OBDZero User Manual 3.8

OBDZero reads, displays and stores data from the iMiev, CZero and iOn electric cars. The data such as speed and electricity use are available on the car's CAN computer network via a Bluetooth dongle attached to the car's OBD port. OBDZero presents this data in 8 different screens. A 9th screens logs messages between the app, the OBD dongle and the car. Two screens are intended for use while driving. One of these, the WATTS screens, shows the car's average watts, speed and watt-hours per km. The other screen, DRIVE, updates the distance to the next charging station, the difference between the remaining (aka rest) range and the distance to the station, and suggests a speed to the station.

OBDZero saves data in semicolon separated text files, either in the phone's storage or on a SD Card depending on how the phone is set up. The app was developed on an older phone running Android 4.3 with an INTEY OBDII, an inexpensive Bluetooth dongle. It also runs on the more expensive OBDLink LX dongle, though a firmware update of the OBDLink may be necessary. It has been tested on a newer phone running Android 7.0. The app does not exchange data with the Internet and it does not use GPS. This user manual applies to OBDZero version 3.8 released on August 10, 2021.

What's new in version 3.8

On the OBD and the PIDs screens if the screens are scrolled they remain where they are when updated with new data. This makes it possible to follow changes in parameters and PIDs further down the list.

On the Drive screen when the distance to the next charging station is 0, the speed shown is the speed that will result in the true rest range agreeing with the rest range indicated by the car.

There are now two battery 100% capacity measurements, Cap1 and Cap2. In both cases the capacities of all the cells are measured. In previous versions only the capacity of the weakest cell was measured. Also in previous versions it was only necessary at reduce the SoC to less than 20% in order to do the measurement. Now the SoC must be reduced to less than 15%. This significantly improves the accuracy of the measurement.

Cap1 is a simplified measurement compared with Cap2. It doesn't included charging the battery. Therefore it is less accurate than Cap2 in that it usually gives lower values for the capacity. However it can still be used to compare the capacities of the cells. And if it does show a greater capacity than the car's estimate then the car's estimate is probably lower than the true capacity.

In some versions of Android the data files saved by OBDZero were not immediately visible in Windows Explorer even though the phone was connected to the PC and set to file transfer. This problem has been fixed in version 3.8. However in Android 10 and 11 a new file storage structure was introduced. Version 3.8 requests the old structure on Android 10 but the old structure is not available on Android 11. This means that data storage does not work on Android 11 phones. On the other hand, the other app functions should work. I would really appreciate feedback from Android 11 users in this regard.

Bluetooth dongles

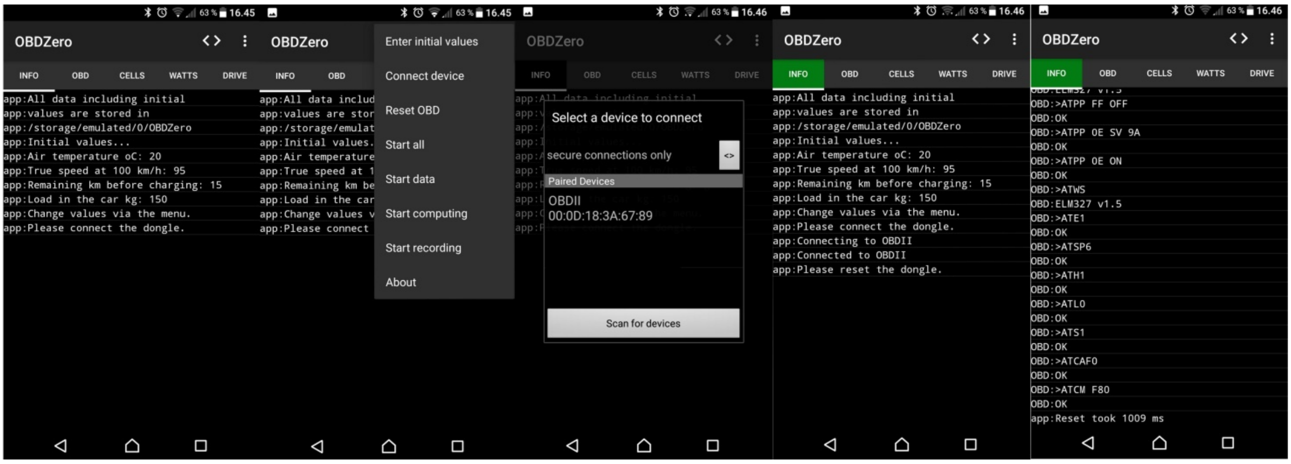
The dongles I have tested show differences in speed and stability. The app collects data in cycles. For example the OBDLink LX cycle time can be less than 1 sec. For the Vgate Scan the cycle takes between 1.5 and 2.5 sec. For the INTEY the cycle takes almost 4 sec.

As a rule more expensive dongles work better than less expensive models. For example the inexpensive INTEY runs for some hours before losing contact with the phone. On occasion the INTEY takes more than a minute to connect to the phone and often must be restated before it works properly. Also some dongles show a curious problem in which the cycle time increases during the first 10 min. of operation after which it suddenly returns to normal and stays that way for the rest of the session. I also tested a very inexpensive dongle, a blue ELM 327 mini. It did not work at all.

The Vgate Company sent a number of their dongles for testing and the results are positive. In particular I can recommend the iCar Pro BLE4.0 Dual and the vLink MC. The iCar Pro has a cycle time of 3 to 4 secs while the vLink MC has a cycle time of 1 to 2 sec which is similar to the more expensive OBDLink LX. Both these dongles are quick to pair and link with the telephone and the Bluetooth connection is stable. There are pirate copies of Vgate dongles for sale on the internet. In fact I have used a pirate copy of the Vgate Scan dongle. Tests of the copy and a true Vgate Scan showed that the true Scan is more stable and faster than the copy. When buying a dongle that purports to be manufactured by Vgate, check that Vgate is the supplier.

Startup Instructions

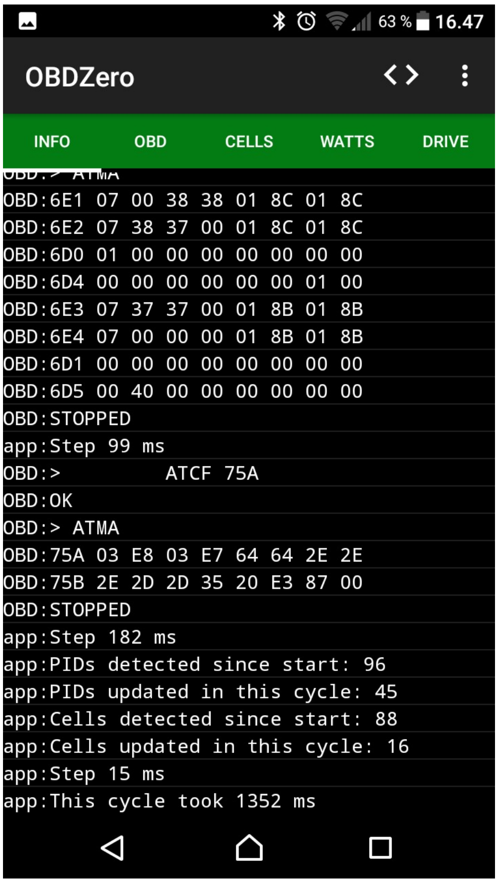
- Plug in the dongle and turn on the car.
- Pair the phone and the dongle using the phone's Bluetooth function
- Start OBDZero and stay with the INFO screen
- Open the menu (usually the menu button is three dots in the upper right hand corner)
- Connect the device (dongle). This may require more than one try.
- Once connected open the menu again and reset the dongle.
- Wait for the reset to complete (about 2 sec)
- Open the menu and press one of the Start buttons



The 5 buttons just below the title are used to move between screens. They change color from black to green not as they are pressed but as the apps functions come online. The buttons and functions are:
INFO: the dongle and the telephone are connected via Bluetooth
OBD: the telephone is receiving data from the car and presenting raw information such as speed.
CELLS: the telephone has received data from all the cells of the battery
WATTS: the telephone is computing the numbers shown on the watts and drive screens
DRIVE: the telephone is saving data in semicolon separated text files.

When all the functions are online the INFO screen will look like the screen shown in the next section.

The INFO screen



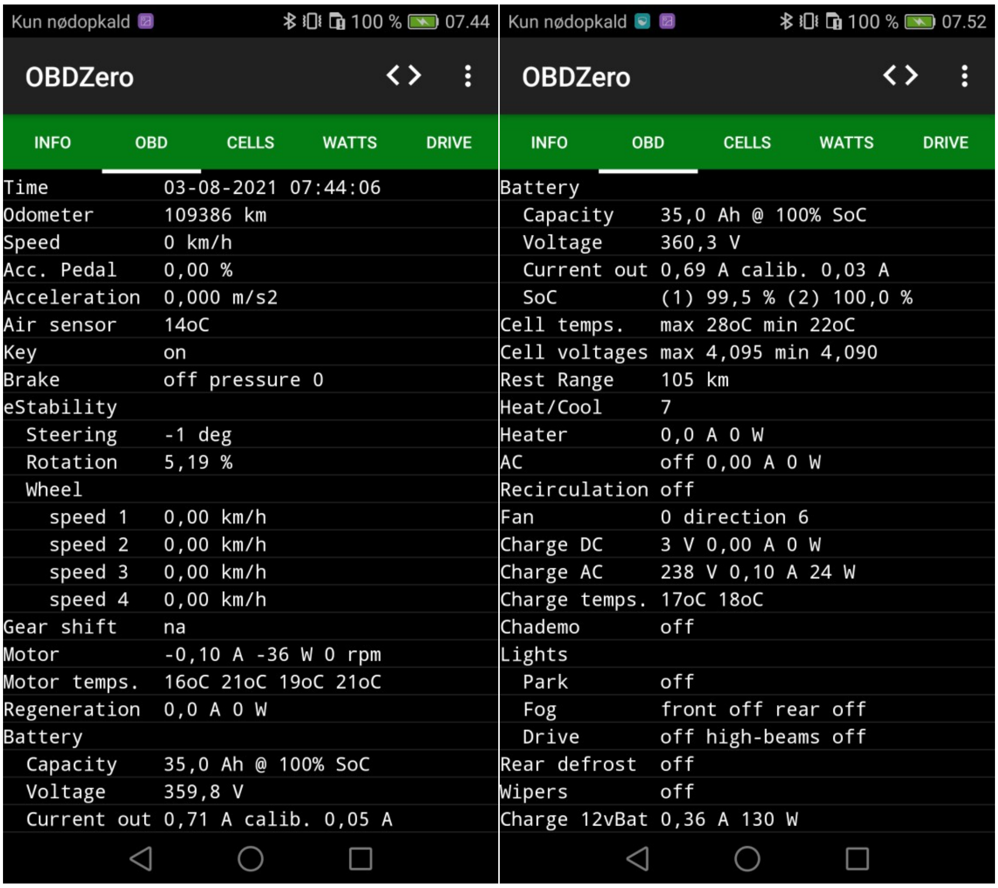
The INFO screen shows messages between the app and the OBD dongle.

Lines that begin with “OBD:” are either commands sent to the dongle or data from the car passed through the dongle. OBD: lines containing “AT” followed by other letters and numbers are commands originally sent from the app and then echoed by the dongle. These commands are found in ELM327DSH.pdf from www.elmelectronics.com. OBD: lines followed by a three character code and up to 8 two character codes contain the raw data from the car.

Lines beginning with “app:” contain information from the app such as the number of milliseconds required for each step of data collection. The number of unique lines of data collected in each collection cycle, 45 in the example shown, the number of unique cell readings collected in a cycle, in this example 16 and the time required for the cycle, in this example 1352 ms. The dongle in use in this case was the Vgate Scan. Note that not all readings are updated in each cycle. It takes about 5 cycles to update all the readings. However, fast changing values such as speed and amps from the battery are updated in most cycles. Other app: messages are instructions or information to the user. Therefore the INFO screen is the screen to use when starting the app and when diagnosing problems.

When the app is shut down, the first lines during start up and the last lines during shut down shown on the INFO screen are stored in the Info_ file in the OBDZero folder on the phone. This log can be used for troubleshooting communication problems.

The OBD screen



The OBD screen shows the raw data from the car converted to numbers that are easily understood. The rules for converting this data are not readily available from the car manufacturer. Instead interested owners have spent hours observing the raw data, deducing the rules and publishing them on the Internet. In particular I must thank jjlink, garygid, priusfan, plaes, dax, cristi, and kiev all of whom have contributed conversions on http://myimiev.com/forum/. However, there are still many lines of data which we don't know how to interpret yet.

