Collection, analysis and presentation of Shubnikov-de Haas oscillation data

Using the Jupyter environment

Abi Graham, Olivia Armitage, Anthony Matthews

Quantum Design

OXFORD

Contents

1 Introduction

In this application note the entire measurement lifecycle: from making measurements on a sample; through managing and analysing the data generated; to producing reports for the dissemination of results is described.

All of the steps outlined were taken within the Jupyter environment provided with the Quantum Design Oxford (QDO) measurement server included with **Teslatron**PT Plus.

1.1 Measurement background

Low temperatures and high magnetic fields are essential for revealing the intrinsic behaviour of charge carriers in low-dimensional, correlated, and topological materials [1] [2] [3]. Under these conditions, quantum effects become dominant, enabling the observation of phenomena such as the integer and fractional quantum Hall effects in two-dimensional electron gases (2DEGs) [4] [5], as well as quantum oscillations that provide insight into the effective mass of charge carriers and their mobility [6].

At high magnetic fields, energy levels become quantised into Landau levels and carriers are forced into cyclotron motion [6]. As the magnitude of the perpendicular magnetic field, $B_{\perp} = \vec{B} \cdot \hat{n}$, where $\hat{n}$ is the unit vector normal to the 2DEG plane, increases the spacing and degeneracy of these Landau levels grows; causing them to intersect with the Fermi level and produce variation in the accessible density of states that directly modulates the electrical conductivity. The resistance oscillates periodically as a function of $1/B_{\perp}$ and these oscillations are known as Shubnikov-de Haas (SdH) oscillations [7].

This inherent quantum behaviour motivates the study of the magnetotransport properties of charge carriers. The amplitude and temperature dependence of SdH oscillations are described by the Lifshitz–Kosevich formalism [8], which allows extraction of key parameters such as the effective mass and quantum lifetime.

The effective mass of charge carriers controls both the electrical and optical behaviour of a material. As modern devices rely on precise control of carrier motion, knowing the effective mass, and understanding whether it changes as a function of magnetic field, is important for predicting device performance and designing new and novel meta-materials [9].

Here, we study the magnetotransport properties of a 2DEG constrained in an $\mathrm{AlN}/\mathrm{GaN}$ heterostructure which displays well defined SdH oscillations at high magnetic-fields and low temperatures. Subsequent

analysis of these SdH oscillation data is presented which allows the effective mass and quantum lifetime of the carriers to be determined.

1.2 Sample

The 2DEG is patterned into a Greek cross shape (one of the best van der Pauw (vdP) geometries [10] for minimising finite-contact errors [11]). Au contact pads are deposited at the end of each arm, as shown in figure 1a). The sample is mounted in a 20-pin leadless chip carrier (LCC20) and wire bonds connect the four contact pads to individual LCC20 contacts, as shown in figure 1b). Adjacent LCC20 contacts are avoided so that each sample connection can be routed through its own individually shielded twisted pair, configured to be used as a single sample connection in driven-guard mode, as depicted in figure 2.

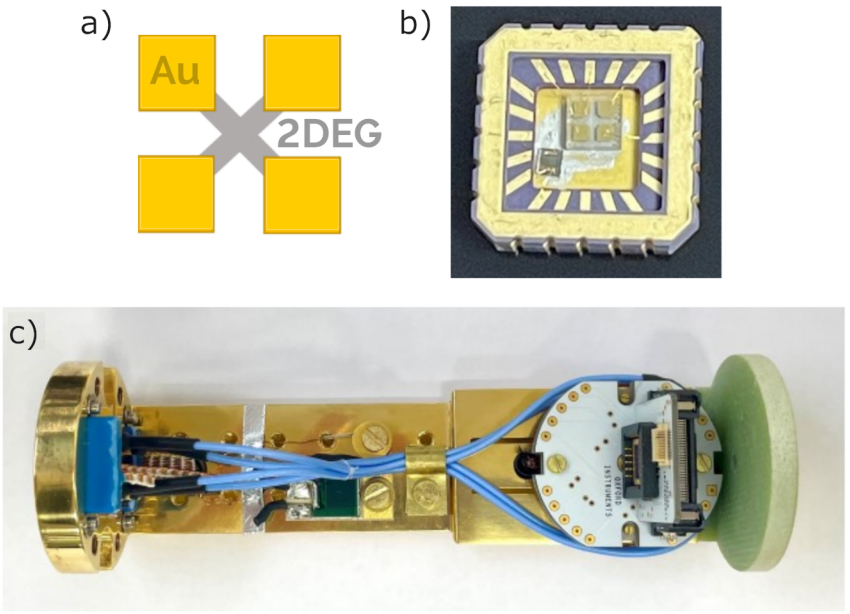

Figure 1: AlN/GaN heterostructure. a) Schematic of 2DEG sample geometry and Au contact pads. b) Photograph of sample mounted in leadless chip carrier. c) Photograph of sample holder mounted perpendicular to the magnetic field direction on the QDO Electrical Transport Fixed Cell.

The LCC20 is straightforward to mount to the QDO samples holders, which can orient the sample either parallel or perpendicular to the magnetic field direction. The sample holder is fixed to the end of a QDO Electrical Transport Fixed cell, as shown in figure 1c) for measurements perpendicular to magnetic field. The cell can then be connected to the end of the Universal Measurement Probe, ready to be loaded into the cryostat.

