

A Linear Regression Data Compression Algorithm for an Islanded DC Microgrid

Ibrahim A. Bello^{1*}, Malcolm D. McCulloch² and Daniel J. Rogers²

¹Dyson Technology Ltd., Bristol, BS1 5TE, United Kingdom.

²Department of Engineering Science, University of Oxford, OX1 3PJ, United Kingdom.

*Corresponding author(s). E-mail(s): ibrahim.bello@ieee.org;
Contributing authors: malcolm.mcculloch@eng.ox.ac.uk;
dan.rogers@eng.ox.ac.uk;

Abstract

The exchange of data between energy stakeholders will play an important role in future smart energy systems. A key component of smart energy systems is the smart meter, which enables the utility provider to obtain energy consumption readings of customers at regular intervals. When smart meters are fully deployed, it is expected that several billions of data points will be generated per annum, which makes it necessary to compress the data to a suitable size, while preserving vital energy information. In this paper, we present a technique for data compression, which jointly compresses current and voltage time series data by substituting data points with linear regression coefficients. Given a set of input parameters, we demonstrate that the algorithm can also be invoked iteratively to autonomously improve the compression result. We apply the algorithm to load profiles obtained from an islanded DC microgrid laboratory experiment, and we demonstrate that our technique has a near-lossless compression performance and a high compression ratio of more than 50-to-1 for most of the datasets considered. As a proof of concept, we also apply the algorithm to energy data from an AC-grid-connected household and the results suggest that the developed algorithm is potentially applicable to more conventional energy systems. We compare our results with a similar linear-regression-based algorithm and our proposed technique demonstrates a better energy error performance on average for a given compression ratio and a 30-fold reduction in the data compression time.

Keywords: Microgrid, Smart grid, data compression, load profile, energy measurement

1 Introduction

This paper proposes a novel linear-regression-based algorithm for the compression of energy load profile data for an islanded DC microgrid system, which we presented in [1]. The algorithm is validated using two datasets comprising load profiles from a laboratory DC microgrid experiment and a residential household using conventional AC power supply.

This research is motivated by the increased role of data in energy systems which has been highlighted in several works. Data in energy systems has several uses such as forecasting, demand-side response, anomaly detection, and system planning [2]. However, as energy systems become more digitised, energy data is expected to pose communications, storage and processing challenges [3–6].

In one study [7], the full deployment of smart meters in the UK using a 30 min resolution for 27 million households is expected to generate up to 500 billion data points per annum. However, various studies have identified the need for even higher resolutions in the order of seconds to obtain more fine-grained control and insight into energy systems [8, 9]. If higher resolutions are employed, the volume of the data will be significantly increased, and this has motivated several works in data compression algorithms. Data compression could be lossless or lossy, but the latter is acceptable in cases where not all data points are relevant [10]. Furthermore, generic lossless compression algorithms (e.g. bzip2 [11] and DEFLATE [12]) tend not to achieve the high compression ratios offered by lossy and more domain-specific compression algorithms [13].

In [14], a K-length running length encoder (RLE) was proposed, which substitutes values within a certain precision, determined by the user, with a single value. This technique was used to compress temperature data collected by a wireless sensor network. In [13], a linear regression technique was employed, where a straight line was fitted to a time series of measured and simulated current and voltage data. When a point deviates from the value predicted by the line, within a predefined number of standard deviations, the algorithm restarts a new line segment.

In [15], a compression technique was proposed in which three polynomials of increasing orders are used in a piecewise manner to fit a segment within a time series. If a polynomial compresses the segment to within a certain predetermined deviation constraint, the coefficients of the polynomial are saved; otherwise, the algorithm tries other polynomials and the polynomial with the least mean square error (MSE) is selected. In [16], a similar strategy to [15] was adopted, but in this case, all polynomials from the provided options are used for each point in the series, and the coefficients of the polynomial with the smallest MSE are selected. In [10], a set of compression algorithms comprising the proposed algorithm in [13], singular value decomposition (SVD) and wavelet transform [17] are used to compress grid impedance measurements. Although the use of several curve fitting models can improve the energy error, it nevertheless increases the computational time and memory costs of the data compression algorithm.

The proposed algorithm in this paper will be deployed to a single-board computer (SBC), which will operate remotely in an islanded microgrid, and will transmit the compressed load profile data from several connected homes over a mobile network to a central server. The SBC will oversee the data collection and compression for tens to hundreds of homes; as such, a low complexity compression scheme is required to improve the scalability of the algorithm in terms of the communications latency and the cost of Internet data subscription. To this end, we have identified a number of desirable features that the data compression algorithm should possess as follows:

1. Low energy error: The algorithm should have a sufficiently low energy error in order to reconstruct the data as accurately as possible at the server, where the energy error measures the disparity between the energy of the original data and the compressed data.
2. High compression ratio: A high compression ratio is mandatory to reduce the communications costs of the system. If the compression ratio is too small, the use of the algorithm will not be justifiable.
3. Short computation time: The data compression algorithm should finish execution within a reasonable period of time so as not to overburden the SBC when the microgrid size scales up.

To achieve the aforementioned objectives, we employ a similar linear regression strategy as [13], which avoids the use of more complicated curve-fitting models as used in [15] and [16]. Simple linear regression is an attractive option for energy load profile data since load profiles typically present data points with small variations which could be substituted with one or more line segments.

The main objectives of the paper are as follows:

1. Implement a data compression algorithm using linear regression and compare the results in terms of the energy error, compression ratio and computation time with the results of [13]. The algorithm will avoid the use of a single global constant, such as was used in [14], which could lead to a poor energy error if the dataset consists of loads of significantly varying magnitudes.
2. Develop a technique where the energy error, as well as the compression ratio, can be improved iteratively depending on a maximum energy error target specified by the user. This is unlike previous works which focus on minimising the energy error only.
3. Analyse the impact of rounding errors on the linear regression results and devise a mitigation strategy which will make the linear regression more robust to rounding.
4. Apply the algorithm to load profiles obtained from a DC microgrid laboratory experiment and load profile data obtained from a UK AC residential household.

The rest of the paper is organised as follows. In Section 2, we describe the proposed algorithm. In Section 3 we describe our laboratory experiment for generating the dataset considered in this study. In Section 4, we present

and discuss the results of the data compression. The paper is concluded in Section 5.