Most of the numbers shown on the OBD screen are self-explanatory. However a few can be confusing. E.g. there are two numbers for battery out amps. The first is the amps computed using the conversion found on the iMiev forum. The second is amps I compute after having calibrated the amps measurement on our car. I believe that this is the more accurate value for all iMiev/CZero/iOns.

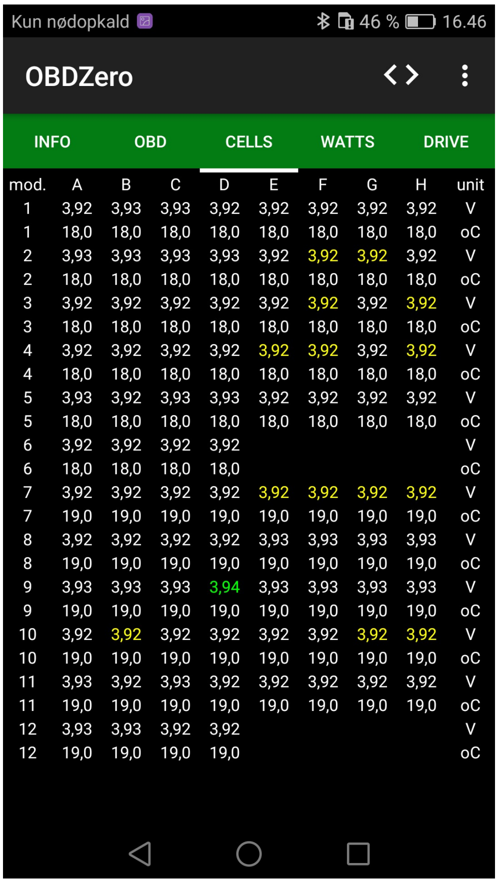
There are two values for the State of Charge (SoC). Both are computed by the car as part of a system for estimating the present 100% capacity of the battery aka State of Health (SoH). As such neither is more correct than the other.

Charge DC is the amps and volts from the car's charging unit to the battery. Charge AC is the amps and volts from the mains to the charging unit via the port on the right hand side of the car.

The data shown on the OBD screen is stored in the OBD_ file on the phone.

Annex 2 is a list of OBD data including the rules for converting the raw data on the car's network to the data shown on the OBD screen.

The CELLS screen



mod.	A	B	C	D	E	F	G	H	unit
1	3,92	3,93	3,93	3,92	3,92	3,92	3,92	3,92	V
1	18,0	18,0	18,0	18,0	18,0	18,0	18,0	18,0	oC
2	3,93	3,93	3,93	3,93	3,92	3,92	3,92	3,92	V
2	18,0	18,0	18,0	18,0	18,0	18,0	18,0	18,0	oC
3	3,92	3,92	3,92	3,92	3,92	3,92	3,92	3,92	V
3	18,0	18,0	18,0	18,0	18,0	18,0	18,0	18,0	oC
4	3,92	3,92	3,92	3,92	3,92	3,92	3,92	3,92	V
4	18,0	18,0	18,0	18,0	18,0	18,0	18,0	18,0	oC
5	3,93	3,92	3,93	3,93	3,92	3,92	3,92	3,92	V
5	18,0	18,0	18,0	18,0	18,0	18,0	18,0	18,0	oC
6	3,92	3,92	3,92	3,92					V
6	18,0	18,0	18,0	18,0					oC
7	3,92	3,92	3,92	3,92	3,92	3,92	3,92	3,92	V
7	19,0	19,0	19,0	19,0	19,0	19,0	19,0	19,0	oC
8	3,92	3,92	3,92	3,92	3,93	3,93	3,93	3,93	V
8	19,0	19,0	19,0	19,0	19,0	19,0	19,0	19,0	oC
9	3,93	3,93	3,93	3,94	3,93	3,93	3,93	3,93	V
9	19,0	19,0	19,0	19,0	19,0	19,0	19,0	19,0	oC
10	3,92	3,92	3,92	3,92	3,92	3,92	3,92	3,92	V
10	19,0	19,0	19,0	19,0	19,0	19,0	19,0	19,0	oC
11	3,93	3,92	3,93	3,92	3,92	3,92	3,92	3,92	V
11	19,0	19,0	19,0	19,0	19,0	19,0	19,0	19,0	oC
12	3,93	3,93	3,92	3,92					V
12	19,0	19,0	19,0	19,0					oC

For most iMiev/CZero/iOns the battery pack contains 88 cells. There cells are packed together in units of 4 cells. The units are then mounted in modules. 10 modules have 2 units each and two modules, 6 and 12, have one unit each. Cells are referred to by letters from A to H here and in the car's technical manual. The voltage for each cell is measured with a resolution of 0.005 volts however because of the limited width of the screen the voltage is shown with 2 decimals instead of 3.

In each 4 cell unit there are three temperature sensors mounted between the cells. Therefore there are only 3 temperature measurements for 4 cells. The temperatures for the cells shown on the screen are:
Cell A = Sensor 1,
Cell B = (Sensor 1 + Sensor 2)/2,
Cell C = (Sensor 2 + Sensor 3)/2 and
Cell D = Sensor 3
Temperatures for the cells in the second unit in each module are computed in the same way.

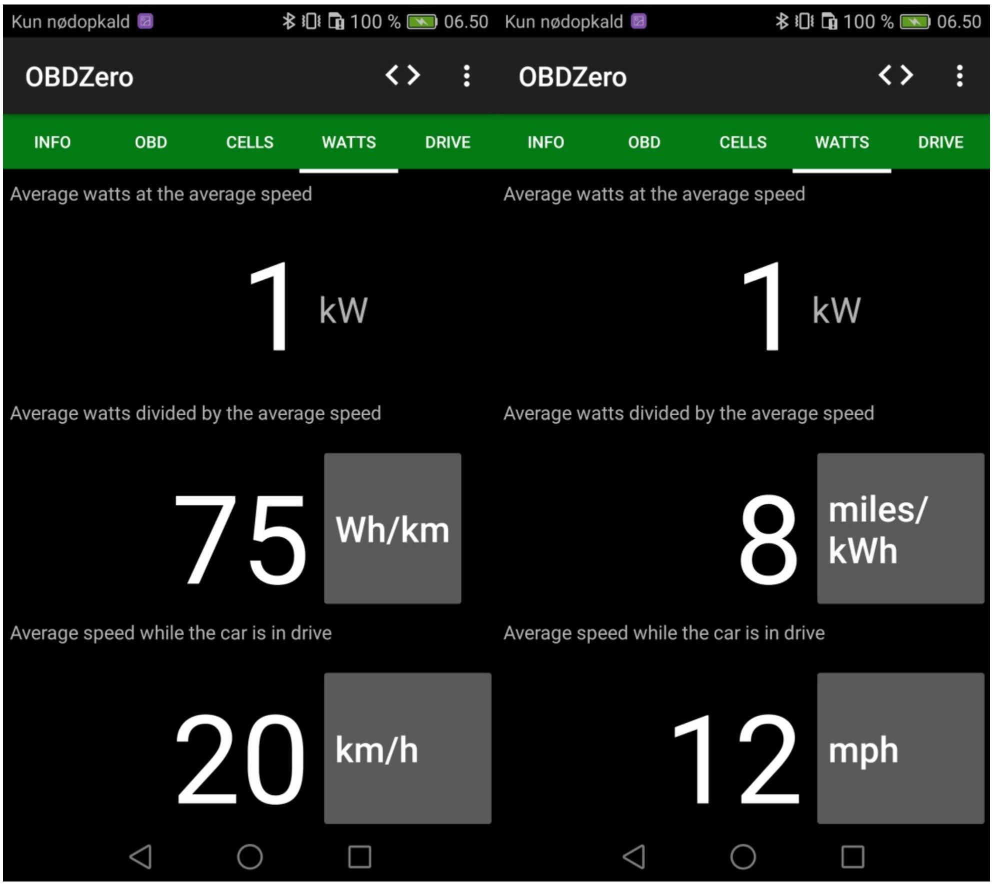
In order to make diagnosing of cell problems easier the cells with the highest and lowest values are marked in color such that the highest voltages and lowest temperatures are green and the lowest voltages and highest temperatures are marked with yellow. Red is not used because low voltages and high temperatures are not necessarily indicative of failure. Also to avoid marking insignificant differences the difference between the highest and lowest voltages must be greater than 0.02 volts and the difference between the highest and lowest temperatures must be greater than 1oC before the cells are marked in color.

Some cars built after 2012 have only 80 cells. These cars do not have modules 6 and 12.

Unfortunately, starting with cars built in 2015, the manufacture stopped the transmission of individual cell data on the car's computer network (CAN). Therefore this data cannot be shown on the Cells screen

Cell voltages and the computed temperatures are stored in the Cells_ file on the phone. Temperature readings from the sensors are stored in the CellTemperatures_ file.

The Watts screen



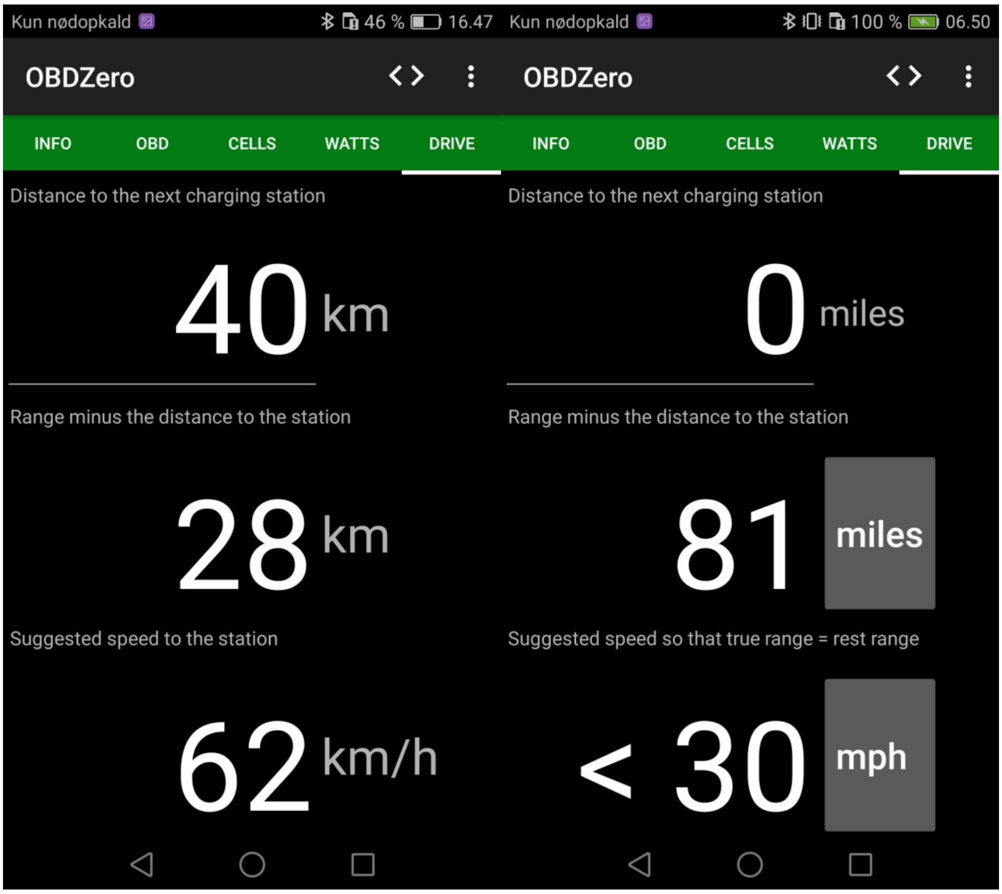
This screen shows average values for the watts from the battery for all purposes, the car's speed and the resulting value for watt-hours per km. By pressing the units buttons Wh/km can be changed to miles/kWh and km/h can be changed to mph. The car's average speed is computed using an exponential filter. This filter is used because it is easy to implement in code. More information on exponential filters can be found on the Internet.

kW are the average measured kW from the battery. As with the speed this average is computed using an exponential filter. To account for changes such as turning the heater on and off the watts for auxiliary functions are subtracted from the measured watts before averaging then added back after averaging. This is the kW shown on the screen. Therefore the kW shown will increase as soon as e.g. the heater is turned on and decrease again as soon as the heater is turned off.

Wh/km is the average watts as shown divided by the average speed. Miles/kwh is the average mph divided by the average kW. Therefore these numbers will also change as e.g. the heater is turned on and off.