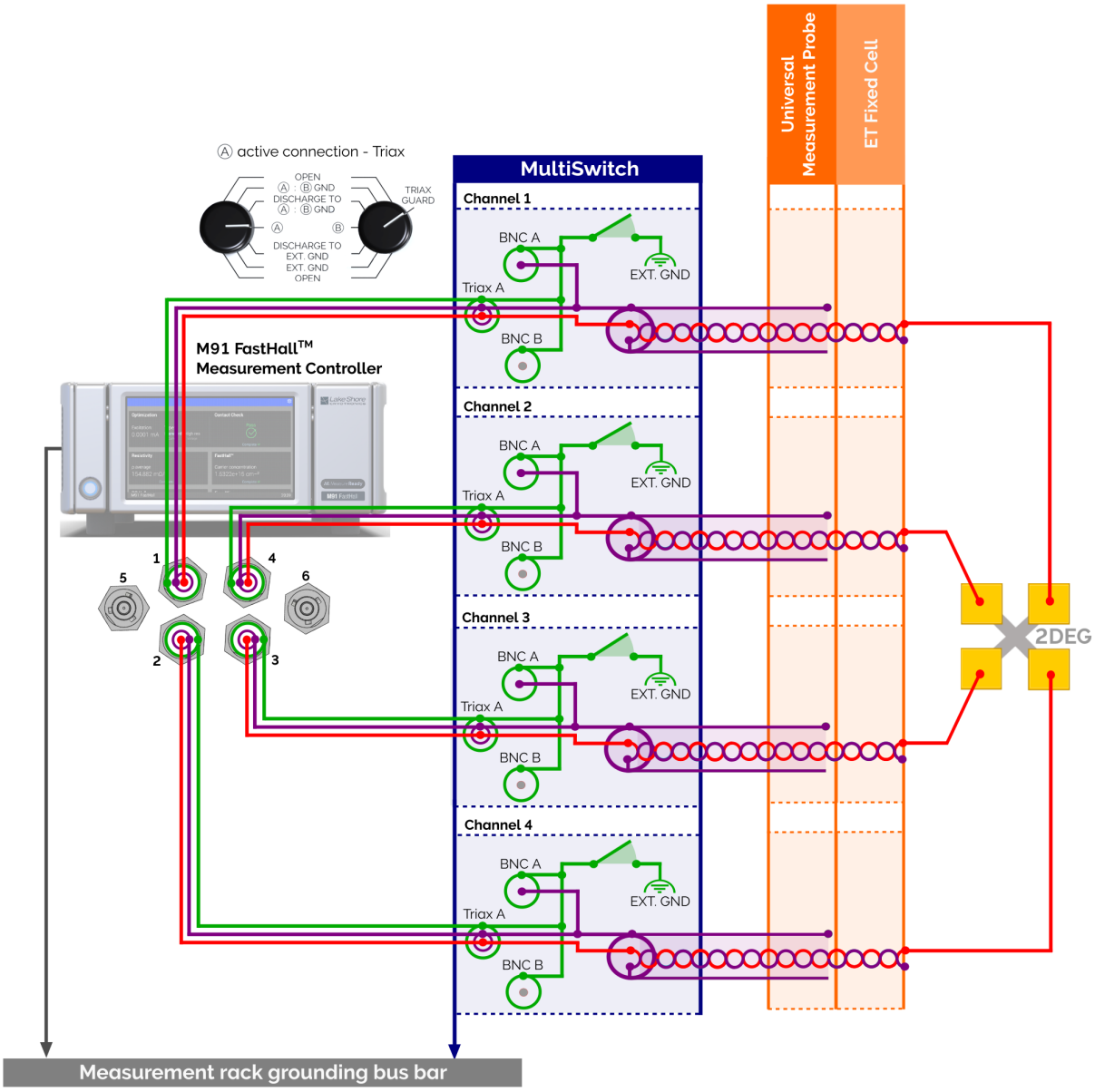

Figure 2: Sample connections to Lake Shore M91 FastHall™ Measurement Controller.

2 Data collection

Experiments typically require the integration and automation of many different components, including the cryostat temperature and magnetic field, and often multiple measurement instruments. The Jupyter environment enables remote control of the cryostat environment and measurement instrumentation, along with automated data collection by utilising the QCoDeS data acquisition framework.

2.1 Cryogenic platform

To control the temperature and magnetic field of the sample environment, the sample was loaded into the Quantum Design Oxford **Teslatron**PT Plus Measurement system. **Teslatron**PT Plus is a cryogen-free variable temperature insert cryostat with a temperature range of 1.5 - 300 K and magnet options up to 14 T. For the measurement described here, a system with a 12 T solenoid magnet was used.

Equipped with our cryostat environmental control software, DECS, **Teslatron**PT Plus provides automated operation and continuous monitoring of system parameters. More information about **Teslatron**PT Plus can be found in the 'About **Teslatron**PT Plus' section at the end of this document.

2.2 Instrumentation

The QDO measurement rack includes the measurement server, breakout boxes and Lake Shore MeasureReady™ instruments. For this measurement the Lake Shore M91 FastHall™ measurement controller, QDO MultiSwitch and Universal Measurement Probe - Driven Guard were used to perform the resistivity measurement.

The detailed wiring setup for the measurement is shown in figure 2. The MultiSwitch breakout box provides triaxial connections to the Lake Shore M91, and together with the Universal Measurement Probe - Driven Guard, routes the measurement signals to the sample. The combination of Universal Measurement Probe - Driven Guard and MultiSwitch allows the guard potential generated by the Lake Shore M91 to be carried down to the probe-cell interface, ensuring a reduction of capacitance effects in the measurement.

2.3 Cryostat and Instrument control

DECS and the measurement instruments are controlled directly from the Jupyter interface using QCoDeS, a Python-based measurement framework that provides standardised drivers for scientific instruments [12]. This open-source Python project already contains many Python drivers for a range measurement instrumentation. Contributing to this project, a QCoDeS driver has been developed to control the temperature and magnetic field of **Teslatron**PT Plus *via* DECS. Along with a driver for the Lake Shore M91 FastHall™ Measurement Controller, this provides complete control of the measurement environment through a single software interface.

An example of connecting to the **Teslatron**PT Plus *via* its QCoDeS driver is shown in figure 3.

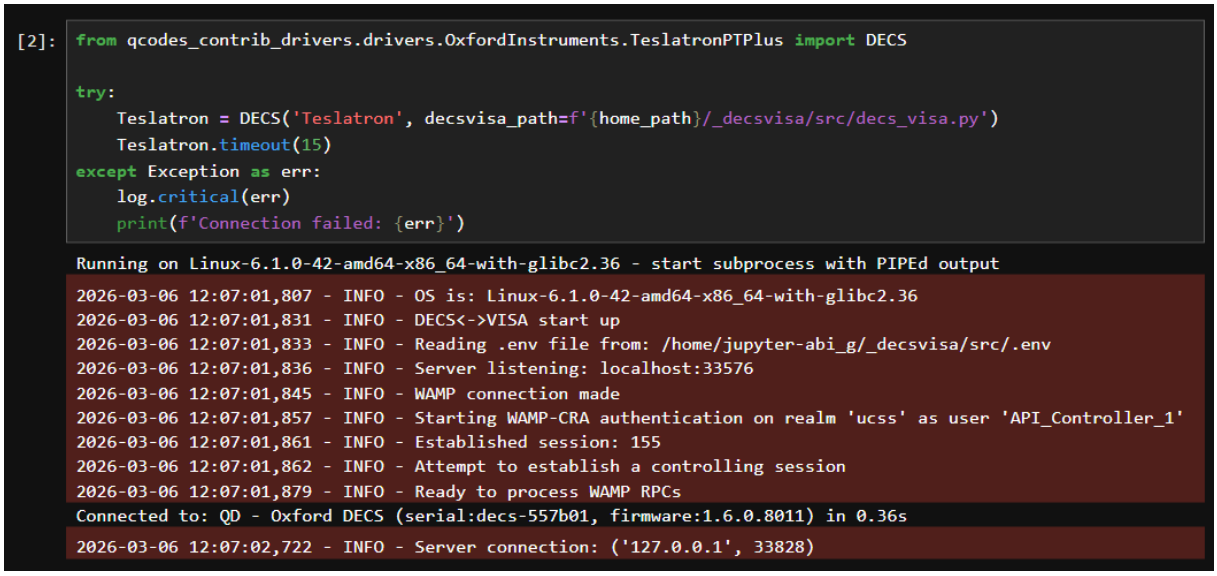

Figure 3: Connecting to the **Teslatron**PT Plus using QCoDeS instrument driver.

The **Teslatron**PT Plus QCoDeS drivers utilises DECS↔VISA [13], a virtual server that allows straight forward integration of DECS based systems with other instruments and common data acquisition methods, which typically conform to Virtual Instrument Software Architecture (VISA) communication standards.