2 Data Compression Algorithm

In this paper we consider the case of a microgrid [18] comprising a utility provider, data aggregator, and energy consumers each equipped with a smart meter. A microgrid setup is shown in Fig. 1 where the data aggregator, or “hub”, acts as both the electricity generator and data aggregator. Such *islanded* microgrid setup has been described in [1]. At intervals, the connected homes send their load profile data to the hub and this is used for computing the energy consumption of each home and performing other real-time control operations. The collected data will then be transmitted over a mobile network to the utility provider. To reduce the data communications cost, data compression is applied at the hub before transmission. Data compression can also be useful in reducing the energy consumption of the transmission [19]. It should be noted that data compression could be applied at the customer-end as well (e.g., [20]); however, in this case study, the microprocessor at the customer-side has limited computational and memory resources and is dedicated to providing time and safety-critical power electronics functionalities, such as battery cell balancing, and DC-DC conversion. The data compression algorithm is described in more detail in the following sections.

2.1 Data Compression using Linear Regression

Consider a series of current data points of length N collected from an individual home as follows:

$$\mathbf{i} = \{i_1, i_2, \dots, i_N\}, \quad (1)$$

corresponding voltages,

$$\mathbf{v} = \{v_1, v_2, \dots, v_N\}, \quad (2)$$

sampled at times,

$$\mathbf{t} = \{t_1, t_2, \dots, t_N\}, \quad (3)$$

where i_n and v_n represent the current and voltage measured at the n th time sample, t_n , respectively. It is desired to compress both \mathbf{i} and \mathbf{v} to a smaller size to reduce the communications cost from the data aggregator to the utility provider. In [13], the compression begins with an initialisation step where n_0 points are selected and linear regression is performed to derive the coefficients, b_0 and b_1 , where the value of the compressed voltage or current at a time, t , is given as:

$$y_t = b_0 + b_1 t. \quad (4)$$

The linear regression is performed using the least squares estimate which computes the coefficients as follows [21]:

$$b_1 = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2}$$

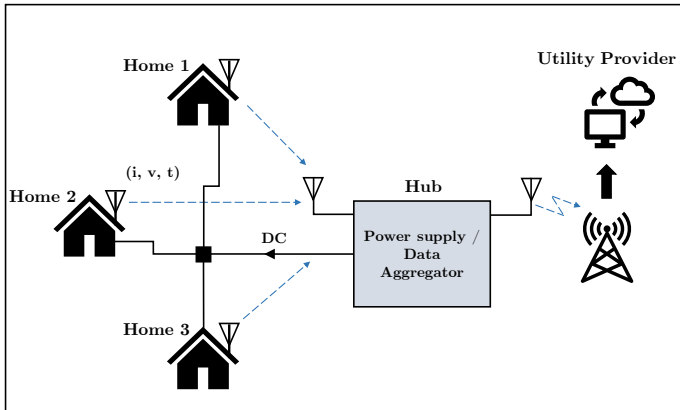


Fig. 1: Microgrid setup showing centralised hub and connected homes

$$b_0 = \bar{y} - b_1 \bar{x},$$

where x and y represent the independent and dependent variables respectively.

The standard deviation of the n_0 points is also computed, which is an estimate of how well the line fits the data. After the initial n_0 points are fitted, subsequent points are estimated using (4). If the absolute error between the estimated and actual values is greater than a number of standard deviations, m , then a new line is started, otherwise, the new point is included in the line segment and the standard deviation is updated. All the points that fit on the line are represented by a single tuple $(t_{\text{start}}, b_0, b_1, \sigma)$, where t_{start} is the first time sample in the line segment, and σ is the standard deviation and provides an estimate of the accuracy of the fitting. To recover the signal, the decompression algorithm simply enumerates the time samples in (3) by assuming a fixed time step, Δt , between the data samples, and applying (4) at every iteration.

2.2 Proposed Data Compression Algorithm

In the previous section, we provided a background of data compression using linear regression as proposed in [13]. One flaw of that approach is that the initial n_0 points may contain data points belonging to different loads or even to regions that do not correspond with an actual load reading (i.e. noise), which will make the constructed line fit less accurately to the data points. In our proposed method, this initial fitting is eliminated; instead, the linear regression is only applied to groups of points, \mathbf{g} , that are determined to be sufficiently close together. The premise is that for a sufficiently small value of Δt , successive data points will be highly correlated. The point, i_n , is considered to belong to the same *group* as i_{n-1} , i.e. $g(i_n) = g(i_{n-1})$, if the absolute deviation between

Algorithm 1 Proposed data compression algorithm

```

1: function MLRA(t, i, v,  $B$ )
2:    $n \leftarrow 1$ 
3:    $m \leftarrow 1$ 
4:    $\mathbf{g}_1^{(i)} \leftarrow \{i_1\}$ 
5:    $\mathbf{g}_1^{(v)} \leftarrow \{v_1\}$ 
6:    $\mathbf{g}_1^{(t)} \leftarrow \{t_1\}$ 
7:
8:   while  $n < N$  do
9:     if  $g(i_n) = g(i_{n+1})$  then
10:        $\mathbf{g}_m^{(i)} \leftarrow \{\mathbf{g}_m^{(i)}, i_{n+1}\}$ 
11:        $\mathbf{g}_m^{(v)} \leftarrow \{\mathbf{g}_m^{(v)}, i_{n+1}\}$ 
12:        $\mathbf{g}_m^{(t)} \leftarrow \{\mathbf{g}_m^{(t)}, t_{n+1}\}$ 
13:     else
14:       if  $|\mathbf{g}_m^{(i)}| < 2$  then
15:          $b_{0,m}^{(i)}, b_{1,m}^{(i)} \leftarrow i_n, 0$ 
16:          $b_{0,m}^{(v)}, b_{1,m}^{(v)} \leftarrow v_n, 0$ 
17:       else
18:          $(b_{0,m}^{(i)}, b_{1,m}^{(i)}) \leftarrow \text{linereg}(\mathbf{g}_m^{(i)}, \mathbf{g}_m^{(t)})$ 
19:          $(b_{0,m}^{(v)}, b_{1,m}^{(v)}) \leftarrow \text{linereg}(\mathbf{g}_m^{(v)}, \mathbf{g}_m^{(t)})$ 
20:       end if
21:
22:        $m \leftarrow m + 1$ 
23:        $\mathbf{g}_m^{(i)} \leftarrow \{i_{n+1}\}$ 
24:        $\mathbf{g}_m^{(v)} \leftarrow \{v_{n+1}\}$ 
25:        $\mathbf{g}_m^{(t)} \leftarrow \{t_{n+1}\}$ 
26:     end if
27:
28:      $n \leftarrow n + 1$ 
29:   end while
30: end function