Note that the average speed is updated when the car is in drive. This means that when the car is stopped at lights or by traffic 0 is averaged into the speed and the speed shown will decrease. To avoid unrealistically large values for the Wh/km the average speed doesn't decrease below 1 km/h (0.6 mph).

The DRIVE screen



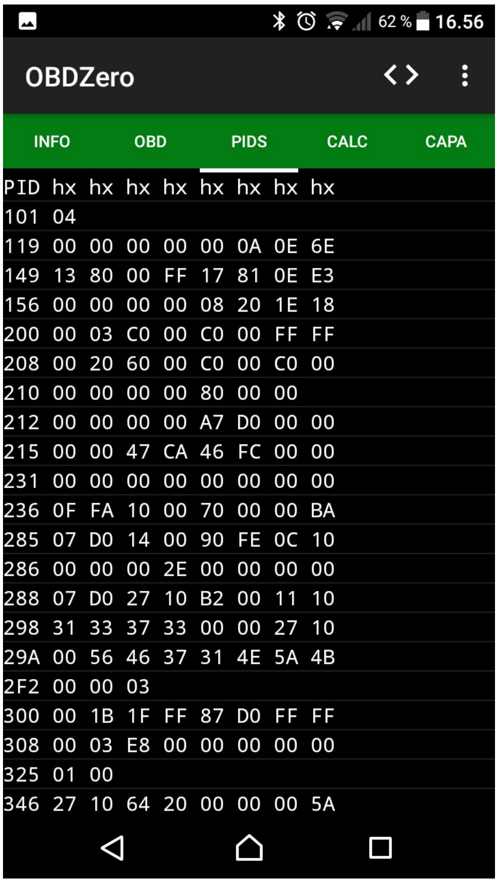
This screen is a trip planning and execution tool. Prior to starting a trip the user enters the expected distance to the next charging station found using e.g. Google Maps. In some cases this may be an out and back distance. The app then computes the difference between the range reported by the car and the distance needed. Hopefully this will be a positive number. While driving, the app computes the distance travelled by multiplying the speed by the time between readings. It then updates the distance to the charging station accordingly. The range minus the distance is also updated. There is a correction factor which adjusts for errors in the speed indicated by the car. As a rule the indicated speed is about 5% greater than the true speed. The correction factor for the speed can be changed in the Initial Values menu, more on this later.

There is no GPS connection so if one deviates from the planned route the distance to the charging station will be in error.

The suggested speed uses a model of the car that computes watts used at any speed. It finds a speed that according to the model will get the car to the charging station with e.g. 15 km in reserve. This may be a speed less than the present speed of the car if the battery is running low or it may be a speed greater than the present speed of the car if there are watt-hours to spare. The preferred number of km in reserve when arriving at the charging station can also be changed in the Initial Values menu. This model is, as with any method of predicting the future, subject to error.

As an extra function, when the distance to the next charging station is 0, the speed shown is the speed that will result in the true rest range agreeing with rest range computed by the car. The preferred remaining distance at the charging station is ignored for this computation.

The PIDS screen



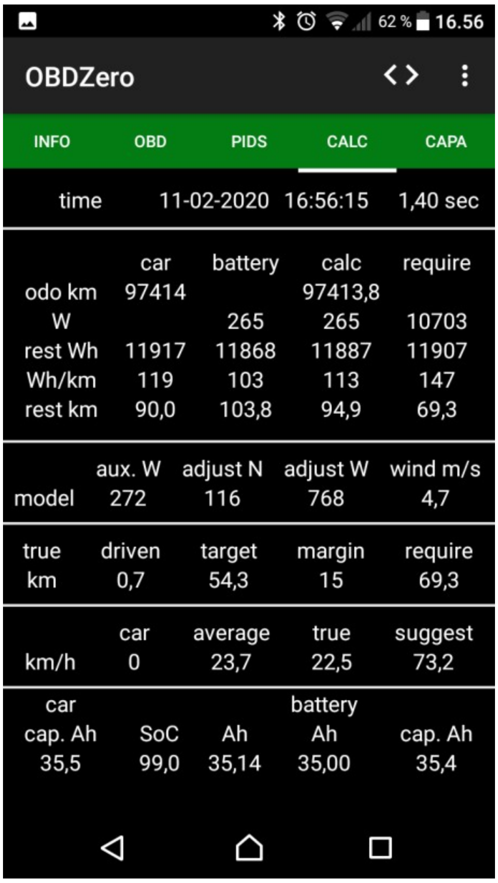
By pressing the <> button in the upper right hand corner of the display, a second set of screens becomes available. The INFO screen is the same as before. The PIDS screen shows all the raw data from the car. Each line starts with a hexadecimal Parameter ID (PID) followed by 1 to 8 bytes of data also in hexadecimal. As mentioned above the meaning of much of this data is as yet unknown. However most of the important data has been decoded. As an example the PID 298 above contains information about the car's motor. PID 298's 7th and 8th bytes encode the motor rpms. 27 hex is 39 decimal and 10 hex is 16 decimal. The rule with these two numbers is:

$$0 = 39*256 + 16 - 10000$$

Therefore, the rpms were 0 when this data was recorded.

The PIDs are stored in two files on the phone. The PID_ file contains lines in which both the PID and the data are in hex as shown above. In the PIDint_ file the PID is in hex but the data is converted to decimal values.

The CALC screen



This screen has primarily to do with the model that was mentioned in the description of the Drive screen. Details of the model will have to wait but in short it computes the watts from the battery at any one moment based on:

- Any change in the speed
- Rolling resistance
- Air resistance and
- Auxiliary power

Auxiliary power is used for heating, AC, headlights etc.

Computing these values is fairly straight forward. The challenge is everything OBDZero doesn't know such as the wind direction and speed and whether the car is moving up or down hills. To compensate for this lack of knowledge the model has a 6th input called an error term in modeling theory. Each time the app completes a data collection cycle the model computes the watts it expects the car to use based on the data collected. This is compared with the measured watts and the error term is adjusted. If the measured value is greater than the model value the error term is increased by a small amount and if the reverse is true the error term is reduced by the same small amount. This adjusted error term brings the model watts closer to the measured watts with each cycle.

Using this method the model values are able to follow the measured values quite well. For example in the screen above, there are four values for the remaining watt-hours in the battery (rest Wh). The first value is the rest Wh based on the SoC and the car's estimate of the rest range. The second rest Wh is based on a sum of the measured watts-hours drawn from the battery. The third rest Wh is based on a sum of the model computed watt-hours drawn from the battery. The numbers aren't exactly the same, it would be surprising if they were, but they are similar and they remain similar over long trips. This indicates that the model is working well. It also indicates that the error term does contain useful information about the factors such as wind speed that we are unable to measure directly. For those of you that are interested, this method of obtaining information is called an “observer” in modeling theory. More information on observers can be found on the Internet.

Computing the suggested speed shown on the DRIVE screen is the primary reason for the model. Using the model with the error term we can estimate the watts needed to power the car at any speed. We also know the distance to the next charging station so we can estimate the time to that station at any speed. With an estimate of the watts and the time we can compute the watt-hours needed at any speed. Then all we have to do is find the speed for which the watt-hours equal the car’s estimate of the watt-hours remaining in the battery and we know how fast we can drive to the charging station. It’s a bit more complicated than that but the rest is details.

In the CALC screen above, the fourth rest Wh is the model computed watt-hours needed to drive 69.3 km at 73.2 km/h. 69.3 km is the remaining distance to the charging station, 54.3 km, plus 15 km in reserve.

Car rest Wh is based on the SoC (state of charge in %) reported by the car. It is calculated using this equation:

Car rest Wh = (SoC/100) * cap.Ah * (Volts + Volt0)/2

Where Volts is the present battery voltage and Volts0 is the battery voltage at SoC = 0. Cap.Ah is the present battery 100% capacity as reported by the car. It is the capacity shown on the OBD screen after the battery voltage. Remember that the last 10% of the battery’s Ah is only available in “turtle” mode. It doesn’t contribute to the rest range shown by the car.

Battery rest Wh is the watt hours based on the sum of the amps used since the OBDZero was started. Battery rest Wh equals Car rest Wh when the app is started. The equation is:

Battery rest Wh = (rest Ah0 – Sum (A * step)) * (Volts +Volts0)/2

Where rest Ah0 is rest Ah based on SoC when the app was started, step is the time step in hours between measurements. Sum (A * step) is the sum of the amp hours used since the app was started.

Model rest Wh is the watt hours remaining based on the model watt hours used. This is a bit more complicated but similar to the Battery rest Wh.

Car Wh/km is the Car rest Wh - 10% of the battery’s capacity in Wh divided by the rest range shown by the car. In reality the car measures the Wh/km continuously and from this computes the estimated rest range. I haven’t found the car’s estimate of Wh/km in the raw data but this is a way of computing it.

Battery Wh/km is the average Watts used divided by the car’s average speed while in drive. You can see the average speed further down the screen. Average Watts are not shown. The Battery Watts shown are the immediate measured Watts.

The Car Wh/km and the Battery Wh/km are usually close to each other. However if you turn on e.g. the heater the Car Wh/km increases suddenly. The Battery Wh/km should increase slowly until it is more or less the same as the Car Wh/km.

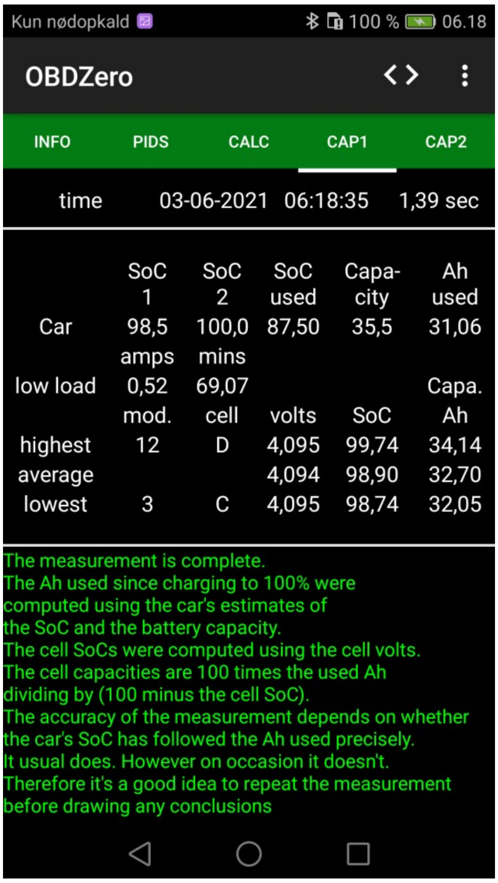
The Model Wh/km is similar to the Battery Wh/km when the heater is off and similar to the Car Wh/km when the heater is on. Again the model requires much more explanation.

The Car cap. Ah at the bottom of the Calc screen is the battery’s 100% capacity as reported by the car. The SoC is SoC 2 as reported by the car. The first Ah is the Car cap. Ah times the SoC divided by 100. The second

Ah is based on the sum of the Ah used. The last cap. Ah is the second Ah divided by the SoC times 100. This is a check sum and it should be close to the Car cap. Ah. It isn't exactly the same because of measurement errors. However if there is a large difference it may indicate that the car's estimate of the battery's 100% capacity has drifted from the true value. I have observed this drift a number of times and it is usually in the downward direction. In other words the car is under estimating the battery capacity.

The values shown on the CALC screen, the WATTS screen, the DRIVE screen and the CAPA screen are stored in the Calc_ file on the phone.

The CAP1 screen



This screen contains a procedure for measuring the present 100% capacity of the battery. The car keeps a running estimate of the battery capacity which as a rule is quite good. However under certain circumstances the car's estimate can be in error. In any event there may be other reasons to want an independent estimate of the present 100% capacity. The procedure for measuring the capacity shown here is much simpler than the procedure performed by the service department at a dealership and therefore less accurate. Computing must be started for this function to work.

The following steps are performed when doing the measurement:

- Before the procedure can begin both SoC1 and SoC2 must be less than 15%, equal to about a rest range of 5 km.
- When the SoC is less than 15% the SoC of all the cells is found. This is based on each cell's voltage and it requires the load on the battery to be less than 1 amp for 30 min.

This procedure is detailed in the lower half of the CAP1 screen. As each step is completed the next step turns green.

The SoC of a cell is computed using the cell's voltage and temperature and a model of the lithium cell. To use this model the cell must be close to equilibrium. Equilibrium means no current flowing through the cell, all charges within the cell are balanced and the temperature is constant. In the real world equilibrium is difficult to achieve therefore we approximate it by keeping the amps below 1 for 30 min. During the 30 minutes you will notice that the cell voltage will either increase or decrease to a more or less constant value.

