. Chou, Everlast: Long-Life, Supercapacitor-Operated Wireless Sensor Node, in: Proc. Intl. Symp. on Low Power Electronics and Design, ISLPED, 2006.

[5] V. Kyriatzis, N. S. Samaras, P. Stavroulakis, H. Takruri-Rizk, S. Tzortzios, Enviromote: A New Solar-Harvesting Platform Prototype for Wireless Sensor Networks, in: Proc. Intl.

Symp. on Personal, Indoor and Mobile Radio Communications, PIMRC, 2007.

[6] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, W. Hong, A Macroscope in the Redwoods, in: Proc. 3rd Intl. Conf. on Embedded Networked Sensor Systems, SenSys, 2005.

[7] L. Mottola, G. Picco, M. Ceriotti, S. Guna, A. Murphy, Not All Wireless Sensor Networks Are Created Equal: A Comparative Study On Tunnels, Transactions on Sensor Networks (TOSN) 7 (2).

[8] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, J. Anderson, Wireless Sensor Networks for Habitat Monitoring, in: Proc. 1st Intl. Wksp. on Wireless Sensor Networks and Applications, WSNA, 2002.

[9] D. Kruger, C. Buschmann, S. Fischer, Solar Powered Sensor Network Design and Experimentation, in: Proc. of the 6th Intl. Symposium on Wireless Communication Systems, ISWCS, 2009.

[10] P. Dutta, J. Hui, J. Jeong, S. Kim, C. Sharp, J. Taneja, G. Tolle, K. Whitehouse, D. Culler, Trio: Enabling Sustainable and Scalable Outdoor Wireless Sensor Network Deployments, in: Proc. 5th Intl. Conf. on Information Processing in Sensor Networks, IPSN, 2006.

[11] J. Taneja, J. Jeong, D. Culler, Design, Modeling, and Capacity Planning for Micro-solar Power Sensor Networks, in: Proc. 7th Intl. Conf. on Information Processing in Sensor Networks, IPSN, 2008.

[12] B. Zhang, R. Simon, H. Aydin, Energy Management for Time-Critical Energy Harvesting Wireless Sensor Networks, in: Proc. 12th Intl. Symp. on Stabilization, Safety, and Security of Distributed Systems, SSS, 2010.

[13] A. Eswaran, A. Rowe, R. Rajkumar, Nano-RK: An Energy-Aware Resource-Centric Operating System for Sensor Networks, in: Proc. IEEE Real-Time Systems Symp., RTSS, 2005.

[14] C. Moser, L. Thiele, D. Brunelli, L. Benini, Adaptive Power Management for Environmentally Powered Systems, IEEE Trans. Computers 59 (4).

[15] D. Brunelli, L. Benini, C. Moser, L. Thiele, Robust and Low Complexity Rate Control for Solar Powered Sensors, in: Proc. Conf. on Design, Automation and Test in Europe, DATE, 2008.

[16] J. Jeong, A Practical Theory of Micro-Solar Power Sensor Networks, Ph.D. thesis, EECS Department, Univ. of California, Berkeley (2009).
URL `http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-49.html`

[17] A. Kansal, J. Hsu, S. Zahedi, M. B. Srivastava, Powermanagement in Energy Harvesting Sensor Networks, Trans. on Embedded Computing Sys.

[18] C. Renner, J. Jessen, V. Turau, Lifetime Prediction for Supercapacitor-powered Wireless Sensor Nodes, in: Proc. 8th GI/ITG KuVS Fachgespräch "Sensornetze", FGSN, 2009.

[19] J. Recas Piorno, C. Bergonzini, D. Atienza, T. Simunic Rosing, Prediction and Management in Energy Harvested Wireless Sensor Nodes, in: Proc. 1st Intl. Conf. on Wireless Communications, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology, VITAE, 2009.

[20] M. Ali, B. Al-Hashimi, J. Recas, D. Atienza, Evaluation and Design Exploration of Solar Harvested-Energy Prediction Algorithm, in: Proc. Conf. on Design, Automation and Test in Europe, DATE, 2010.

[21] C. Bergonzini, D. Brunelli, L. Benini, Comparison of Energy Intake Prediction Algorithms for Systems Powered by Photovoltaic Harvesters, Microelectronics Journal.

[22] C. Moser, L. Thiele, D. Brunelli, L. Benini, Robust and Low Complexity Rate Control for Solar Powered Sensors, in: Proc. Conf. on Design, Automation and Test in Europe, DATE, 2008.

[23] U.S. Department of Energy, National Renewable Energy Laboratory (NREL), Measurement and Instrumentation Data Center (MIDC), http://www.nrel.gov/midc (2011).

Christian Renner is a Ph.D. student advised by Prof. Volker Turau at Hamburg University of Technology. His research interests include applications of networked sensing and embedded systems, focussing on energy harvesting and budgeting. He received his Diploma degree from Hamburg University of Technology in 2008. He expects to finish his disseratation in 2012.

Volker Turau is professor for distributed systems at Hamburg University of Technology in Germany. Since 2008 he is the head of the Institute of Telematics. He received a Ph.D. in 1984 from the University in Mainz and held post doc positions at the Universities of Manchester and Karlsruhe. After that he was a visiting scientist at the University of Berkeley in California and Hewlett-Packard Labs in Palo Alto. His general research interest falls within the scope of distributed systems with a focus on wireless sensor networks, energy aware computing, and fault tolerance. He has published five books and more than 50 scientific papers in international journals and conferences. He regularly serves as a member of program committees of international conferences.