```

them is within a certain percentage of i_{n-1} , expressed as follows:

$$|i_n - i_{n-1}| \leq \frac{B}{100} i_{n-1}, \quad (5)$$

where B is taken as a number between 0 and 100. More generally, $g(i_j) = g(i_k)$ if

$$g(i_n) = g(i_{n+1}), n = j, j + 1, \dots, k - 1,$$

where $j < k \leq N$. This procedure converts the currents defined in (1) to a list of M groups as follows:

$$\mathbf{g}^{(i)} = \{\mathbf{g}_1^{(i)}, \mathbf{g}_2^{(i)}, \dots, \mathbf{g}_M^{(i)}\}.$$

At the same time, corresponding groups for (2) and (3), $\mathbf{g}^{(v)}$ and $\mathbf{g}^{(t)}$ respectively, are constructed. This check is continued for each point in the series until the condition no longer holds true after which, linear regression is performed for the most recent group that has been obtained.

If B is 0, then it means that the data point has remained constant over the time period, Δt ; whereas, if the value is 100 it means that the data point has deviated by 100% at the next time step. We note that a higher value than 100 will imply a very high deviation between the consecutive data points which could have a negative impact on the compression performance if linear regression is applied. If the value of B is small, the compression ratio tends to be small as more groups will be formed, and thus more line segments will be created. On the other hand, if the value is too large more data points will be admitted into a group, leading to a large compression ratio, but at the expense of a higher compression error.

When the algorithm completely processes all the points in the series, the coefficients of the line so formed are saved. The output of the algorithm is a list of tuples, each consisting of five variables as follows: $(t_{\text{start}}, b_0^{(i)}, b_1^{(i)}, b_0^{(v)}, b_1^{(v)})$, where t_{start} is equal to the first time point in $\mathbf{g}_1^{(t)}$, and $(b_0^{(i)}, b_1^{(i)})$ and $(b_0^{(v)}, b_1^{(v)})$ are the intercept and slope of the current and voltage data samples respectively. The last time data point in the original series, t_N , is also appended to the list to assist in the decompression process. The algorithm requires at least two data points to perform the linear regression. If only one point is in a group, then $b_0^{(i)}$ is simply set to i_n , and $b_1^{(i)}$ is set to 0. The algorithm is shown in Algorithm 1, where $b_{0,m}^{(i)}$ and $b_{1,m}^{(i)}$ represent the linear regression coefficients for the m th data group, $\mathbf{g}_m^{(i)}$.

A minimum current value (computed as the ratio of P_{\min} to V_{load}) is defined such that all currents less than this are fitted onto the same line segment, where P_{\min} is the minimum load power, and V_{load} is the load supply voltage. This prevents (5) from being applied to noise values that do not correspond to a legitimate load. Like B , the choice of P_{\min} affects the performance of the data compression. If $P_{\min} = 0$, then all points will be subjected to the check in (5). However, if P_{\min} is larger than the computed power of all the data points, then all the data points will be fitted onto a single line segment.

Line 9 of Algorithm 1 checks if two consecutive points belong to the same group based on (5), while `linereg` computes the linear regression of its first argument with respect to the second. The worst-case complexity of the algorithm occurs when the linear regression is computed for all the points in the series, that is, $g(i_1) = g(i_N)$, which is $\mathcal{O}(N)$. Meanwhile, the algorithm incurs only an $\mathcal{O}(1)$ space complexity since only two points are compared at any

given time irrespective of the data size. It should be noted that although the algorithm is described for a bivariate time series, it is easily applicable to a time series with only one variable without any significant modification. Subsequently, the linear regression-based algorithm in [13] will be referred to as LRA, while the modified algorithm proposed in this paper will be referred to as MLRA.

2.3 Number Representation

After the data is compressed at the hub, it is transformed using JavaScript object notation (JSON) and transmitted via MQTT [22] to the utility provider. Any extra transmitted digit in the compressed result increases the number of bytes to transmit, and thus, potentially increases the communications cost. As such, it is necessary to round off the results returned in line 18 of Algorithm 1 to a suitable number of decimal points before transmission.

However, because the linear regression is carried out with respect to the time in seconds, which could take comparatively large values (up to 86,399 in a 24-hour period), the slope (i.e. the b_1 coefficient in (4)) could often become very small, since the dependent variable (i.e. the current), is often small in comparison to time. On the other hand, the intercept, b_0 , tends to be of similar magnitude as the current and is thus more robust to rounding errors.

If b_1 is small and is rounded off to zero, some useful information could potentially be lost as the $b_1 t$ term in (4) is not insignificant at large time values.¹ Furthermore, if all numbers after the decimal point are transmitted, several redundant zeros will be transmitted, which unnecessarily increases the bandwidth of the system. To avoid this, we multiply b_1 by a suitable scaling factor, F , prior to rounding. During the decompression, the same scaling factor is used to descale b_1 back to its actual value.

Although data values at the measurement sites are taken as the outputs of analogue-to-digital converters (ADCs) and are hence integers, the case of floating-point values was considered in this paper. However, the algorithm will still work on raw integer values, which could be useful in reducing rounding errors due to floating point operations [23]. Nevertheless, the outputs of the linear regression will still be floating point, which will be the values to be transmitted to the utility provider. If integer values are used, then the scaling operation described here may no longer be necessary as the integer outputs of the ADC will typically be numerically larger than the converted floating point values. For example, a 16-bit ADC will have a full-scale integer value of 65,535 for a current or voltage signal that will be much smaller in comparison. However, since the data compression is carried out at the data aggregator in our case study, transmitting the raw integer values may increase the bandwidth

¹Consider a scenario where $b_1 = 1e^{-5}$ and $t = 50,000$ s. If b_1 is rounded to four decimal places, then b_1 will be equal to zero, and as much as 0.5 A ($b_1 t$) of data would be lost as a result. Using $F = 10000$, only two digits will be transmitted compared to the six digits that would be transmitted without any scaling.

requirement at the measurement sites if the data is transmitted on a character-by-character basis. Furthermore, in a heterogeneous smart energy space, the use of integer values will require the data aggregator to have knowledge of the binary encoding and ADC data width of each smart meter which may not be practical.