When the measurement is complete the cells with the maximum and minimum capacities and the average capacity of the cells are shown. These capacities are computed by dividing the amp-hours (Ah) drawn from the battery since it was last charged to 100% divided by the SoC drawn from each cell and times 100. The SoC drawn from the cell is 100% minus the SoC of each cell. The Ah drawn from the battery is the car's estimate of the battery capacity times (100 – SoC2) divided by 100. The Ah drawn is shown as AhUsed in the first line under the date and time. The accuracy of this measurement depends on whether the car has kept an accurate account of the Ah to and from the battery since it was charged to 100%. It does do a fairly good job of keeping this account but errors do occur which will affect the results. On the other hand, this procedure will give a good relative measurement of the cell capacities. This means that if it shows a 2 Ah difference between the highest and lowest cells then there is about 2 Ah difference in the capacity of the cells. This difference will increase as the battery ages. We have two CZero's, one with a present capacity of 39 Ah and a cell difference of 2 Ah and the other with a present capacity of 34 Ah and a cell difference of 3 Ah.

The best way to determine if one cell is defect is to compare the lowest cell to the next lowest cells. This is possible because OBDZero computes all of the cell capacities and records them in the Cell_ file on the phone. Annex 3 in this manual describes a method for working with the Cell_ files in the Notepad program such that all the cell capacities can be read and compared.

For cars built after 2015 cell data is not available. However the car does provide the voltages of the cells with the highest and lowest voltages. Based on these voltages and the battery voltage Cap1 computes the highest, lowest and average cell capacities. Unfortunately there is no information on which cells have the highest and lowest capacities.

OBDZero cannot change the car's estimate of the battery's 100% capacity. However the the service department at a dealership can. This requires the service center to repeat the measurement and they will probably charger a fee for this service.

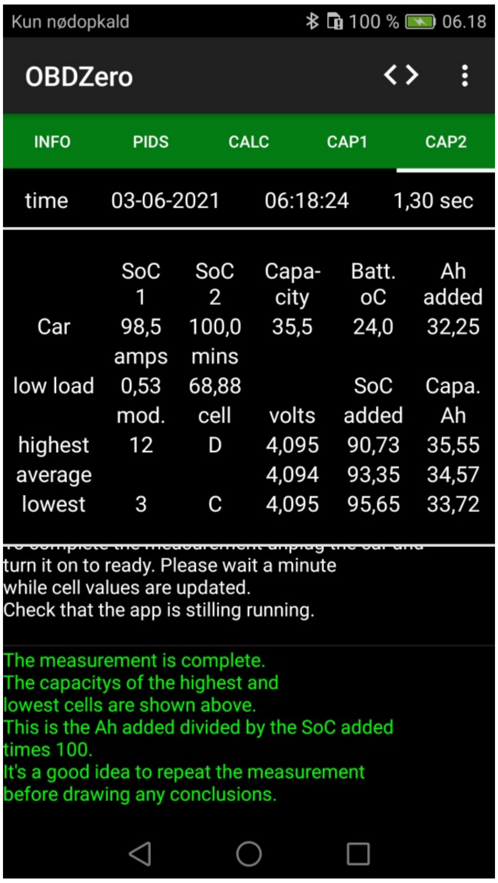
The State of Health (SoH) of the battery is the present Ah capacity divided by the battery's original capacity times 100. The battery's Ah capacity when it left the battery assembly line was 50 Ah. So the SoH of the battery shown above is:

71 % = 100 * 35.5/50

Even though it isn't in use the battery looses capacity in transport to the car factory and in the car on it's way to the dealership. For this reason the highest observed capacity in a car is about 48 Ah or 96% SoH.

Any feedback to me at DavidCecil@outlook.dk about this procedure will be appreciated.

The CAP2 screen



This screen contains a more extensive procedure for measuring the present 100% capacity of the battery than that shown on the Cap1 screen. As mentioned before the car keeps a running estimate of the battery capacity which as a rule is quite good but it can be in error. The procedure for measuring the capacity shown here is similar to the procedure performed by the service department at a dealership and is therefore more rigorous and more accurate than the Cap1 procedure. The first two steps of the procedure are the same as for Cap1. Therefore you can continue with a Cap2 measurement immediately after seeing the results of the Cap1 measurement. As with Cap1 computing must be running for this function to work.

The following steps are performed when doing the measurement:

- Before the procedure can begin both SoC1 and SoC2 must be less than 15%, equal to about a rest range of 5 km.
- When the SoC is less than 15% the SoCs of the cells are measured. This requires the load on the battery to be less than 1 amp for 30 min. The SoC's of the lowest and highest cells are shown.
- Now the car battery must be charged up to 100%.
- Once charging is complete the SoCs of the cells are measured again. As before this requires the load on the battery to be less than 1 amp for 30 min. This 30 min period starts as soon as charging is completed.
- As a rule the 30 min have passed when one returns to the car after charging. Therefore the only step necessary to complete the measurement is to turn the car on so that the voltages and SoCs of all the cells can be read.

This procedure is detailed in the lower half of the CAP2 screen. As each step is completed the next step turns green.

The cell capacities are the amp-hours (Ah) added to the battery during charging divided by the change in the SoC of each cell and times 100. The app keeps track of both.

The SoC of a cell is computed using the cell's voltage and temperature and a model of the lithium cell. To use this model the amps through the battery must be less than 1 for 30 min. During the 30 minutes you will notice that the cell voltage will either increase or decrease to a more or less constant value.

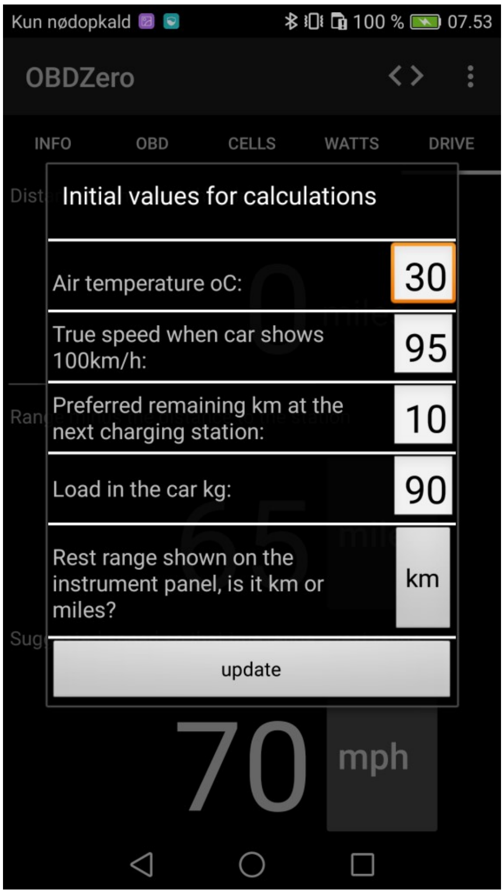
As with Cap1 the best way to determine if one cell is defect is to compare the lowest cell to the next lowest cells. This is possible because Cap2 also computes all of the cell capacities and records them in the Cell_ file on the phone. Annex 3 in this manual describes a method for working with the Cell_ files in the Notepad program such that all the cells can be read and compared.

For cars built after 2014 cell data is not available. However the car does provide the voltages of the cells with the highest and lowest voltages. Based on these voltages and the battery voltage Cap2 computes the highest, lowest and average cell capacities. Unfortunately there is no information on which cells have the highest and lowest capacities.

OBDZero cannot change the car's estimate of the battery's 100% capacity. However the the service department at a dealership can. This requires the service center to repeat the measurement and they will probably charger a fee for this service.

Any feedback to me at DavidCecil@outlook.dk about this procedure will be appreciated.

Initial Values



This screen shows the default initial values. The app will work satisfactorily with these values under most conditions. However the accuracy of many of the calculations mention previously will improve if these values are updated.

The air temperature is used to compute the air density which is a variable in the calculation of the air resistance of the car. The correct air temperature will give a marginal improvement in the car model used to compute the suggested speed in the DRIVE screen.

The true speed when the indicated speed is 100 km/h is used to compute the correction factor for the updated remaining distance to the next charging station. This distance is shown on the DRIVE screen. This correction has a noticeable effect on the accuracy of this distance. If you observe an error in the remaining distance compared to the true distance to the station adjusting this speed will correct the error. If the remaining distance is greater than 0 when you arrive at the charging station then add 1 km/h to the true speed. If on the other hand the remaining distance goes to zero before arriving at the charging station then subtract 1 km/h. By aware that there will always be small errors in the remaining distance so don't spend much time on this calibration unless the error is large. Also true speed cannot be changed while the app is collecting data so each correction is made before the next trip.

The preferred remaining km is added to the remaining distance to the charging station when computing the suggested speed to the next charging station. This can be set to 0 but I don't.

The load in the car is the sum of the passengers and the baggage in the car. The load plus the empty weight of the car is used to calculate the rolling resistance and the watts for acceleration in the car model. Errors in this value are usually small in relation to the total weight of the car. However when the car is fully loaded this can be significant.

For cars sold in countries where miles are in common use the rest range shown on the instrument panel can be in miles rather than km. As yet there is no simple way for OBDZero to detect if this is the case. Many of the functions in the app depend on knowing if the rest range is in miles or km. This is particularly true for the Drive function. Therefore the driver must tell the app which it is. By pressing the button to the right of the text the units came be changed from km to miles and back again. Note that the preferred remaining km doesn’t change when the units for the rest range change. The driver must convert his or her preferred miles to km before entering this value.

The initial values are stored in the OBD_ file on the phone

Corrections and additions to this user manual

If you have questions or you believe there are errors in this manual please contact me.
There are many subjects that require further explanation e.g. the model of the car or the model of the cell. I plan on adding annexes to the manual with more detailed descriptions including the equations used. This will take time.

Acknowledgements and references

I have already mentioned the information on the rules for the conversion of PIDs to data made available by users on the iMiev forum. I couldn't have written this app without their help.

The code for OBDZero is built on the Bluetooth terminal program, BlueTerm, by pymasde.es found on GitHub and dated the 7th May 2014

The technical manuals for the iMiev can be found at http://mmc-manuals.ru/manuals/i-miev/online/Service_Manual/2013/index_M1.htm

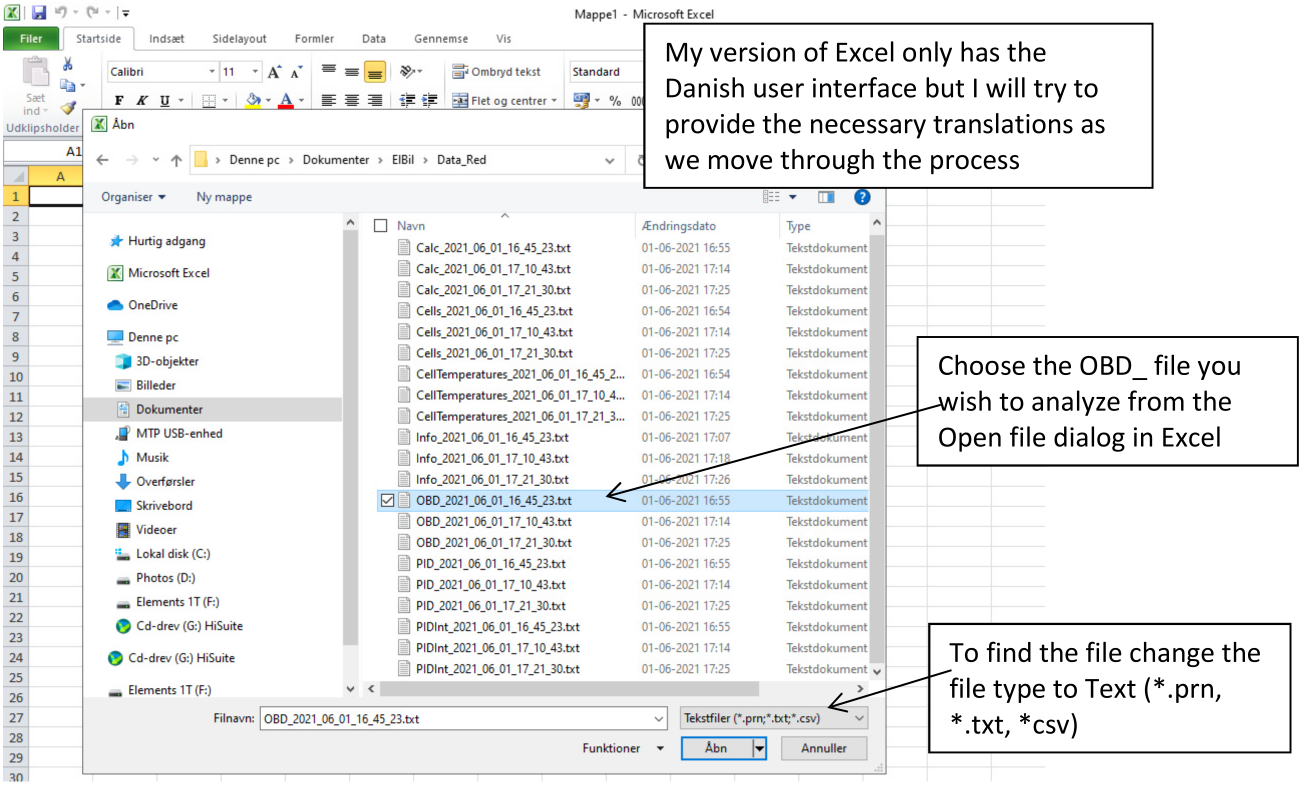
The AT commands for the OBD dongles are found in ELM327DSH.pdf from www.elmelectronics.com