A powerful feature of QCoDeS is the ability to capture the full configuration of the cryostat, or instrument, with the `snapshot()` function. This compiles a dictionary of all parameters including their current values and units. An example of this is shown in figure 4. This snapshot enables future reproducible measurements by preserving the complete instrument state at the time of data acquisition.

```
[4]: Teslatron.print_readable_snapshot(update=True)

Teslatron:
        parameter                value
--------------------------------------------------------------------------------
AV903_state              :      OPEN
AV908_state              :      OPEN
AV911_state              :      OPEN
AV915_state              :      OPEN
High_Flow_Enabled        :      False
IDN                      :      {'vendor': 'QD - Oxford', 'model': 'DECS', 'serial...
Magnet_Current_Vector    :      (0.0, 0.0, 0.0049) (('A', 'A', 'A'))
Magnet_Field_Target      :      (0.0, 0.0, 0.0) (('T', 'T', 'T'))
Magnet_State             :      Holding Persistent
Magnet_Temperature       :      3.1755 (K)
Magnetic_Field_Vector    :      (0.0, 0.0, 0.003046450215779139) (('T', 'T', 'T'))
PT1_Plate_Temperature    :      38.684 (K)
PT2_Plate_Temperature    :      2.9868 (K)
Probe_Heater_Power       :      0.001 (W)
Probe_Target_Temperature :      1 (K)
Probe_Temperature        :      2.528 (K)
Sample_Temperature       :      2.528 (K)
Sample_and_VTI_Linked    :      True
Switch_State             :      CLOSED
System_State             :      Manual
Temperature_Ramp_Rate    :      0 (K/min)
Temperature_Ramp_Target  :      0 (K)
Temperature_Stable       :      True
VTI_Heater_Power         :      0 (W)
VTI_Pressure             :      480 (Pa)
VTI_Target_Pressure      :      550 (Pa)
VTI_Target_Temperature   :      2 (K)
VTI_Temperature          :      2.5311 (K)
timeout                  :      15 (s)
```

Figure 4: Snapshot of **Teslatron**PT Plus parameters.

Similarly, a connection from the Jupyter notebook to the LakeShore M91 FastHall™ Measurement Controller can be made as shown in figure 5.

```
[5]: from qcodes_contrib_drivers.drivers.Lakeshore.M91_FastHall import M91_FastHall

     try:
         M91 = M91_FastHall('M91','TCPIP::192.168.2.24::7777::SOCKET')
         M91.timeout(1)
     except Exception as err:
         log.critical(err)
         print(f'Connection failed: {err}')

     Connected to: Lake Shore M91 (serial:LSA2P79, firmware:2.0.2023121202) in 0.29s
```

Figure 5: Connecting to Lake Shore M91 using QCoDeS instrument driver.

Remote control of DECS and the measurement instruments via QCoDeS drivers in the Jupyter environment allows easy automation of complex temperature and magnetic-field dependent electrical transport measurements, but QCoDeS is more than just a set of instrument drivers.

2.4 Data acquisition

QCoDeS also provides tools for running measurements and storing data in a structured way. Its measurement functions make it easy to define parameter sweeps, automate experiments, and collect results with very little additional code. A collection of template measurement scripts for running common electrical-transport measurements which utilise these QCoDeS measurement functions are provided with the QDO measurement server.

To measure the magnetotransport properties of the 2DEG the magnetic field was swept from 0 to 12 T at 5 different cryogenic temperatures, the basic steps for which are demonstrated in listing 1.

Listing 1 Example code snippet for stepping through different temperature setpoints using QCoDeS instrument drivers and measurement functions.

```
temperature_setpoints = [2, 4, 6, 8, 10]

for temperature in temperature_setpoints:
    # create QCoDeS measurement object
    meas = Measurement(name=f'Field sweep measurement at {temperature} K.')

    # go to target temperature
    Teslatron.ramp_temperature(temperature, 0)
    Teslatron.wait_until_temperature_stable()

    # set magenetic field to zero
    Teslatron.set_magnet_target(0, 0, 0, 0, 'ASAP', 0, False)
    Teslatron.sweep_field()
```

The subsequent measurement was automated using QCoDeS measurement functions and instrument drivers, demonstrated in listing 2.

The relationship between various measurement parameters are also defined for QCoDeS measurements (i.e. y is measured as a function of x) and data are structured in the measurement database to reflect these dependencies.

Extensive metadata are also recorded along with each dataset.

Having data stored consistently in this way greatly simplifies automation of subsequent analysis.

Listing 2 Example code snippet for recording resistivity measurements using QCoDeS instrument drivers and measurement tools

```
# start measurement
with meas.run() as datasaver:

    # start z feild ramp to 12 T at 0.3 T/min
    Teslatron.set_magnet_target(0, 0, 0, 12, 'RATE', 0.3, False)
    Teslatron.sweep_field()

    # read and save a data point every second until field stops ramping
    while Teslatron.Magnet_State() != 'Holding Not Persistent':

            # wait a one second sampling period
            time.sleep(1)

            # perform M91 FastHall measurement
            data = M91.FastHall.start()

            # save data point of measured time, temperature, magnetic field
            # and resistivity to SQL database
            datasaver.add_result(
                (timestamp, time.time()),
                (sample_temperature, Teslatron.Sample_Temperature()),
                (magnetic_field_z, Teslatron.get_z_field()),
                (resistivity, data.ResistivityAverageInOhmMeters()),
                )
# access data from database
dataset = datasaver.dataset
```

3 Results and Analysis

In the previous section examples of collecting data from various instruments within the Jupyter environment provided by the QDO measurement server were presented. The next step would typically be some post-processing of these data.

Historically this could have involved copying *.csv or other text-based data files between systems; needing to re-import these data into different analysis platforms; and manually keeping track of dependencies between these files and various outputs.

Here we show how the data dashboards and powerful analysis, and presentation tools available within the Jupyter environment simplify this workflow.

3.1 Dashboards

During measurements, live data can be displayed as it is acquired using a Grafana dashboard, reading directly from the results database.

Each dataset is given a unique identifier which, along with the database, experiment and sample names, can be used to search and filter results.

Measurement parameters and metadata stored with the data in the database can also be displayed on the dashboards, making them a convenient way to identify datasets for further analysis.

Standard dashboards are provided with the template measurement scripts; these can be easily adapted to create custom dashboards for different experiments. An example dashboard displaying data from the SdH oscillation experiment is shown in figure 6.