2.4 Performance Metrics

Several metrics have been devised for determining the goodness of fit of estimated data [24, 25]. These metrics are typically applied to forecasting where the forecasted values are compared point-by-point to the measured values. In our case study, the original and decompressed data typically have unequal number of points due to the variable resolution of the original dataset as a result of the wireless data collection. As such, conventional point-wise metrics such as the mean absolute error and root mean square error is not suitable for this case study.

We compare the original and decompressed dataset using the “energy error”, e , defined as follows [13]:

$$e = \frac{|E_{\text{original}} - E_{\text{compressed}}|}{E_{\text{original}}} \times 100\%, \quad (6)$$

where E_{original} is the energy of the original data and $E_{\text{compressed}}$ is the energy obtained after decompressing the compressed data. The energy is approximated using the Trapezium rule as follows:

$$E = \int_{t_{\text{start}}}^{t_{\text{stop}}} P(t) \delta t \approx \sum_{n=1}^N \frac{(P_n + P_{n-1})}{2} \Delta t_n, \quad (7)$$

where $P(t)$ is the continuous power function, $\Delta t_n = t_n - t_{n-1}$, and P_n is a discrete sample of $P(t)$ at time sample t_n and is computed as $i_n v_n$. To compute the compression ratio, r , we use the following [26]:

$$r = \frac{\text{Original data size}}{\text{Compressed data size}}, \quad (8)$$

where the sizes of the original and compressed data are expressed in bits. In our case study, the compressed result of the proposed algorithm is a tuple comprising the time and the b_0 and b_1 coefficients of the current and voltage points. We exclude the standard deviation, σ , from the compression ratio computation. Assuming each floating-point variable is represented using the same number of bits, we can approximate (8) as the ratio of the number of the measurements in the original and compressed data sets, where the original data is a list of tuples each with three variables, (i, v, t) , and the compressed result each has

five values (see Section 2.2) per group. We can therefore approximate (8) as:

$$r = \frac{3N}{5M + 1}, \quad (9)$$

where N is the length of the original dataset and M is the number of groups formed by the proposed algorithm. We add 1 to the length of the compressed data to account for the last time point, t_N , appended to the compressed series.

2.5 Iterative Data Compression Algorithm

In (5), it is clear that the choice of B could have a significant impact on the performance of the algorithm. Since the choice of B is not known prior, the algorithm will need to be run several times manually for different values of B which is not suitable for unsupervised and autonomous operation, especially in a remote microgrid. Thus, an iterative implementation is desirable where the algorithm can try different values of B without human intervention.

If B is large, then the compression ratio is likely to be high as well, while if B is small, then the algorithm will discriminate even small deviations between two points, which may result in a better energy error. If we start with an initial value for B , we could improve both the energy error and the compression ratio by iteratively invoking the algorithm with new values for B such that both the compression ratio and the energy error are optimised.

To invoke the algorithm iteratively, we define a minimum acceptable target energy error, ϵ , and a maximum number of iterations, J , and step size, ΔB , by which B is increased or decreased, and an initial value for B , B_0 . If $e \leq \epsilon$, then B is altered such that the compression ratio would be increased during the next iteration. If $e > \epsilon$, B is altered in a direction that tries to reduce the energy error towards ϵ . The direction in which to alter B is determined by comparing the energy errors and compression ratios of the results of two successive invocations of the algorithm. We define the figure of merit (FOM) as follows:

$$\text{fom}(s_j) = \begin{cases} r_j, & \text{if } e \leq \epsilon \\ \frac{1}{e_j}, & \text{otherwise,} \end{cases} \quad (10)$$

where s_j , e_j and r_j are the results of the data compression and the corresponding energy error and compression ratio at the j th iteration respectively. We pay a penalty by incurring extra memory to store the result of the current call to the MLRA as well as the best result found so far. Division-by-zero error in (10) is avoided by assigning a very large value to the FOM whenever $e_j = 0$.

The flowchart for the iterative algorithm is shown in Fig. 2, where z indicates the direction of the stepwise change to B and s_{best} represents the result with the best FOM found so far. The initial value of z is selected as +1 if the first solution found satisfies the energy error requirement. This is not a critical choice; however, it relaxes the error bound in (5), such that the compression ratio can be increased, while not violating the energy error constraint.

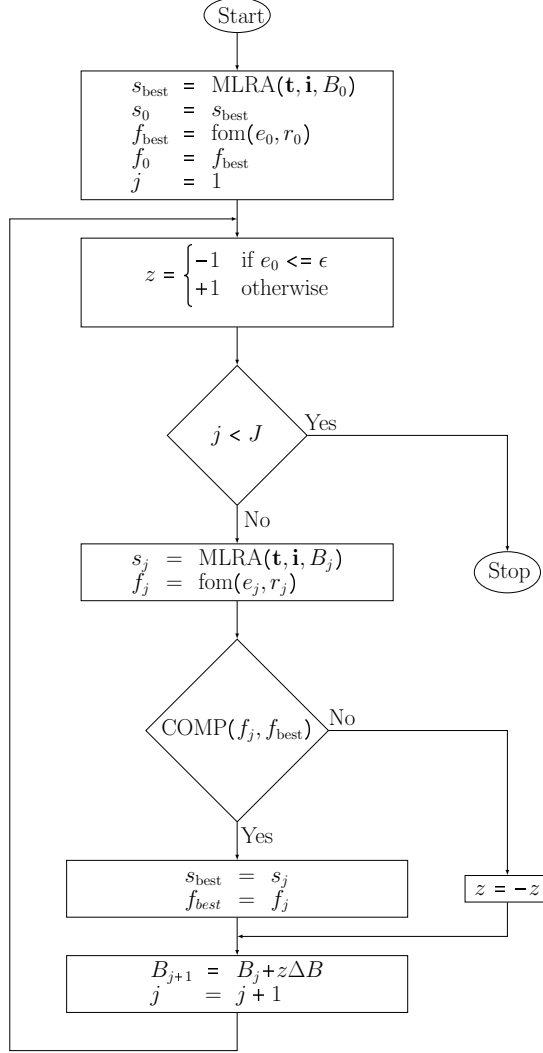


Fig. 2: Iterative implementation of the proposed algorithm

On the other hand, if the first solution does not satisfy the energy error requirement, z is chosen as -1 , which reduces the value of B at the next iteration. The comparison between the FOM of the current solution and the best solution found so far is performed using the function **COMP**, which returns *true* if its first argument has a better FOM than its second argument.