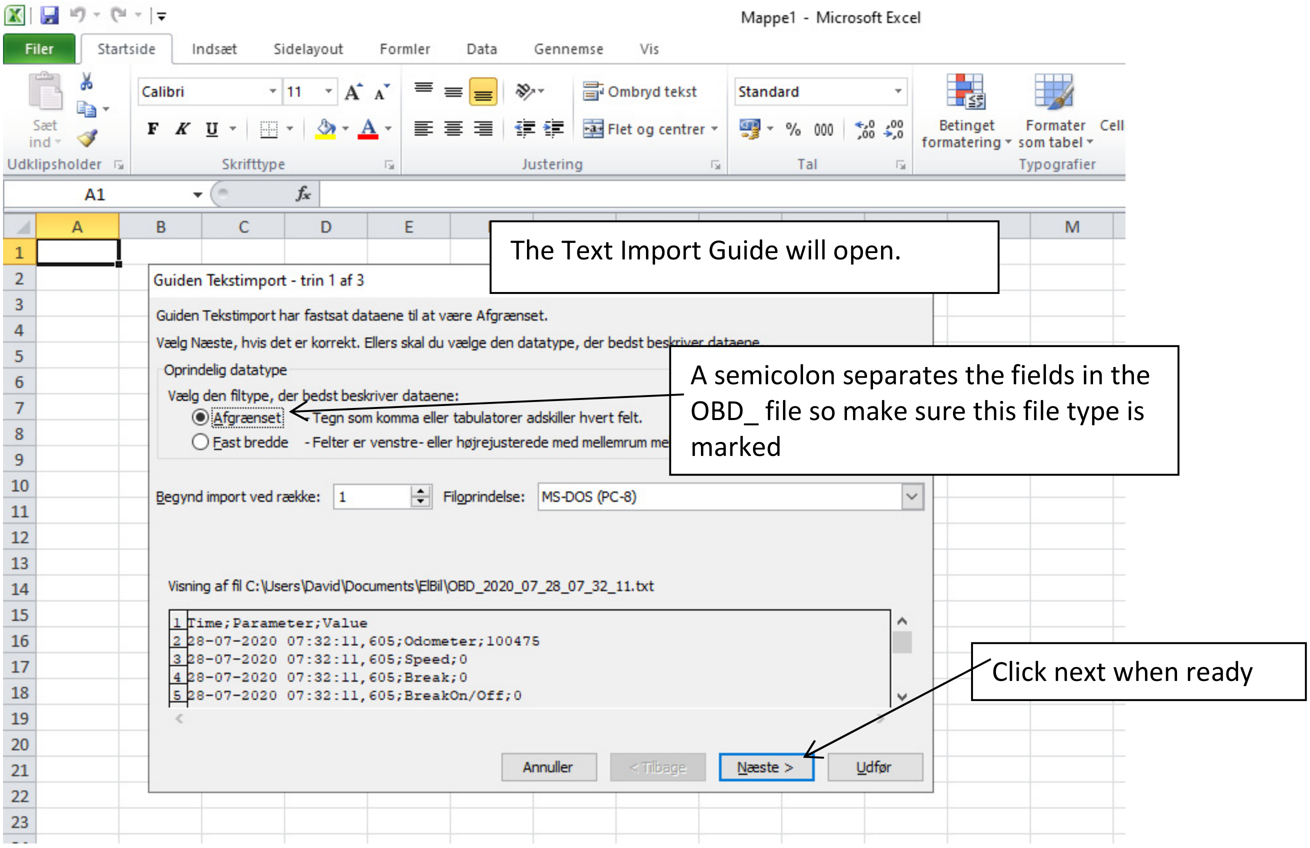
A manual for the OBDLink LX can be found at https://www.scantool.net/downloads/98/stn1100-frpm.pdf

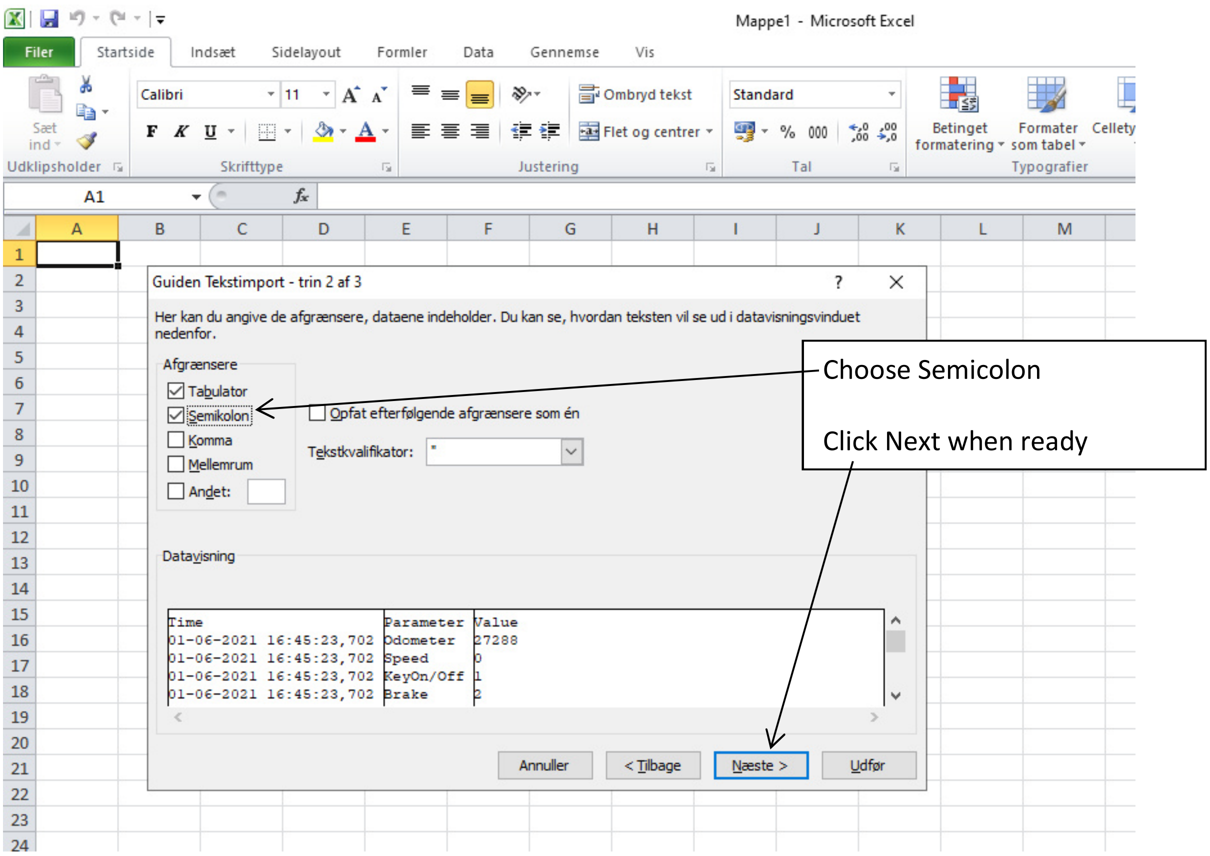
Thanks to the following users of OBDZero who have pointed out errors and made helpful suggestions: Richi, Andon, Kruno, Jason, Albert, Paolo, Florian, DginiSG, Greg, Toby, Tibor and Allan.

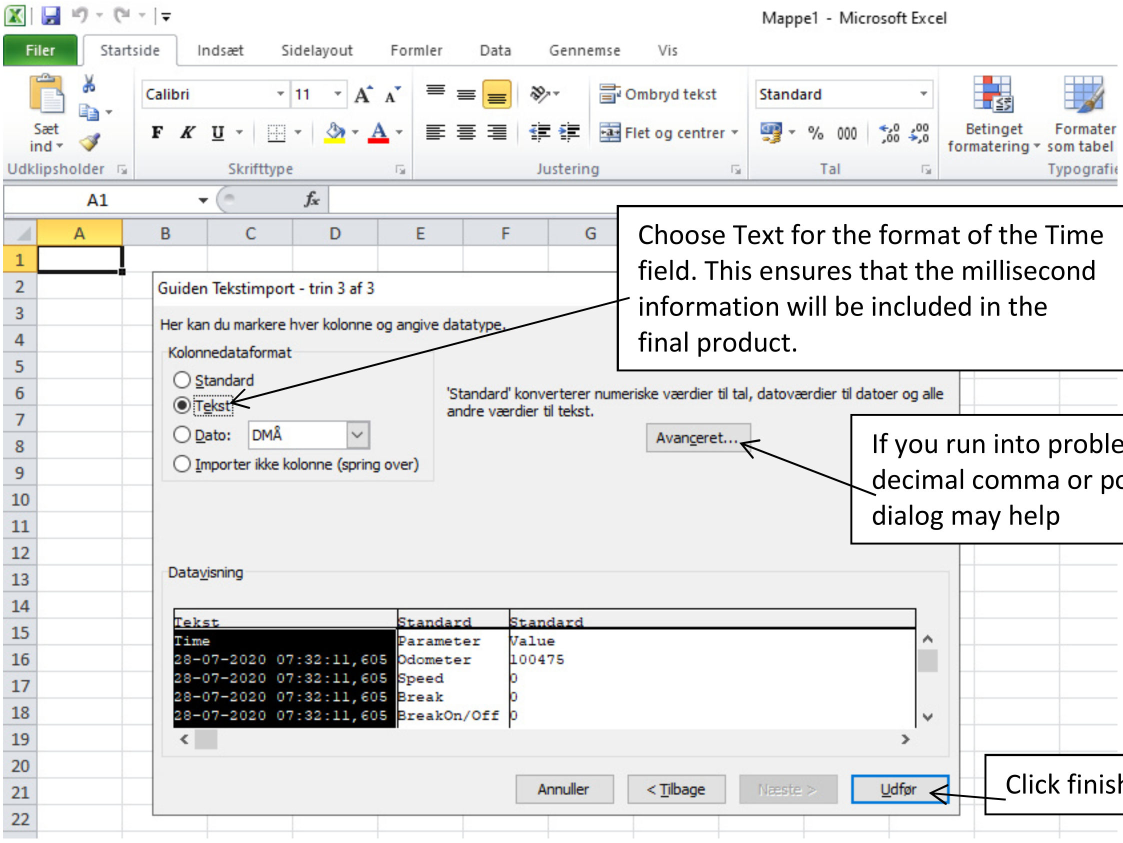
Annex 1 Working with the OBD_ file in Excel

The first step is to copy the data files from your phone to a folder on your computer. Connect the phone to the PC with a USB cable and choose file transfer. Different phones have different ways to choose file transfer. Open Windows Explorer, find the OBDZero folder on the phone and then copy the txt files to a folder on the PC. Once there the OBD_ file can be opened in Excel.