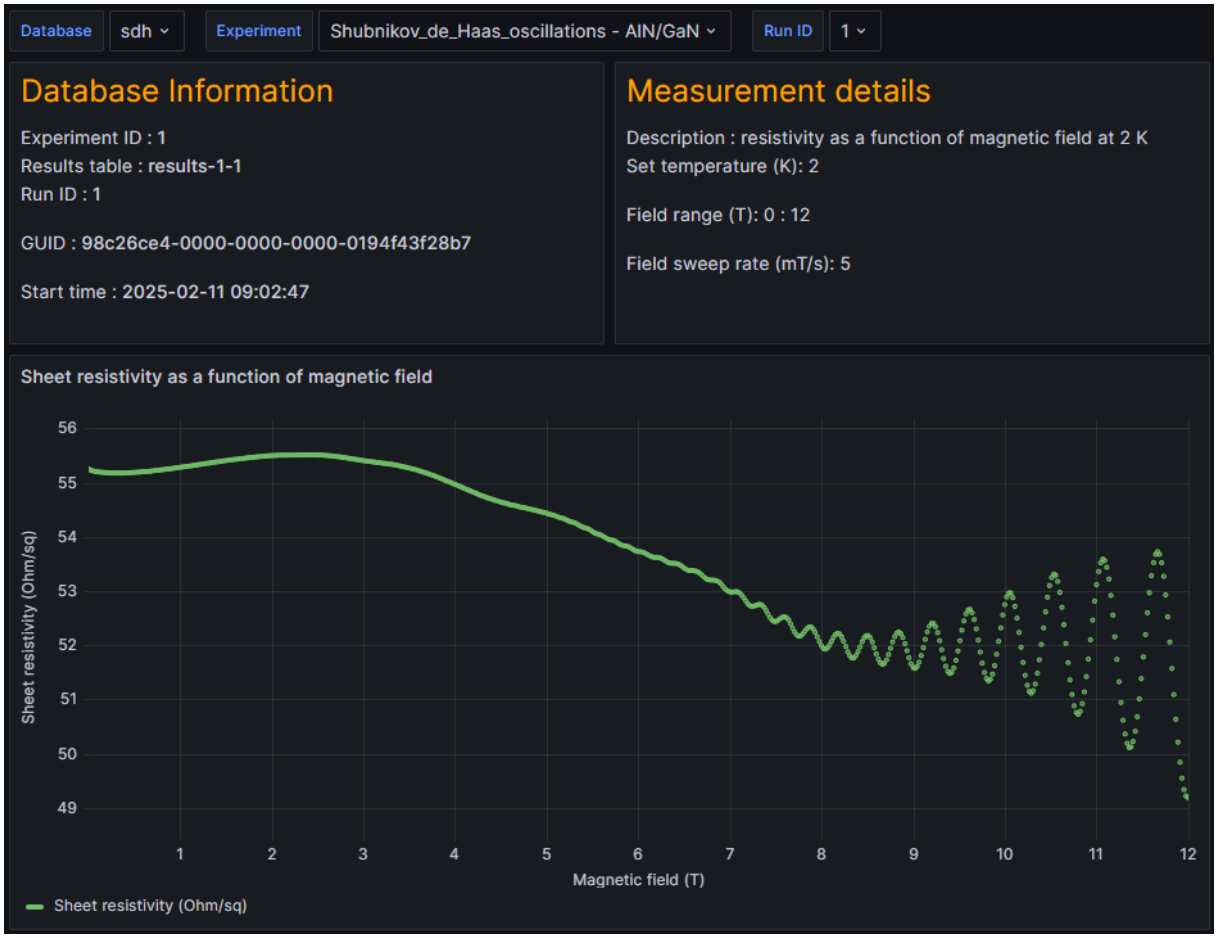

Figure 6: Using Grafana dashboards to identify datasets for further analysis

3.2 Jupyter notebooks

Data can be loaded from the SQL database using QCoDeS functions, providing easy access to the extensive Python libraries for data analysis and plotting available within the Jupyter environment. Since each of the measurement runs from an experiment are stored in the same format, with the same parameters and metadata information, and differ only in their run IDs, the same code can be run for multiple datasets without modification.

Listing 3 Example code loading datasets identified by their run IDs into Python list for subsequent analysis.

```
db_path = "../data/sdh.db"
experiment_name = "Shubnikov_de_Haas_oscillations"
sample_name = "AlN/GaN"

initialise_or_create_database_at(db_path)
load_or_create_experiment(
    experiment_name=experiment_name,
    sample_name=sample_name
    )

# Selected datasets
data_run_ids = [1,2,3,4,5]

raw_data_sets = []

for data_run_id in data_run_ids:
    raw_data_sets.append(
        load_by_run_spec(captured_run_id=data_run_id)
        .get_parameter_data()["Sheet_Resistivity"]
        )

# subsequent analysis
for data_set in raw_data_sets:
    ...
```

This ease of data handling greatly simplifies and speeds up analysis by allowing automatic processing of multiple datasets. Figures can be produced in (and exported from) the same notebook as the data analysis and inserted into reports as described in section 4.3.1 and section 4.1.5.

3.3 Resistance oscillations

Sheet resistivity measurements of the 2DEG were taken at temperatures between 2 K and 10 K. All five datasets are loaded into a Jupyter notebook, where they are analysed and plotted in together. A plot of the sheet resistivity, R_S, as a function of $B_\perp$, up to 12 T is shown in figure 7, SdH oscillations appear at higher fields, having a greater amplitude at lower temperatures.

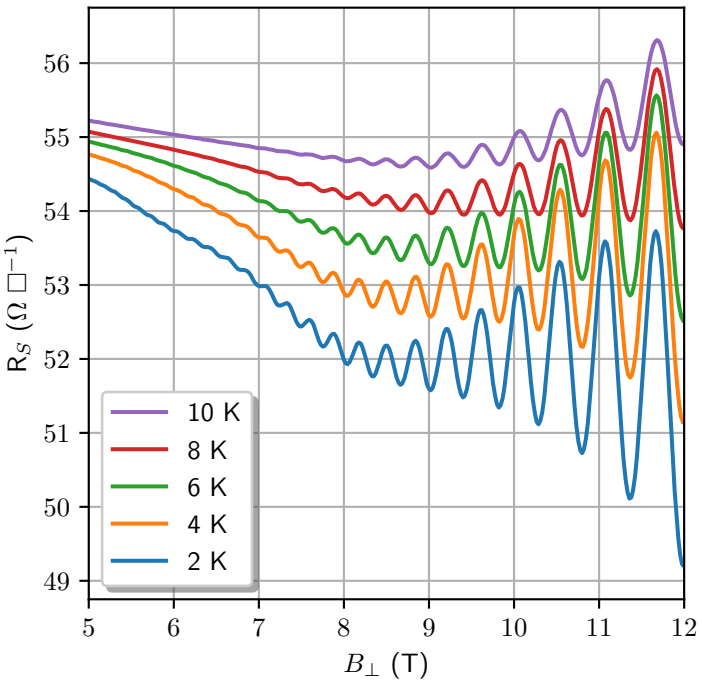

Figure 7: Sheet resistivity, R_S, as a function of magnetic field, $B_\perp$, for the 2DEG at different temperatures showing the development of SdH oscillations at higher fields and lower temperatures.

3.3.1 Background subtraction

In order to extract information from these oscillations and to quantitatively compare the datasets for different temperatures, the slowly varying background changes in the resistivity should be subtracted. The SdH oscillations are periodic in $1/B_\perp$ with a period related to the carrier density, n_e, in the 2DEG. Assuming a value for n_e allows this background to be determined by averaging the sheet resistivity over each oscillation and fitting a polynomial (in this case of 5^{th}-order) to these points.

An example of this averaging and resulting polynomial fit is plotted along with the raw data in figure 8.