If both the current solution and the best solution have $e \leq \epsilon$, then the energy errors are compared. If both solutions have $e > \epsilon$, then the compression ratios are compared. If the current solution has $e > \epsilon$, while the best solution found so far has $e \leq \epsilon$, then the best solution is not updated. This check is done so as not to inadvertently compare the energy error and the compression

ratio directly, and also to give preference to the solution with the smaller energy error. If the FOM of the current situation is worse than that of the best solution found so far, z is toggled, which allows the algorithm to try a different direction.

3 Laboratory Experiment

We performed a laboratory experiment consisting of a power supply “home-box” which communicates its power consumption over a 433 MHz LoRa wireless link to the serial port of a Beaglebone Black SBC [27]. The “home-box” provides 12 V DC power over cigarette lighter ports to user-connected loads. This is illustrated in Fig. 3, showing the homebox, which consists of an STM32 microcontroller, an INA233-based current and voltage measurement board, and an RF module for transmitting the measurements to the hub. The measurement board acquires (i, v) readings from each port and communicates these to the microcontroller via an I²C interface. A more detailed description of the homebox has been provided in a separate paper [1]. The experiment was conducted over six hours with the homebox connected to one 50 W DC bulb, one 20 W DC bulb, one 4.6 W DC bulb, and one AC 45 W table fan connected to the homebox via an inverter.

An ESP32 microcontroller was used to randomly switch the loads by controlling 12 V relays connected to each port of the homebox. To make the generated load profiles realistic, a switching probability of 40% was assigned to each of the loads with minimum and maximum ON times of 10 m and 1 h respectively. After a load has been switched off, it is kept off for a time

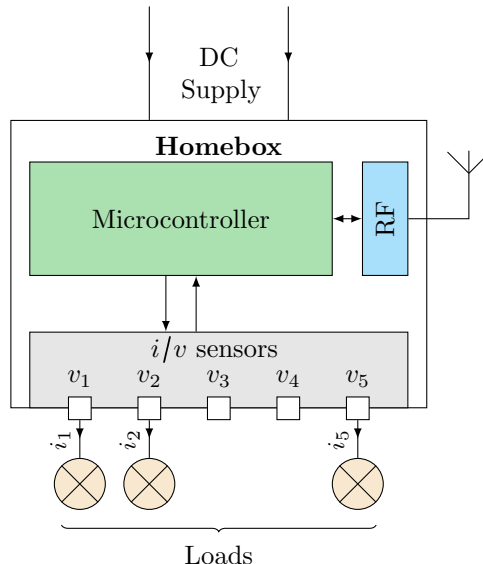
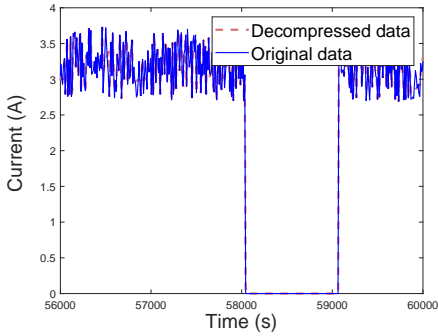
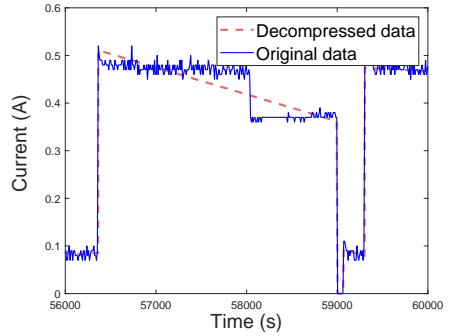


Fig. 3: Block diagram of laboratory setup

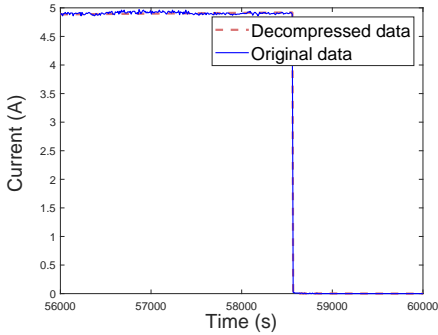
randomly chosen between one and 30 minutes to prevent erratic switching activities on the load profile. The fan has three speed settings and the speed was manually controlled at intervals during the duration of the experiment. The current and voltage data from each of the ports were collected and sent to the SBC. A total of 11,500 (i, v, t) data points were collected at a frequency of approximately 10 s between consecutive samples.



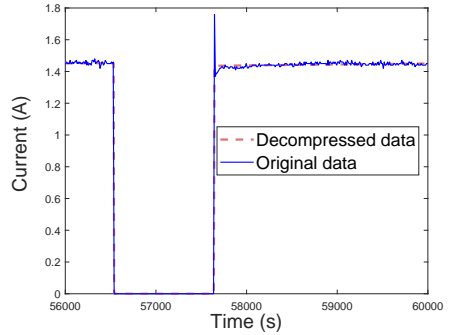
(a) 45 W AC Fan



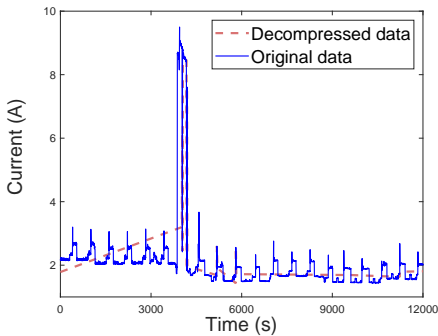
(b) 4.6 W DC Bulb



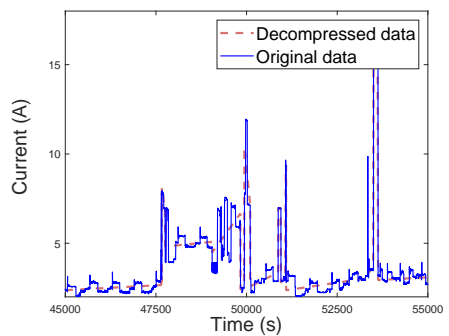
(c) 50 W DC Bulb



(d) 20 W AC Fan



(e) Residential 12 am to 3 am



(f) Residential 6 pm to 9 pm

Fig. 4: Reconstruction of load profile data compressed using the MLRA

4 Results and Discussion

In this section, we will use the proposed algorithm to compress energy load profiles collected from two datasets. The first dataset is taken from the laboratory experiment described in Section 3, which has a resolution of approximately 10 s, while the second dataset is taken from a UK residential household over a period of 24 hours with a resolution of approximately one second totalling 85,903 current, voltage and time tuples. For missing data, linear interpolation was performed to determine the current and voltage values anytime the time difference between two consecutive samples was equal to or more than twice the expected resolution.