The Text Import Guide will open.
A semicolon separates the fields in the OBD_ file so make sure this file type is marked
Click next when ready
Guiden Tekstimport - trin 1 af 3
Guiden Tekstimport har fastsat dataene til at være Afgrænset.
Vælg Næste, hvis det er korrekt. Ellers skal du vælge den datatype, der bedst beskriver dataene.
Oprindelig datatype
Vælg den filtype, der bedst beskriver dataene:
Afgrænset - Tegn som komma eller tabulatorer adskiller hvert felt.
Fast bredde - Felter er venstre- eller højrejusterede med mellemrum
Begynd import ved række: 1
Filoprindelse: MS-DOS (PC-8)
Visning af fil C:\Users\David\Documents\ElBil\OBD_2020_07_28_07_32_11.txt
Time;Parameter;Value
28-07-2020 07:32:11,605;Odometer;100475
28-07-2020 07:32:11,605;Speed;0
28-07-2020 07:32:11,605;Break;0
28-07-2020 07:32:11,605;BreakOn/Off;0
Annuller
< Tilbage
Næste >
Udfør

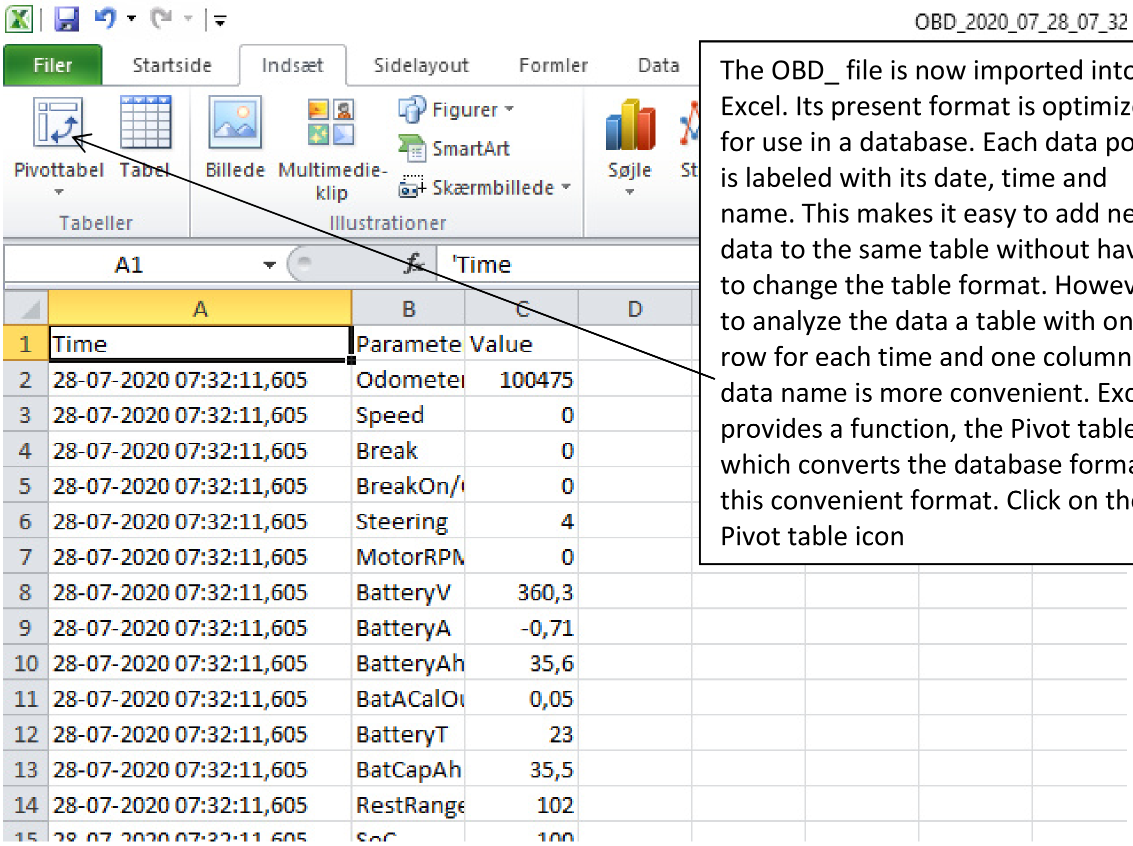

Choose Semicolon
Click Next when ready
Guiden Tekstimport - trin 2 af 3
Her kan du angive de afgrænsere, dataene indeholder. Du kan se, hvordan teksten vil se ud i datavisningsvinduet nedenfor.
Afgrænsere
Tabulator
Semikolon
Komma
Mellemrum
Andet:
Opfat efterfølgende afgrænsere som én
Tekstkvalifikator: "
Datavisning
Time Parameter Value
01-06-2021 16:45:23,702 Odometer 27288
01-06-2021 16:45:23,702 Speed 0
01-06-2021 16:45:23,702 KeyOn/Off 1
01-06-2021 16:45:23,702 Brake 2
Annuller
< Tilbage
Næste >
Udfør



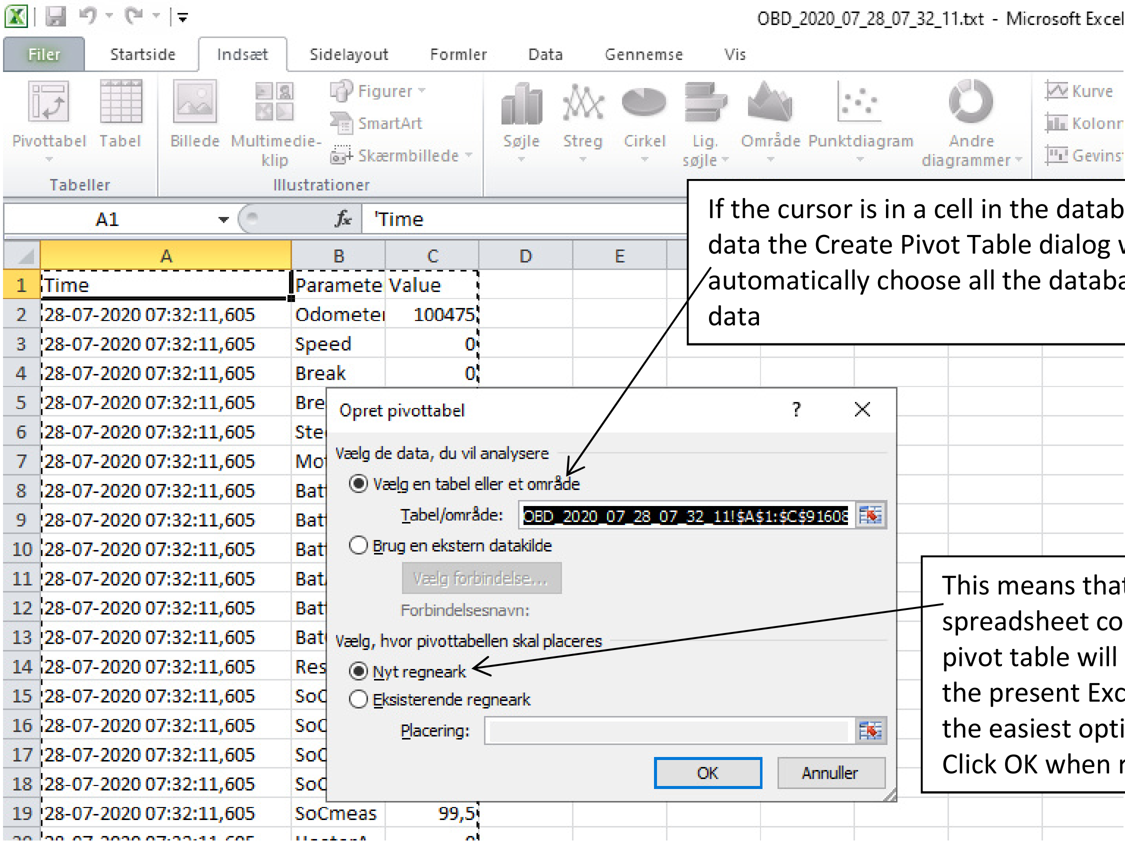
Choose Text for the format of the Time field. This ensures that the millisecond information will be included in the final product.

If you run into problems with the decimal comma or point the Advanced dialog may help

Click finish when ready.

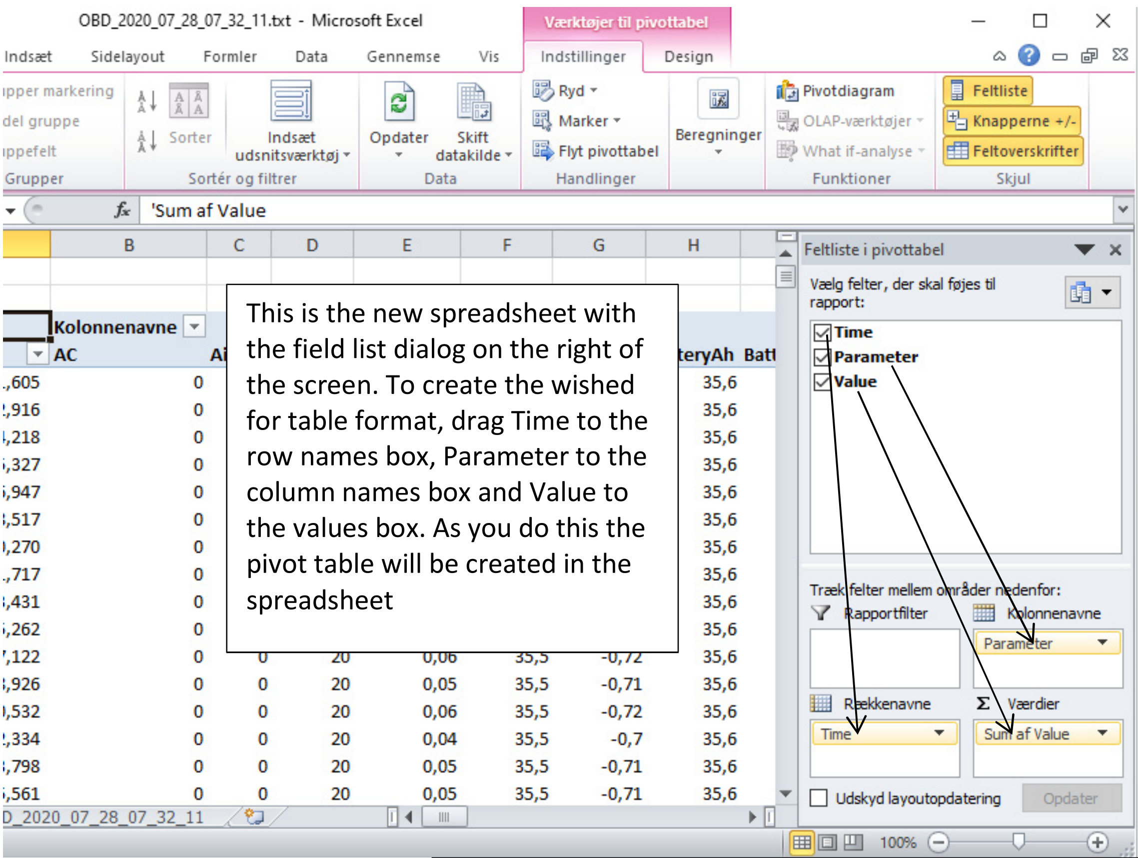


The OBD_ file is now imported into Excel. Its present format is optimized for use in a database. Each data point is labeled with its date, time and name. This makes it easy to add new data to the same table without having to change the table format. However to analyze the data a table with one row for each time and one column for data name is more convenient. Excel provides a function, the Pivot table, which converts the database format to this convenient format. Click on the Pivot table icon



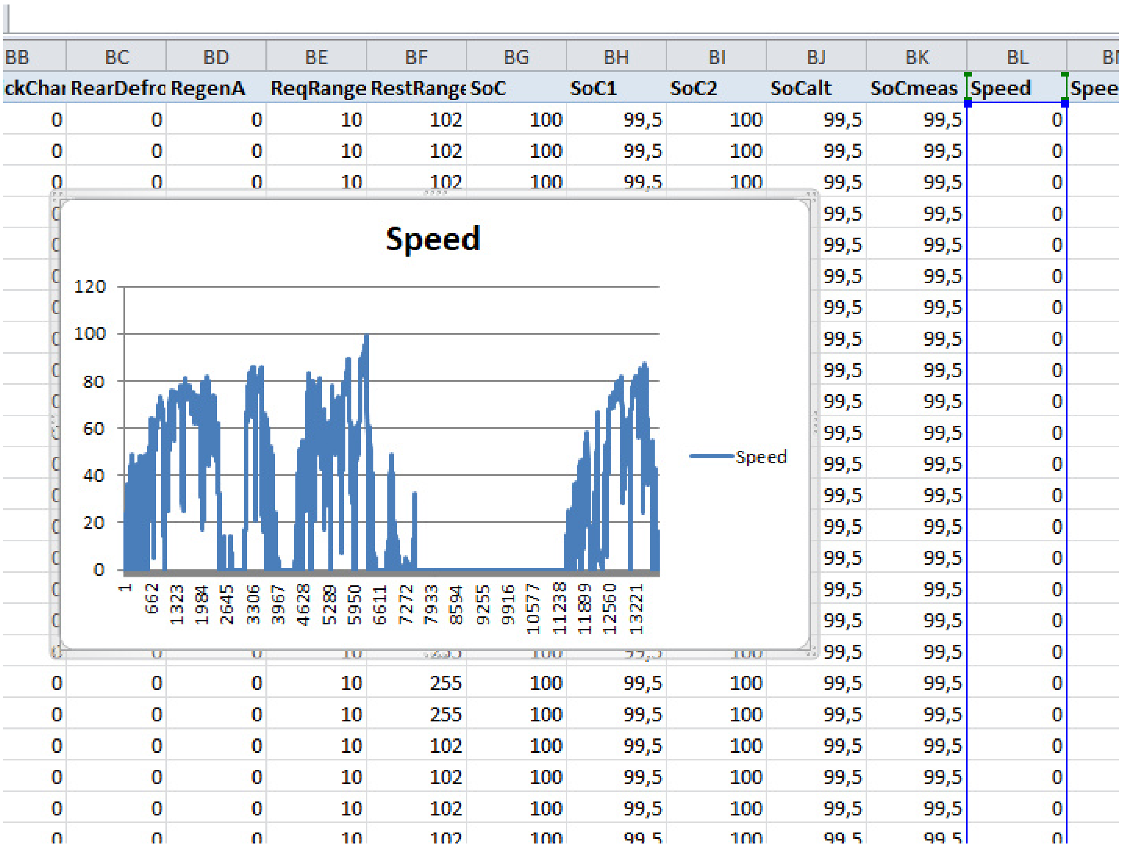
If the cursor is in a cell in the database data the Create Pivot Table dialog will automatically choose all the database data

This means that a new spreadsheet containing the pivot table will be created in the present Excel file. This is the easiest option.
Click OK when ready.