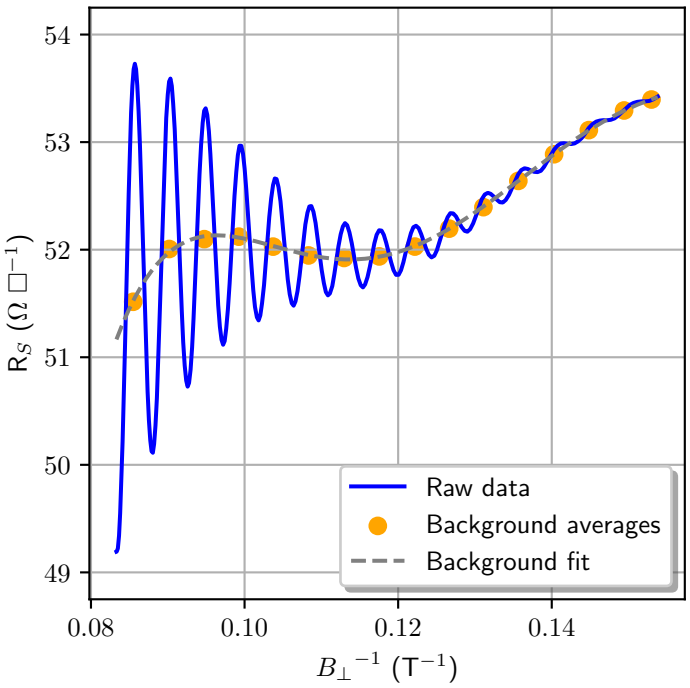

Figure 8: Sheet resistivity, R_S, as a function of inverse magnetic field, $B_\perp{}^{-1}$, at 2 K, with the background level fitted. The orange points show the sheet resistivity and inverse field averages for each oscillation and the grey dashed line shows the 5^{th}-order polynomial fit to these values.

This analysis can be easily repeated for each of the five datasets by running the same code for each measurement run ID, as described in listing 3 - the resulting background fits are shown in figure 9.

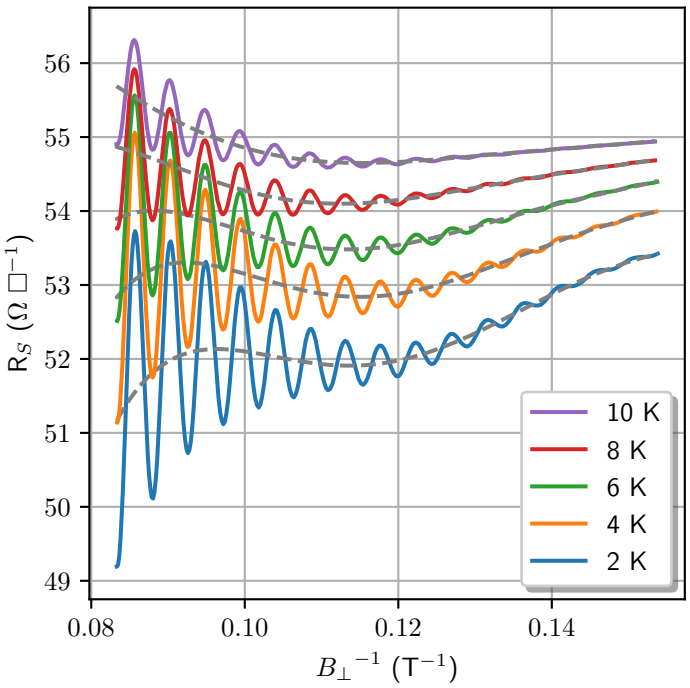

Figure 9: Sheet resistivity, R_S, as a function of inverse magnetic field, $B_\perp{}^{-1}$, with fitted polynomial background (grey dashed lines) shown at different temperatures.

3.3.2 Parent datasets

QCoDeS supports the concept of 'parent' datasets (that contain the raw data) and 'child' datasets that contain, for example, the results of fitting procedures. These child datasets can be linked to parent datasets and stored in the results database. This allows full traceability of results as it is always clear exactly which data were used in any analysis steps.

An example of code used to fit a function to experimental data, save the results to the database and link the two datasets is given in listing 4.

Listing 4 Example code that fits a function to raw data and links the fit to this original dataset.

```
# Create a fitting 'experiment'
meas = Measurement(name=f"Shubnikov-de Haas oscillations - background fit")

# register parameters for fit
meas.register_custom_parameter(
    'fit_axis',
    label='Fit axis'
    )
meas.register_custom_parameter(
    'fit_results',
    label='Fitted curve',
    setpoints=['fit_axis'])

#link measurement data to fitting results
meas.register_parent(
    parent=data_set,
    link_type="curve fit"
    )

#perform the fit and save the fitting results
with meas.run() as datasaver:
    fit = Polynomial.fit(
        data_set["xdata"],
        data_set["ydata"],
        deg=5
        )

    background_fit = fit(data_set["xdata"])

    datasaver.add_result(
        ('fit_axis', data_set["xdata"]),
        ('fit_curve', background_fit)
        )
```

The fit linked to the raw data can be displayed along with the raw data themselves on a Grafana dashboard (as in the example in figure 10). Since the datasets are linked, the run IDs for the analysis can be filtered to only those relevant for a given parent. Multiple fit attempts can be linked to a single experimental dataset and Grafana allows these to be viewed easily for comparison to enable review of various fitting procedures.