The linear regression algorithm (LRA) proposed in [13] will be used for comparison. To simplify the comparison of results and the decompression step, the LRA as implemented in [13] is modified such that there are an equal number of line segments for both \mathbf{i} and \mathbf{v} . Furthermore, the last time sample is appended to the end of the compression results for both algorithms, which is used in the decompression step. The algorithms were executed on a computer with an Intel i5-7360U CPU operating at 2.3 GHz.

4.1 Compression Results

The load profiles from the laboratory experiment and the UK residential household were compressed using $F = 1$, and compared with data compression results using $F = 10,000$ as shown in Tables 1 and 2 respectively. The collected data for the laboratory experiment datasets were compressed using the MLRA with $P_{\min} = 1\text{ W}$ for the experimental datasets, and 300 W for the residential load profile, and V_{load} was selected as 12 V and 240 V respectively. A value of $B_0 = 10$ was used for both cases. For the LRA, the datasets were compressed using $m = 4$ and $n_0 = 10$. We also analyse the use of a second-order polynomial regression, which is denoted by POLY2, based on the LRA, and using the same values of m and n_0 .

From the results, it can be seen that the impact of the linear regression slope scaling is significant. Unsurprisingly, the 4.6 W load experienced the worst energy error performance when no scaling is applied which is due to its comparatively small current values. Interestingly, the LRA benefited more from the application of coefficient scaling, achieving a reduction to the energy error by a factor of more than 100,000 when scaling is applied compared to a reduction by a factor of approximately 2000 for the proposed MLRA. For the laboratory experiment, the LRA achieved an energy error of 0.06% at a compression ratio of 63 on average, while the proposed MLRA achieved an energy error of 0.04% at a lower compression ratio of 40 but with a $\times 68$ reduction in the compression time. For the residential household, the LRA achieved an energy error of 0.012% at a compression ratio of 137, while the proposed MLRA achieved 1/30th of the energy error of the LRA but at half the compression ratio, and with approximately a 200-fold reduction in the compression time. Observe from Tables 1 and 2 that the application of scaling only affects the

Table 1: Data compression results for the LRA and non-iterative MLRA without coefficient scaling

45 W AC Fan				4.6 W DC Bulb				50 W DC Bulb				20 W DC Bulb				Residential AC			
Alg.	DP	e(%)	r	t (ms)	e(%)	r	t (ms)	e(%)	r	t (ms)	e(%)	r	t (ms)	e(%)	r	t (ms)	e(%)	r	t (ms)
LRA	4	13.068	62	589	161.154	36	518	1.706	80	622	18.138	76	663	3.930	137	49220			
MLRA	4	2.188	2	81	140.119	13	14	4.180	76	7	19.830	72	10	2.586	55	246			
POLY2	8	348.619	58	420	308.102	192	1505	247.493	61	470	123.932	54	473	49.547	802	685618			

Table 2: Data compression results for the LRA and non-iterative MLRA with coefficient scaling

45 W AC Fan				4.6 W DC Bulb				50 W DC Bulb				20 W DC Bulb				Residential AC			
Alg.	DP	e(%)	r	t (ms)	e(%)	r	t (ms)	e(%)	r	t (ms)	e(%)	r	t (ms)	e(%)	r	t (ms)	e(%)	r	t (ms)
LRA	4	0.048	62	613	0.000	36	516	0.102	80	637	0.101	76	627	0.012	137	49527			
MLRA	4	0.027	2	88	0.022	13	14	0.005	76	6	0.106	72	5	0.000	55	240			
POLY2	8	0.399	58	503	0.136	192	2274	0.107	61	685	0.056	54	691	0.154	802	672247			

Table 3: Data compression results for iterative MLRA

45 W AC Fan				4.6 W DC Bulb				50 W DC Bulb				20 W DC Bulb				Residential AC			
Iter.	e(%)	r	t (ms)	e(%)	r	t (ms)	e(%)	r	t (ms)	e(%)	r	t (ms)	e(%)	r	t (ms)	e(%)	r	t (ms)	
2	0.060	4	135	0.016	18	30	0.008	85	12	0.106	72	12	0.001	77	479				
4	0.147	19	182	0.014	28	59	0.008	85	29	0.106	72	26	0.003	143	818				
6	0.038	62	194	0.026	55	76	0.008	85	40	0.102	76	58	0.009	278	996				
8	0.027	65	220	0.016	68	94	0.008	85	49	0.102	76	67	0.011	404	1395				
10	0.027	65	242	0.016	68	99	0.008	85	61	0.102	76	74	0.012	493	1425				

energy error and not the compression ratio, since the use of scaling does not affect the number of compression coefficients created as captured in Eq. (9).

It should be noted that the results of MLRA in Tables 1 and 2 depend on the initial values of the run parameters used. Whereas the MLRA achieved a comparable compression ratio with the LRA using $B = 10$ for the 20 W case, it achieved a much lower compression ratio in comparison using the same value of B in the 45 W case, although at a lower energy error. The contrasting performances of the different loads for the same value of B suggest that a variable rather than a global value for B should be employed.

Interestingly, the use of a higher-order polynomial regression did not appear to have a positive impact on the accuracy of the compression. This could be because the load profiles considered are mostly flat, which is well-represented using simple linear regression. Separately, POLY2 was more significantly impacted by rounding as a result of its comparatively smaller compression coefficients as shown in Table 1. To achieve meaningful compression results, the coefficients of POLY2 required eight decimal places compared to only four decimal places for the LRA and MLRA. It should be noted that the compressed data size increases in proportion to the number of decimal places that are retained. POLY2 also incurred a longer compression time due to the higher complexity of the second-order polynomial regression. As such, subsequent discussions will be limited to the linear-regression-based LRA and MLRA.

4.2 Iterative Implementation of MLRA

In this section, the iterative compression results of the MLRA will be presented and compared with the LRA. A scaling factor, $F = 10,000$ will be used for both algorithms. The collected data was compressed using the proposed MLRA with a minimum target energy error of 1%, $B_0 = 10$, $\Delta B = 5$ and five values of J as shown in Table 3.