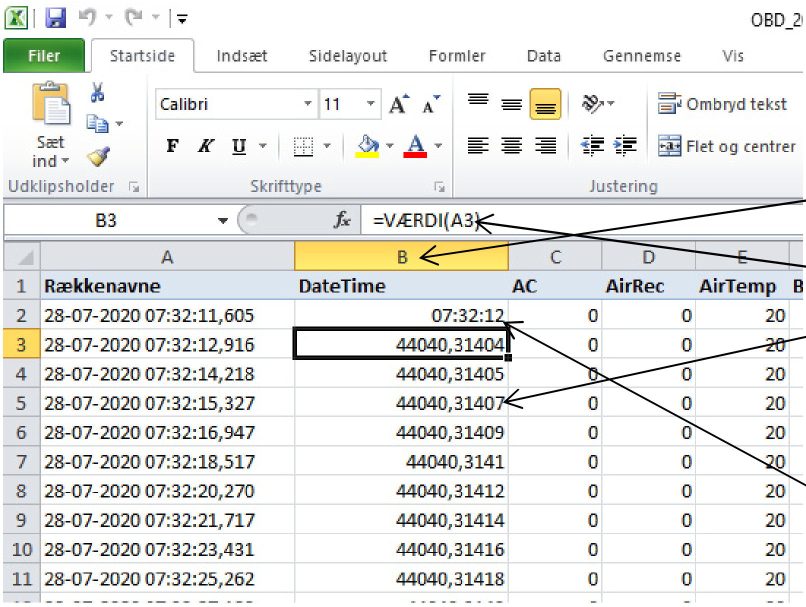
This is the new spreadsheet with the field list dialog on the right of the screen. To create the wished for table format, drag Time to the row names box, Parameter to the column names box and Value to the values box. As you do this the pivot table will be created in the spreadsheet

Note that value is the sum of the value. It isn't possible to choose the value alone and in this case it makes no difference. There is only one value for each time and parameter name so the sum of the value is the same as the value.



Once the pivot table is finished I like to copy the result to a third spreadsheet before creating graphs of the data. This is because the pivot table has some built-in rules that make it different from normal columns of data in Excel. Don't copy the totals along the bottom row and the right hand column of the pivot table. They have no meaning and they only create problems when graphing the data.

There is one last step and that is converting the Time from a text to a number in Excel's date and time system.



In the spreadsheet shown above I have added a column between columns containing the row names (A) and the first column of data (B). In this column I used the value function to convert the date and time in column A to the Excel date series number in the new column, column B. These are the numbers starting in the third row and copied down to convert all the times. Finally I changed the format of the first cell under DateTime to a time format. This can also be copied.

Using the DateTime column together with a data column you can create xy graphs with time on the x axis.

While we are on the subject of time there is a limit we need to be aware of. In my version of Excel there are 1048576 rows. In the OBD_ file there are 63 data points for each time. The shortest time step when the OBDZero collects data is 1 second. 1048576 divided by 63 times 1 second is 16644 seconds or 4.6 hours. If you are using the fastest dongle the OBD_ file will grow beyond the number of rows in Excel after about 4½ hours. With the slowest dongle that time can increase to 16 hours. In other words this isn't normally a problem when collecting data while driving. However while using the Cap 2 function OBDZero has to run for up to 8 hours. If you do have this problem when trying to work with date from a Cap2 measurement, please contact me for help.

Annex 2 OBD data defined

The following is a description of the data recorded in the OBD_ file

Parameter	Units	Description	Source	Conversion
AC	on/off	Airconditioning	PID 3A4	byte(0) bit 128
ACAmps	amps	Airconditioning amps	PID 384	(byte(0) * 256 + byte(1))/1000
AirRec	on/off	Air recirculation	PID 3A4	byte(0) bit 64
AirTemp	oC	Air temperature	Initial values	
BatACalOut	amps	Amps out of the battery calibrated	-(BatteryA + 0.66)	
BatCapAh	amp-hours	Battery 100% capacity	PID 374	byte(6)/2
BatteryA	amps	Amps in to the battery (both + & -)	PID 373	(byte(2)*256 + byte(3) -32768)/100
BatteryT	oC	Average cell temperature	Cells averaged	
BatteryTmax	oC	Maximum cell temperature	PID 374	byte(4) – 50
BatteryTmin	oC	Minimum cell temperature	PID 374	byte(5) – 50
BatteryV	volts	Battery volts	PID 373	(byte(4)*256 + byte(5))/10
Brake	unknown	Brake pressure	PID 208	byte(3)
BrakeOn/Off	on/off	Brake	PID 231	byte(4)
CellVmaxCell	number	Max volts cell number	Cells sorted	
CellVmaxMod	number	Max volts cell module	Cells sorted	
CellVmaxTemp	oC	Max volts cell temperature	Cells sorted	
CellVmaxVolt	volts	Max volts cell volts	Cells sorted	
CellVminCell	number	Min volts cell number	Cells sorted	
CellVminMod	number	Min volts cell module	Cells sorted	
CellVminTemp	oC	Min volts cell temperature	Cells sorted	
CellVminVolt	volts	Min volts cell volts	Cells sorted	
CellVsum	volts	Sum cell voltages	Cells summed	
Charge12Amps	amps	12volt battery charger amps in	PID 384	byte(3)/100
ChargeADC	amps	Charging unit amps to battery	PID 389	byte(2)/10
ChargeTemp1	oC	Charging unit temperature 1	PID 389	byte(3) – 50
ChargeTemp2	oC	Charging unit temperature 2	PID 389	byte(4) – 50
ChargeVAC	volts	Charging unit volts from mains	PID 389	byte(1)
ChargeVDC	volts	Charging unit volts to battery	PID 389	2*byte(0) + 1
ChargeAAC	amps	Charging unit amps from mains	PID 389	byte(6)/10
FanDirect	number	Air direction control position	PID 3A4	byte(1) low nibble
FanMax	on/off	Max fan	PID 3A4	byte(0) bit 32
FanSpeed	number	Fan speed	PID 3A4	byte(1) high nibble
Gear	number	3 = P/N, 1 = Reverse, 4 = Drive	PID 236	see note 2
Heat/Cool	number	Temperature control position	PID 3A4	byte(0) low nibble
HeaterA	amps	Heater amps	PID 384	byte(4)/10
KeyOn/Off	on/off	Key on (either ready or not ready)	PID 412	byte(0) = 254
LDrive	on/off	Headlights	PID 424	byte(1) bit 32
LFrontFog	on/off	Front fog lights	PID 424	byte(0) bit 8
LHigh	on/off	Highbeams	PID 424	byte(1) bit 4
Loadkg	kg	Passengers and baggage	Initial values	
LPark	on/off	Parking lights	PID 424	byte(1) bit 64
LRearFog	on/off	Rear fog lights	PID 424	byte(0) bit 16
Margin	km	Preferred km at next charging station	Initial values	

Parameter	Units	Description	Source	Conversion
MotorA	amps	Motor current	PID 696	(byte(2)*256 + byte(3) - 500)/20
MotorRPM	rpm	Motor rpm	PID 298	byte(6)*256 + byte(7) – 10000
MotorTemp0	oC	Motor temperature 0	PID 298	byte(0) – 50
MotorTemp1	oC	Motor temperature 1	PID 298	byte(1) – 50
MotorTemp2	oC	Motor temperature 2	PID 298	byte(2) – 50
MotorTemp3	oC	Motor temperature 3	PID 298	byte(3) – 50
Odometer	km	Total km driven	PID 412	byte(2)*256*256 + byte(3)*256 + byte(4)
PIDCount	number	Number of new PIDs received	program counter	
QuickCharge%	%	Chademo percent	PID 697	byte(1)
QuickChargeA	amps	Chademo current	PID 697	byte(2)
QuickChargeOn/Off	on/off	Chademo connected	PID 697	byte(0)
RearDefrost	on/off	Rear window defrost	PID 424	byte(6) bit 8
RegenA	amps	Regeneration amps (negative)	PID 696	(byte(6)*256 + byte(7) - 10000)/5
RestRange	km	Car's estimate of remaining km	PID 346	byte(7)
SoC1	%	State of Charge 1	PID 374	(byte(0) - 10)/2
SoC2	%	State of Charge 2	PID 374	(byte(1) - 10)/2
Speed	km/h	Indicated speed	PID 215	(byte(0)*256 + byte(1))/128
Speed100	km/h	True speed when 100 km/h indicated	Initial values	
Steering	degrees	Steering wheel position	PID 236	(byte(0)*256 + byte(1) -4096)/2
WindWiper	on/off	Windshield wipers	PID 424	byte(1) bit 8

Note 1
Each PID contains 1 to 8 bytes. The first byte is numbered 0 and the last byte is numbered 7. When on/off information is code it is usually coded in the bits of a byte, a bit equal to 1 means on. The bits are numbered in the table above by positional value in the byte. Of the 8 bits in a byte bit 128 is the most significant and bit 1 the least significant. The low nibble in a byte is the number represented by the 4 least significant bits and the high nibble is the number represented by the 4 most significant bits.

Note 2
The gear position is a bit complicated. The information is contained in PID 285. If byte(6) equals 12 then the position is either Park or Neutral with no indication of which. If byte(6) equals 14 and byte(7) equals 16 then the position is Drive. If byte(6) equals 14 and byte(7) doesn’t equal 16 then the position is Reverse. There is probably another PID where the gear position is more explicit, but I haven’t found it yet. Another thing, the iMiev has two more gear positions than the CZero and the iOn. Both these positions are shown as Drive in OBDZero.

Note 3
The coding of the individual cell information is found in PIDs 6E1-6E4. Byte(0) is the module number. A more detailed explanation will have to wait for a later manual.

Note 4
The BatteryA, the current to the battery can be both positive and negative. The sign is in relation to the battery so charging is positive and discharging is negative. However most of the computations in OBDZero are from the perpective of the motor and auxiliary functions. So for BatACalOut, the calibrated current, discharge is positive and charging is negative. MotorA, motor current is always positive and RegenA, the regenaration current is always negative. HeaterA, heater current and ACAmps, airconditioning current are both positive. However this convention doesn’t apply to ChargeADC, charging unit current to the battery and QuickChargeA, Chademo current to the battery both of which are positive.