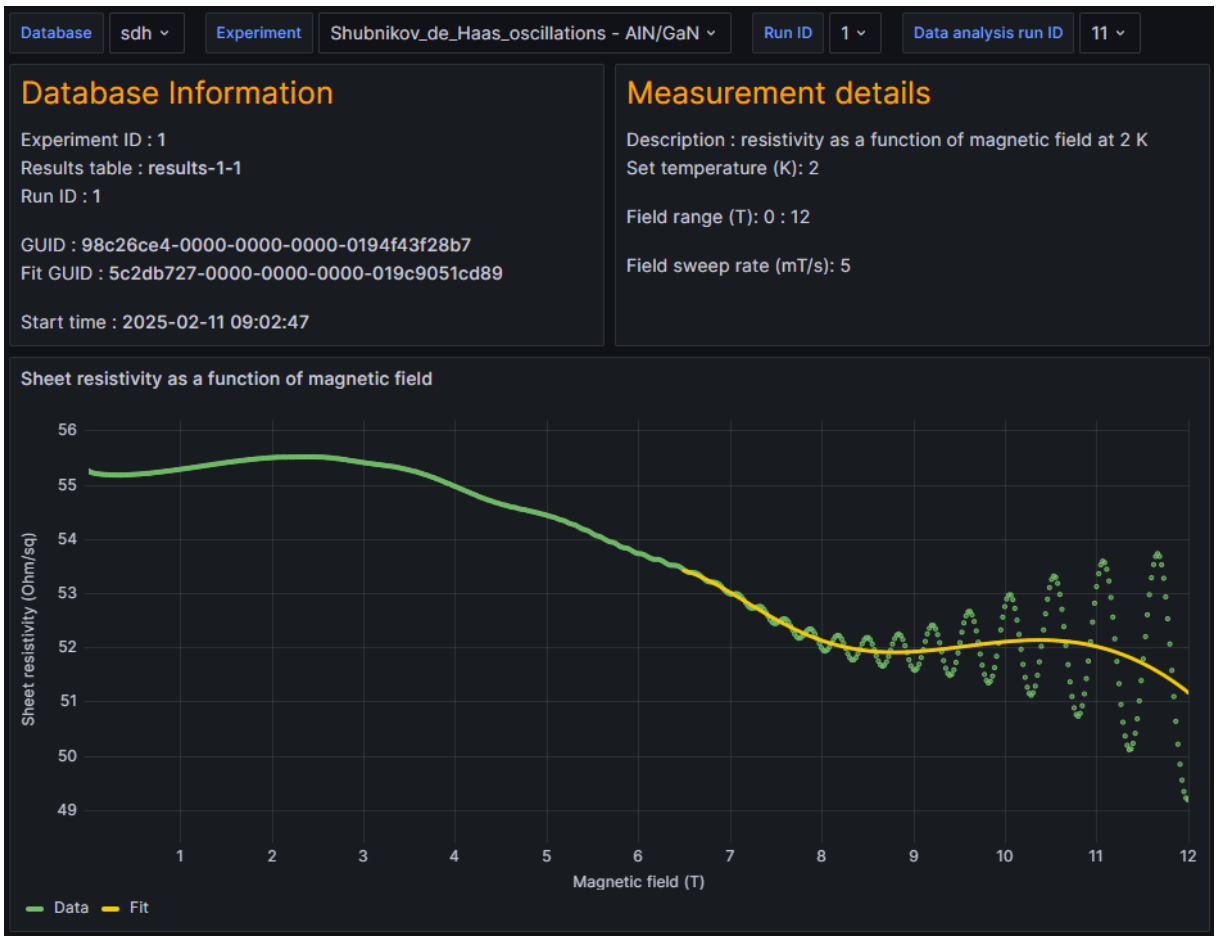

Figure 10: Linking fits to datasets. The 'parent' raw data (green points) and 'child' background fit (yellow curve) can both be displayed on a Grafana dashboard. By first selecting the measurement data run ID, the analysis run IDs can be filtered to just those linked to this parent dataset.

Once the background fit for the SdH oscillations has been optimised, subtracting these polynomials from each dataset leaves only the SdH oscillations in the sheet resistivity (see figure 11), whose amplitudes can now be analysed as a function of inverse magnetic field and temperature.

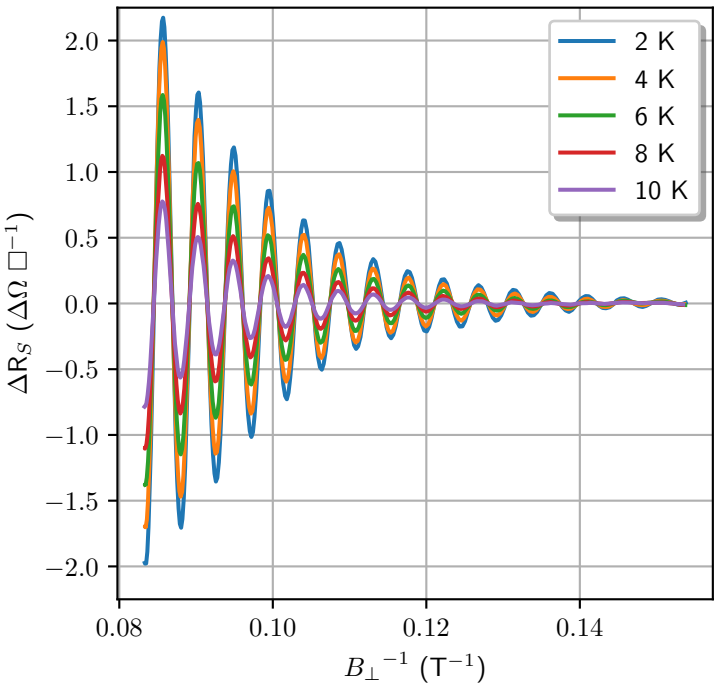

Figure 11: Sheet resistivity oscillations, ΔR_S, as a function of inverse magnetic field, ${B_\perp}^{-1}$, at different temperatures (after background subtraction from the raw datasets).

3.4 Frequency analysis

The SdH oscillations at different temperatures visible in figure 11 can be compared directly. The amplitude of the oscillation increases with decreasing temperature, this is reflected in the Discrete Fourier Transforms (DFTs) of the background-subtracted data (figure 12). The frequency at the peak of the DFT, B_F, can be used to calculate the carrier density $n_e = 2eB_F/h$, where h is Planck's constant and e is the electron charge.

The fact there appears to be a single peak and its position is independent of temperature implies there is a single sub-band and the carrier density is constant for these measurements.

For this heterostructure, a peak frequency of 218 T gives a value of $n_e = 1.056 \times 10^{13}$ cm^{-2}, consistent with the value assumed for the analysis in section 3.3.1.

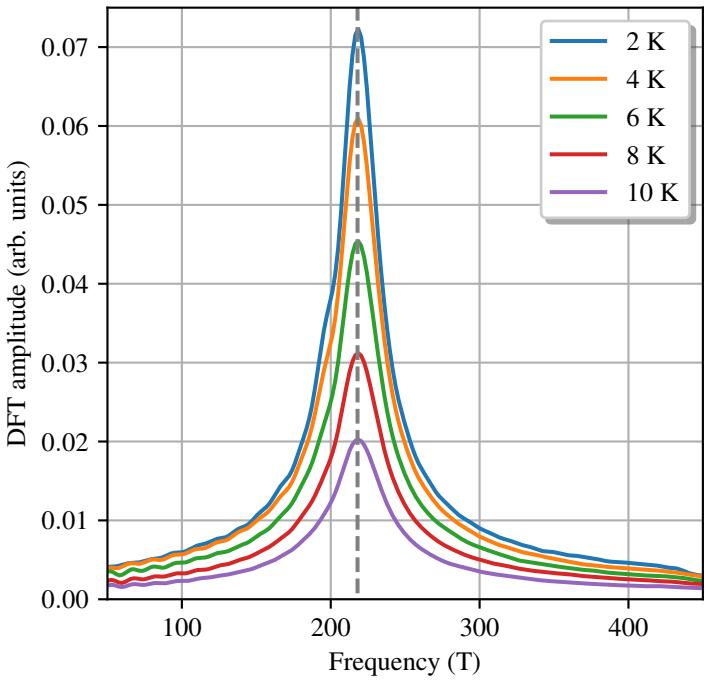

Figure 12: Discrete Fourier transform of the sheet resistivity oscillations ΔR_S vs ${B_\perp}^{-1}$ data, at different temperatures (after background subtraction from raw data). The dashed vertical line shows the value of B_F used to infer the carrier concentration, n_e.

3.5 Temperature and field dependence

There are competing energy scales present in the 2DEG system. Degenerate Landau levels are separated by the cyclotron energy $\hbar\omega_c$, where $\hbar$ is the reduced Planck constant. This energy gap increases with applied field as $\omega_c = eB/m^*$. Against this is the inherent thermal broadening of states $\propto k_B T$, and so one could anticipate a factor of $\sim B/T$ to be important when discussing such effects.

Traditionally, to ease analysis, the temperature and field dependence of the SdH oscillations have been analysed separately [14].

3.5.1 Amplitude ratios and effective mass

As the SdH oscillations maxima or minima occur at constant fields (figure 11) the amplitude of these peaks can be analysed as a function of temperature. The individual peaks can be extracted using a method to find zero crossings similar to root finding by bisection [15]. Such an analysis is shown in figure 13.