An interesting observation is the fact that the algorithm achieved much higher compression ratios in the residential load profile compared to the laboratory load profile. Apart from using a higher resolution (discussed in more detail in Section 4.3), the residential load profile was taken over a longer period, which featured stretches of data points that could be fitted onto a few line segments. For example, the residential load profile from 12 am to 3 am experienced almost thrice the compression ratio as the period from 6 pm to 9 pm (see Figs. 4e and 4f) where energy usage activity was comparatively higher.

Using six iterations, the proposed MLRA achieved an average energy error of 0.04% at a compression ratio of 69, while the LRA achieved a higher average energy error of 0.06% and a lower compression ratio of 63, with the MLRA having almost a $\times 10$ speed advantage in the DC datasets. On the other hand, using four iterations, the proposed MLRA achieved more than a four-times reduction to the energy error, and with a higher compression ratio compared to the LRA while enjoying a $\times 60$ speed advantage in the residential load profile case.

Table 4: Evaluation of performance of different data resolutions on the performance of the data compression algorithms.

$\Delta t(\text{s})$	LRA		MLRA	
	e (%)	r	e (%)	r
1	0.012	137	0.009	278
5	0.050	110	0.006	64
10	0.049	89	0.004	36
15	0.034	101	0.004	28
30	0.033	78	0.003	15
60	0.035	57	0.004	9

In the compression results for the 4.6 W load, it can be observed that both the energy error and compression ratio can be jointly improved by utilising more iterations, achieving a lower energy error and higher compression ratio using ten iterations compared to using only one iteration. However, the use of the iterative mechanism did not always result in any significant impact as observed in the 20 and 50 W load cases. This could be attributed to the fact that the DC microgrid dataset generally featured a simple load profile, where a constant DC load was switched on and off. This resulted in a comparatively smoother load profile, such that changing the value of B did not always result in increasing the number of compression groups, which often results in an unchanging compression ratio. On the other hand, the residential load profile which featured an aggregated load profile appeared to be more impacted by the use of the iterative mechanism. Overall, for a given compression ratio, the proposed algorithm demonstrated a better energy error performance, on average, as well as a shorter computation time compared with the LRA.

4.3 Impact of Resolution

To analyse the impact of the data resolution on the energy error and compression ratio of the algorithms, the residential dataset was compressed at 1, 5, 10, 15, 30 and 60 s resolutions as shown in Table 4. Six iterations were utilised for the MLRA. The results show that as the time step is increased, the compression ratio is negatively impacted. For the MLRA, the compression ratio is reduced by a factor of 30 at a resolution of 60 s, while for the LRA, the reduction is just under a factor of 3. This disparity is due to the fact that irrespective of the resolution, the LRA creates an initial group of n_0 points when creating its line segments (without performing any boundary check) making it more robust to reductions in the data resolution from the perspective of the compression ratio. In the case of the MLRA, we surmise that the reason for the impact on the compression ratio, in this case study, is that using larger time steps tends to reduce the correlations between neighbouring points, meaning more line segments will be created, i.e., as the time resolution reduces, it is

more likely that the user has switched to a different activity, hence generating a different load profile pattern.

On the other hand, the energy error did not appear to show the same correlation with the resolution as it did in the case of the compression ratio. Although the compression ratio of the MLRA was affected more by a reduction in the time resolution, it maintains a better energy error than the LRA for all the resolutions considered. The results suggest that with respect to achieving a high compression ratio, the proposed algorithm is more suited to higher resolution data where a high degree of correlation exists between successive data points. However, the impact on the energy error was less noticeable, which suggests that the energy error is more sensitive to the nature of the data itself than the resolution employed.

4.4 Comparison with Lossless Compression Algorithms

In this section, we compare the proposed algorithm with the bzip2 and DEFLATE lossless algorithms in terms of the compression ratio and compression time using the residential load profile dataset. This is done to analyse the tradeoff of performance versus compression ratio using the proposed algorithm in comparison with lossless algorithms. The algorithms were applied to the residential load profile data using a 1 s resolution, and achieve compression ratios of 7 and 9 at a computation time of 130 ms and 290 ms respectively.

Using a single iteration (see Table 2), the proposed algorithm practically achieves the same energy error performance as the bzip2 and DEFLATE lossless algorithm at more than six times the compression ratio of either algorithm. However, there does not appear to be any significant speed advantage using the proposed algorithm. With more iterations, the proposed algorithm is able to significantly improve the compression ratio, while maintaining an energy error of below 0.2%. This result suggests that the proposed algorithm is a practical alternative to lossless data compression in the case study considered. For other datasets, we recommend extensive tests to determine the suitability of the proposed algorithm to the particular application.

5 Conclusion

In this paper, we have presented a data compression algorithm for an islanded DC microgrid system which substitutes load profile data with linear regression coefficients. Data compression algorithms typically require certain input parameters to be provided by the user. To enable the algorithm to operate autonomously, an iterative mechanism was devised, which allows the algorithm to explore the compressed result with the best combination of accuracy and compression ratio within a given number of iterations.

The algorithm was applied to load profiles from a DC microgrid laboratory experiment and a UK AC-grid-connected household and it achieved an energy error of less than 0.2% and a compression ratio of more than 50-to-1 using six iterations for each of the load profiles considered. Furthermore, the algorithm

was compared to that of a similar linear-regression-based algorithm [13], and it achieved a lower energy error at a higher compression ratio, on average, while also reducing the data compression time by a factor of more than 30. We also presented a scaling technique which reduces the energy error by a factor of more than 1000 when rounding is applied to the linear regression coefficients.

One limitation of the proposed algorithm is that the compression ratio is affected when the resolution is reduced. Therefore, further work is required in investigating algorithms that work comparably with respect to both the energy error and compression ratio for high and low-resolution load profiles. In this work, we also considered the scenario where the operator of the data compression algorithm has foreknowledge of certain parameters of the load profile data, e.g., the minimum load power and the resolution. For future research, it will be desirable if such parameters can be intelligently and efficiently estimated from the input data with minimal input from the operator, which will improve the portability of the algorithm across different load profiles.