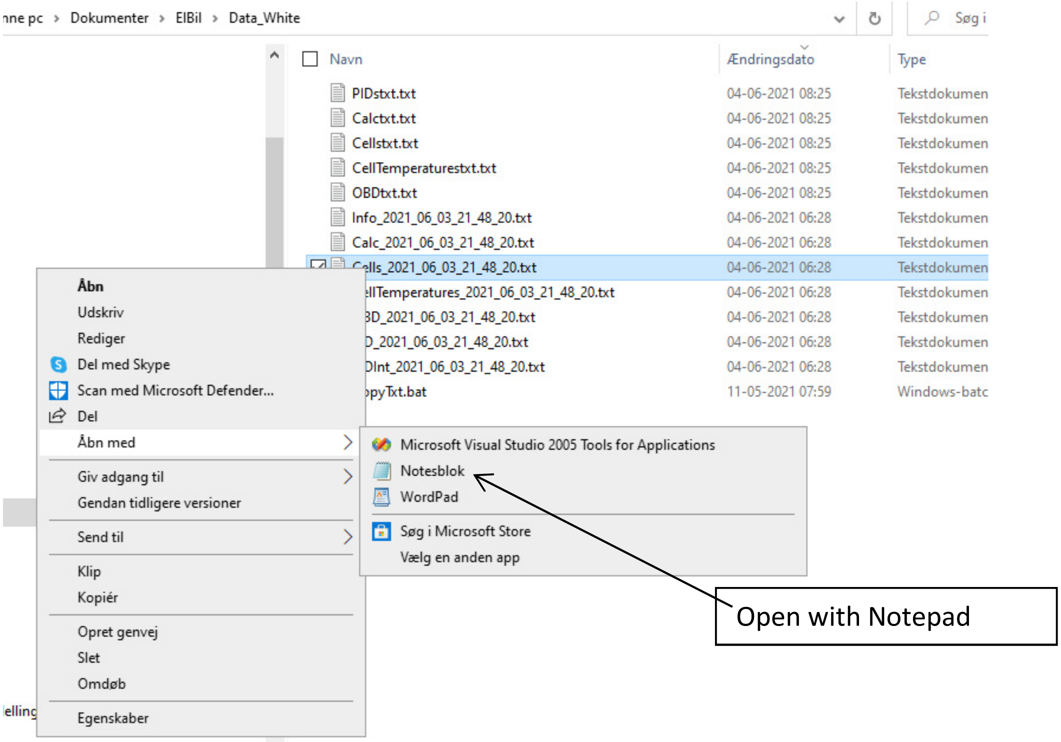
Note 5

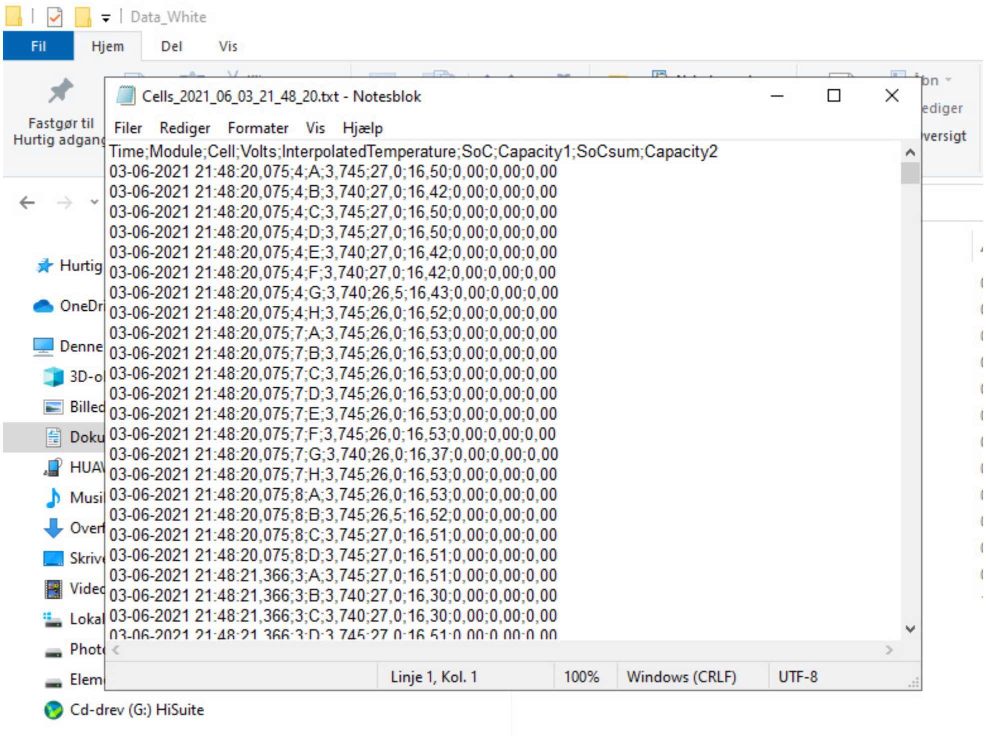
Earlier versions of OBDZero also saved the following parameters. These parameters have been replaced as shown in the table.

Parameter	**Replaced by**	**Reason**
Break	Brake	Spelling error
BreakOn/Off	BrakeOn/Off	Spelling error
SoC	SoC2	More specific
SoCalt	SoC1	More specific
SoCmeas	SoC1	More specific
CellVmaxVolts	CellVmaxVolt	Shorter, saves space in the database
CellVminVolts	CellVminVolt	Shorter, saves space in the database
ReqRange	Margin	More specific
BatteryAh	BatCapAh * SoC2/100	Compute when needed, saves space
ACW	ACAmps * BatteryV	Compute when needed, saves space

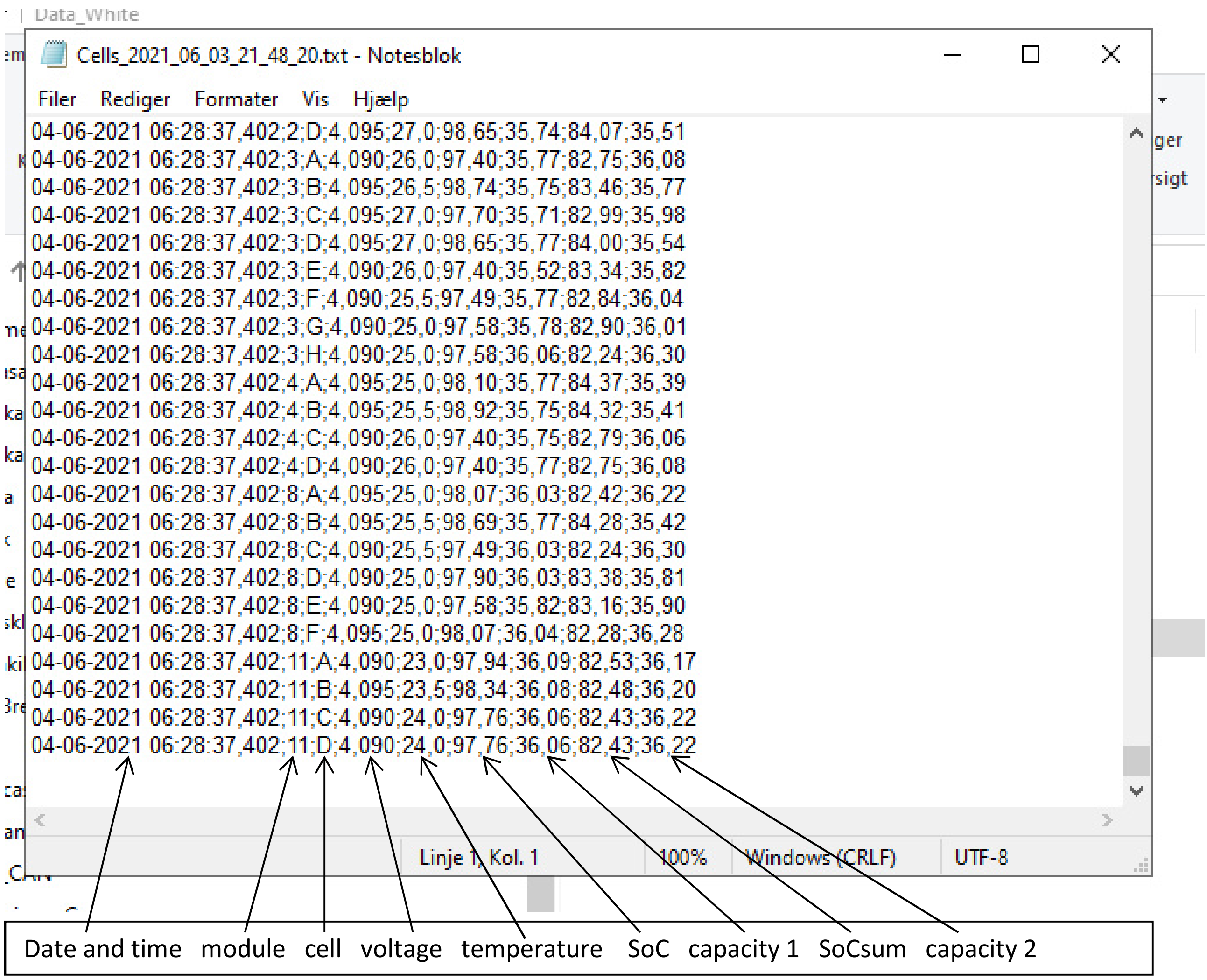
Annex 3. Reading the capacities of individual cells in the Cell_ text file

Once Cap1 or both Cap1 and Cap2 measurements are complete and the app is closed the capacity results can be found at the end of the Cells_ file. As with the OBD_ file, the first step is to copy the data files from your phone to a folder on your computer. Connect the phone to the PC with a USB cable and choose file transfer. Open Windows Explorer, find the OBDZero folder on the phone and then copy the txt files to a folder on the PC. Once there the Cells_ file can be opened in Notepad. In this case, Notepad is easier than using Excel.





Scroll down to the end of the file and note of the values for each cell. As a rule you will find all 88 (80) cells in the last 10 time steps.


Cells_2021_06_03_21_48_20.txt - Notesblok

Filer Rediger Formater Vis Hjælp

04-06-2021 06:28:37,402;2;D;4,095;27,0;98,65;35,74;84,07;35,51
04-06-2021 06:28:37,402;3;A;4,090;26,0;97,40;35,77;82,75;36,08
04-06-2021 06:28:37,402;3;B;4,095;26,5;98,74;35,75;83,46;35,77
04-06-2021 06:28:37,402;3;C;4,095;27,0;97,70;35,71;82,99;35,98
04-06-2021 06:28:37,402;3;D;4,095;27,0;98,65;35,77;84,00;35,54
04-06-2021 06:28:37,402;3;E;4,090;26,0;97,40;35,52;83,34;35,82
04-06-2021 06:28:37,402;3;F;4,090;25,5;97,49;35,77;82,84;36,04
04-06-2021 06:28:37,402;3;G;4,090;25,0;97,58;35,78;82,90;36,01
04-06-2021 06:28:37,402;3;H;4,090;25,0;97,58;36,06;82,24;36,30
04-06-2021 06:28:37,402;4;A;4,095;25,0;98,10;35,77;84,37;35,39
04-06-2021 06:28:37,402;4;B;4,095;25,5;98,92;35,75;84,32;35,41
04-06-2021 06:28:37,402;4;C;4,090;26,0;97,40;35,75;82,79;36,06
04-06-2021 06:28:37,402;4;D;4,090;26,0;97,40;35,77;82,75;36,08
04-06-2021 06:28:37,402;8;A;4,095;25,0;98,07;36,03;82,42;36,22
04-06-2021 06:28:37,402;8;B;4,095;25,5;98,69;35,77;84,28;35,42
04-06-2021 06:28:37,402;8;C;4,090;25,5;97,49;36,03;82,24;36,30
04-06-2021 06:28:37,402;8;D;4,090;25,0;97,90;36,03;83,38;35,81
04-06-2021 06:28:37,402;8;E;4,090;25,0;97,58;35,82;83,16;35,90
04-06-2021 06:28:37,402;8;F;4,095;25,0;98,07;36,04;82,28;36,28
04-06-2021 06:28:37,402;11;A;4,090;23,0;97,94;36,09;82,53;36,17
04-06-2021 06:28:37,402;11;B;4,095;23,5;98,34;36,08;82,48;36,20
04-06-2021 06:28:37,402;11;C;4,090;24,0;97,76;36,06;82,43;36,22
04-06-2021 06:28:37,402;11;D;4,090;24,0;97,76;36,06;82,43;36,22

Linje 1, Kol. 1 100% Windows (CRLF) UTF-8

The Date and Time is day, month, year, hour, minute, and seconds with microseconds. The SoC is the present value. If only Cap1 was performed then this will be less than 15%. If both Cap1 and Cap2 were performed then SoC should be close to 100%. Capacity 1 is the result of the Cap1 function. SoCsum is used to compute capacity 2. Lastly capacity 2 is the result of the Cap2 function. If only Cap1 was performed then these last two number with be 0 or close to 0. Capacity 1 and capacity 2 are almost always different. Capacity 2 is more accurate than capacity 1. However, both can be used to compare the capacities of the cells.

OBDZero cannot change the car's estimate of the battery's 100% capacity. However the the service center at a dealership can. This requires the service center to repeat the measurement and their will probably a charger a fee for this service.