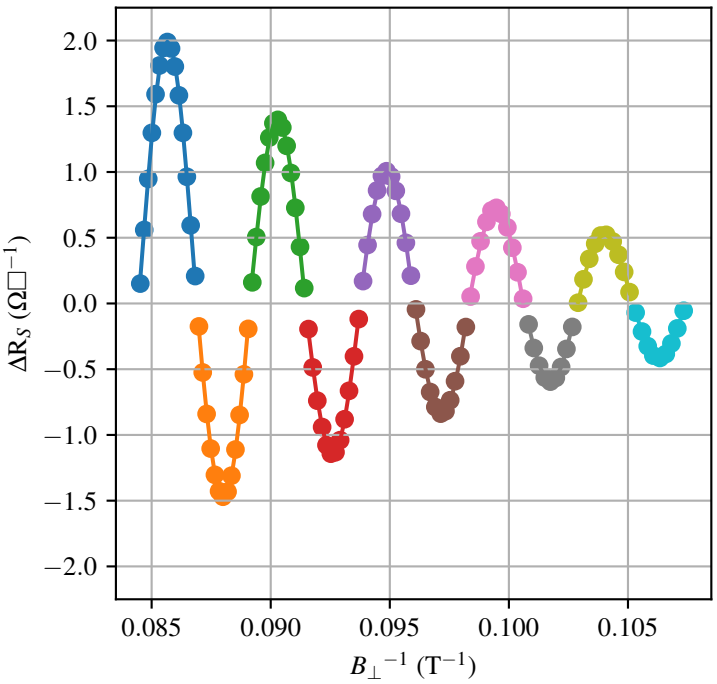

Figure 13: Peak extraction from the background-subtracted 2 K SdH oscillation dataset.

This analysis can be repeated for datasets recorded at different temperatures and the temperature dependence of each peak determined. The resulting data can be normalised by dividing the peak heights by data taken at a reference temperature (usually the lowest accessible temperature): the field-dependent envelope function for the oscillation amplitude then cancels, leaving only the anticipated thermal damping factor [14] [16] [17]

$$\frac{\Delta R_S(T)}{\Delta R_S(T_0)} = \frac{T \sinh \chi(T_0)}{T_0 \sinh \chi(T)} \tag{1}$$

where

$$\chi = \frac{2\pi^2 k_B T m^*}{\hbar e B_\perp}. \tag{2}$$

Such an analysis is shown for several Landau level filling factors in figure 14, along with fits to equation 1.

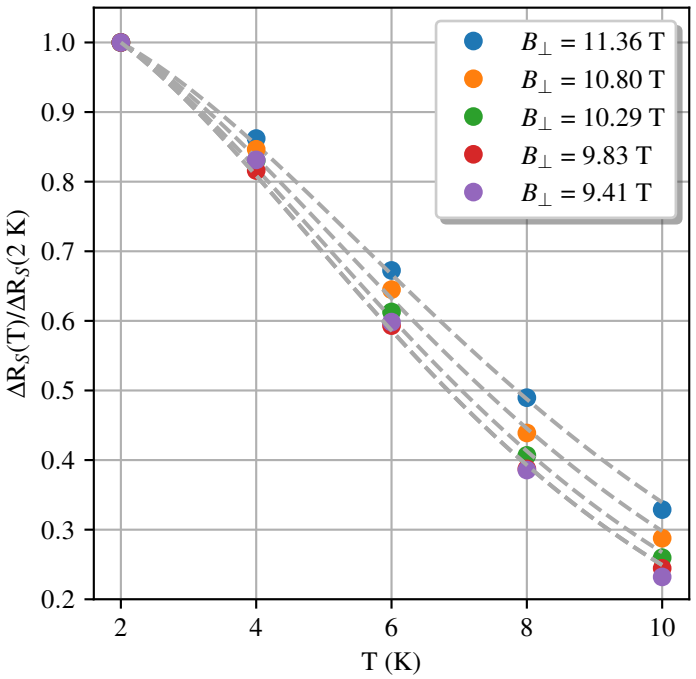

Figure 14: Amplitude ratios of the temperature dependence of the SdH oscillations and associated fits.

From equation 2 it is clear that (since T and $B_\perp$ are know) these fits allow for the determination of the electron effective mass.

3.5.2 Peak heights and scattering times

The peak extraction routine shown in figure 13 also allows the SdH oscillation amplitude to be plotted as a function of (inverse) magnetic field.

The amplitude of the SdH oscillations is anticipated to be exponentially suppressed by the envelope function

$$\exp\left(-\frac{\pi m^*}{e\tau_q}\frac{1}{B_\perp}\right). \tag{3}$$

This exponential suppression is shown, in a so-called Dingle plot, in figure 15.

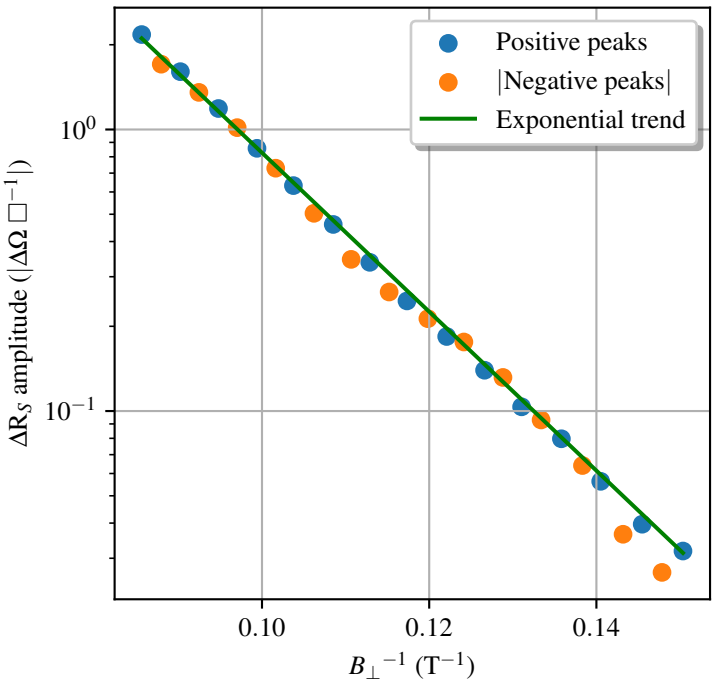

Figure 15: Logarithmic plot of the amplitudes of the peaks in sheet resistivity, as a function of inverse magnetic field, $B_\perp{}^{-1}$, at 2 K. The positive peak values are plotted in blue, while the absolute values of the negative peaks are plotted in orange. The green line shows an exponential trend fitted through the peak height data.

Given the effective mass is now known, the gradient of this (appropriately scaled) fit allows the quantum lifetime, τ_q, to be determined [14].

3.5.3 Lifshitz-Kosevich fitting

The Lifshitz-Kosevich formula describes the damping of the SdH oscillations as a function of inverse magnetic field due to thermal and impurity effects. The form of the oscillations is given by equation 4 [18] [19]:

$$\Delta R_S = 4R_0 \frac{\chi}{\sinh \chi} \exp \frac{-\pi}{\omega_C \tau_q} \cos \left[2\pi \left(\frac{F}{B_\perp} + \phi \right) \right], \tag{4}$$

where F and ϕ are the frequency and phase of the oscillations respectively.