References

- [1] Neal, D., Ding, Y., Bello, I., Han, R., Byamukama, M., Kebir, N., McCulloch, M., Rogers, D., Nyamongo, I., Waweru, K.: Demand Side Energy Management and Customer Behavioral Response in a Rural Islanded Microgrid. (2020)
- [2] Wen, L., Zhou, K., Yang, S., Li, L.: Compression of smart meter big data: A survey. *Renewable and Sustainable Energy Reviews* **91**, 59–69 (2018). Accessed 2020-08-08
- [3] Alahakoon, D., Yu, X.: Smart Electricity Meter Data Intelligence for Future Energy Systems: A Survey. *IEEE Transactions on Industrial Informatics* **12**(1), 425–436 (2016). Conference Name: IEEE Transactions on Industrial Informatics
- [4] Qdr, Q.: Benefits of demand response in electricity markets and recommendations for achieving them. US Dept. Energy, Washington, DC, USA, Tech. Rep (2006). Publisher: Citeseer
- [5] Zhou, K., Fu, C., Yang, S.: Big data driven smart energy management: From big data to big insights. *Renewable and Sustainable Energy Reviews* **56**, 215–225 (2016). Accessed 2020-08-09
- [6] Wang, Y., Chen, Q., Hong, T., Kang, C.: Review of Smart Meter Data Analytics: Applications, Methodologies, and Challenges. *IEEE Transactions on Smart Grid* **10**(3), 3125–3148 (2019).
- [7] Wilcox, T., Jin, N., Flach, P., Thumim, J.: A Big Data platform for smart meter data analytics. *Computers in Industry* **105**, 250–259 (2019). Accessed 2021-02-07

- [8] Chicco, G., Mazza, A.: Impact of the Time Resolution for Data Gathering on Loss Calculation and Demand Side Flexibility. In: 2020 International Conference on Smart Energy Systems and Technologies (SEST), pp. 1–6 (2020).
- [9] Crdova, S., Caizares, C., Lorca, ., Caizares, D.E.: An Energy Management System With Short-Term Fluctuation Reserves and Battery Degradation for Isolated Microgrids. *IEEE Transactions on Smart Grid* **12**(6), 4668–4680 (2021). Conference Name: IEEE Transactions on Smart Grid
- [10] Plenz, M., Meyer, M.F., Grumm, F., Becker, D., Schulz, D., McCulloch, M.: Impact of Lossy Compression Techniques on the Impedance Determination. *Energies* **13**(14), 3661 (2020). Number: 14 Publisher: Multidisciplinary Digital Publishing Institute. Accessed 2020-07-20
- [11] Seward, J.: The bzip2 and libbzip2 official homepage. <http://sourceware.cygnus.com/bzip2/> (2000)
- [12] Deutsch, P.: RFC1951: DEFLATE compressed data format specification version 1.3. RFC Editor (1996). [dl.acm.org/doi/pdf/10.17487/RFC1951](https://doi.org/10.17487/RFC1951) Accessed 2021-01-30
- [13] Clements, A.F., McCulloch, M.D., Nixon, K.J.: Low-loss, high-compression of energy profiles. In: 2015 International Conference on Renewable Energy Research and Applications (ICRERA), pp. 949–953 (2015).
- [14] Capo-Chichi, E.P., Guyennet, H., Friedt, J.-M.: K-RLE: A New Data Compression Algorithm for Wireless Sensor Network. In: 2009 Third International Conference on Sensor Technologies and Applications, pp. 502–507 (2009).
- [15] Eichinger, F., Efros, P., Karnouskos, S., Bhm, K.: A time-series compression technique and its application to the smart grid. *The VLDB Journal* **24**(2), 193–218 (2015). Accessed 2020-06-21
- [16] Mendes, D.L.S., Rabelo, R.A.L., Veloso, A.F.S., Rodrigues, J.J.P.C., dos Reis Junior, J.V.: An adaptive data compression mechanism for smart meters considering a demand side management scenario. *Journal of Cleaner Production* **255**, 120190 (2020). Accessed 2020-07-19
- [17] Engel, D.: Wavelet-based load profile representation for smart meter privacy. In: 2013 IEEE PES Innovative Smart Grid Technologies Conference (ISGT), pp. 1–6 (2013).
- [18] Fang, X., Misra, S., Xue, G., Yang, D.: Smart Grid The New and

- Improved Power Grid: A Survey. *IEEE Communications Surveys Tutorials* **14**(4), 944–980 (2012). Conference Name: IEEE Communications Surveys Tutorials
- [19] Al-Kadhim, H.M., Al-Raweshidy, H.S.: Energy Efficient Data Compression in Cloud Based IoT. *IEEE Sensors Journal* **21**(10), 12212–12219 (2021). Conference Name: IEEE Sensors Journal
 - [20] Huang, J.-F., Zhang, G.-H., Hsieh, S.-Y.: Real-time energy data compression strategy for reducing data traffic based on smart grid AMI networks. *The Journal of Supercomputing* **77**(9), 10097–10116 (2021). Accessed 2022-03-21
 - [21] Weisberg, S.: *Applied Linear Regression*. John Wiley & Sons, (2005)
 - [22] Hunkeler, U., Truong, H.L., Stanford-Clark, A.: MQTT-S A publish/subscribe protocol for Wireless Sensor Networks. In: 2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE '08), pp. 791–798 (2008).
 - [23] Unterweger, A., Engel, D.: Resumable Load Data Compression in Smart Grids. *IEEE Transactions on Smart Grid* **6**(2), 919–929 (2015). Conference Name: IEEE Transactions on Smart Grid
 - [24] Hyndman, R.J., Koehler, A.B.: Another look at measures of forecast accuracy. *International Journal of Forecasting* **22**(4), 679–688 (2006). Accessed 2021-10-02
 - [25] Haben, S., Ward, J., Vukadinovic Greetham, D., Singleton, C., Grindrod, P.: A new error measure for forecasts of household-level, high resolution electrical energy consumption. *International Journal of Forecasting* **30**(2), 246–256 (2014). Accessed 2021-10-01
 - [26] Marcelloni, F., Vecchio, M.: A Simple Algorithm for Data Compression in Wireless Sensor Networks. *IEEE Communications Letters* **12**(6), 411–413 (2008). Conference Name: IEEE Communications Letters
 - [27] Coley, G.: *Beaglebone black system reference manual*. Texas Instruments, Dallas **5** (2013)

Acknowledgments. The authors wish to acknowledge financial support from the Engineering and Physical Sciences Research Council (EPSRC), grant no. EP/R030111/1, Robust Extra Low Cost Nano-grids (RELCON).