As the Jupyter environment provides access to sophisticated regression tools, equation 4 can be fitted directly to the background subtracted data for the SdH oscillations at different temperatures, as in figure 16.

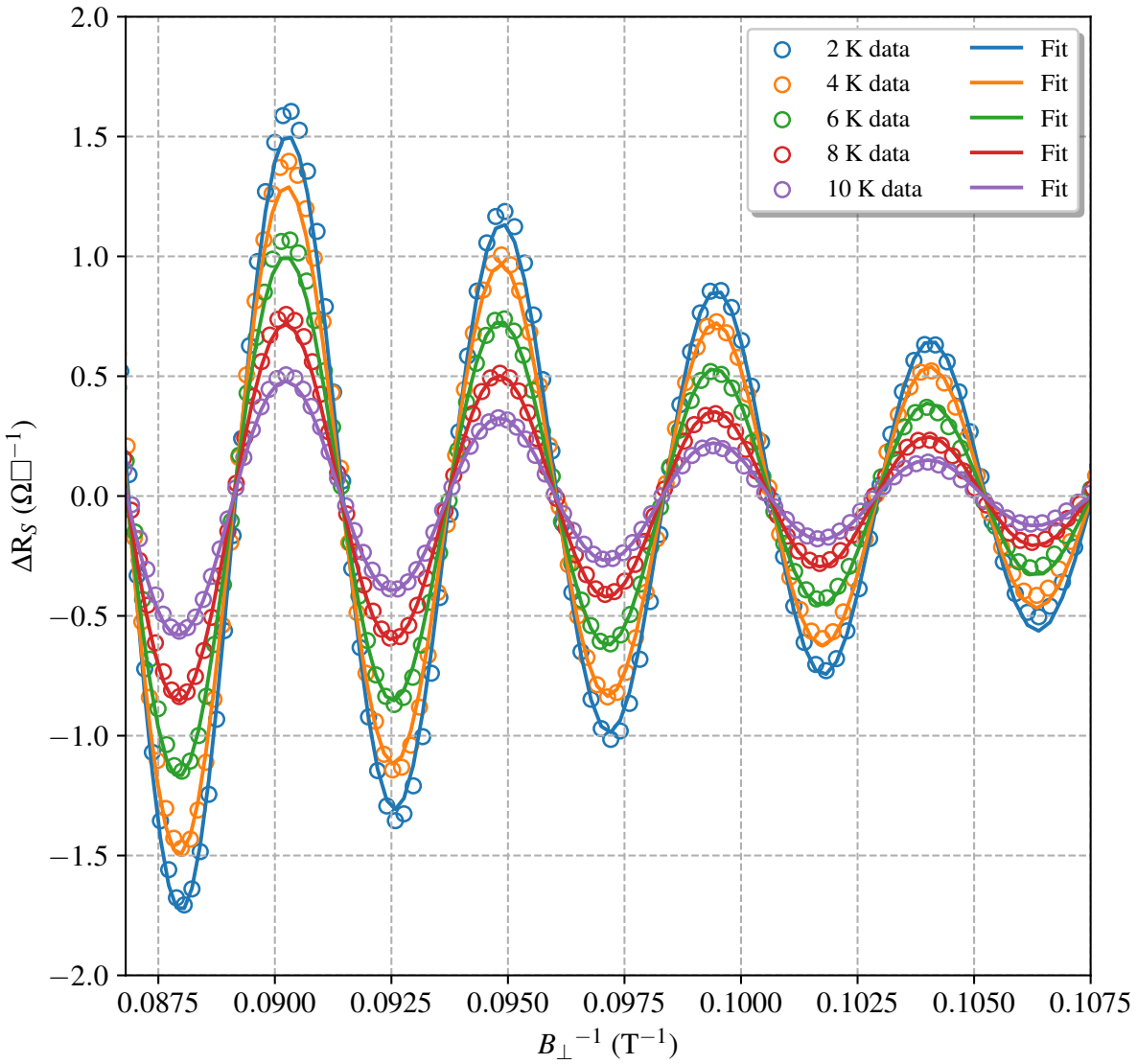

Figure 16: Change in sheet resistivity, ΔR_S, as a function of inverse magnetic field, ${B_\perp}^{-1}$, with the resistance oscillations fitted using equation 4, at temperatures between 2 K and 10 K.

This approach allows all the parameters of interest to be extracted in a single analysis step.

We determine a value of $m^* = 0.22m_e$ for the effective mass and a quantum lifetime $\tau_q = 0.06$ ps, both consistent with expectations for this 2DEG system [16].

Further, as all the analysis and resulting plots presented here can be produced from a single Jupyter notebook. If the experiment is repeated, or other temperature values should be included, only the run IDs selected when loading the data need be modified: running the script will perform the analysis and plot the data without any further user input being required.

4 Document preparation

Collecting and analysing data from an experiment is far from the end of the story. The results generated also need to be disseminated to have impact.

The Jupyter environment can automate the collection and analysis of data, as demonstrated in previous sections, here we show that it can also be used for document preparation.

4.1 Markdown and Pandoc

Often LaTeX [20] is the preferred tool for generating technical documents, but there is something of a learning curve for using or setting this up. The QDO measurement server has a full `tex-live` installation which you can use directly if you wish, however for many use cases Markdown [21] has become the *defacto* standard for preparing documents.

4.1.1 Markdown files and previews

Jupyter has native support for creating and viewing Markdown, as shown in figure 17.

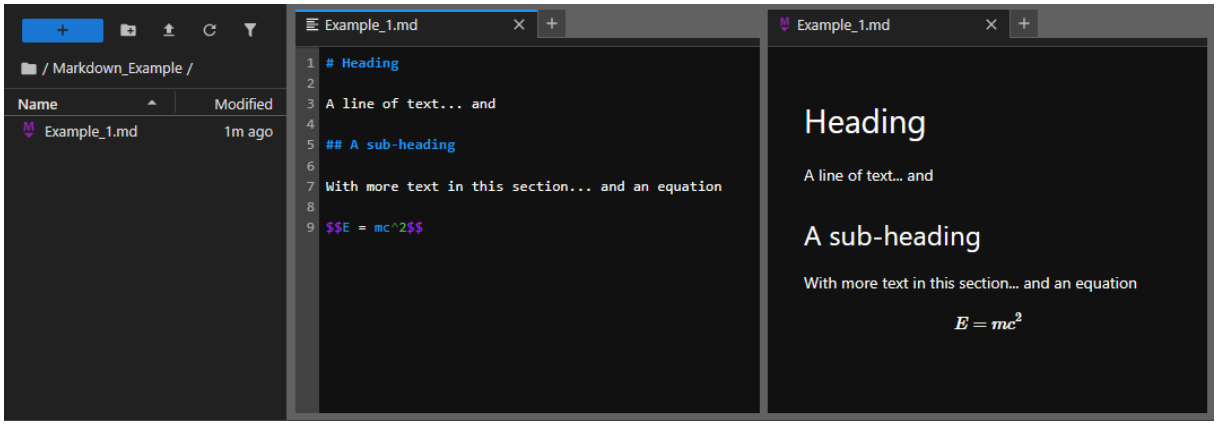

Figure 17: A markdown document and associated preview visible in the Jupyter environment.

The simple Markdown syntax is described in [21], and there are also some useful notes in the following sections.

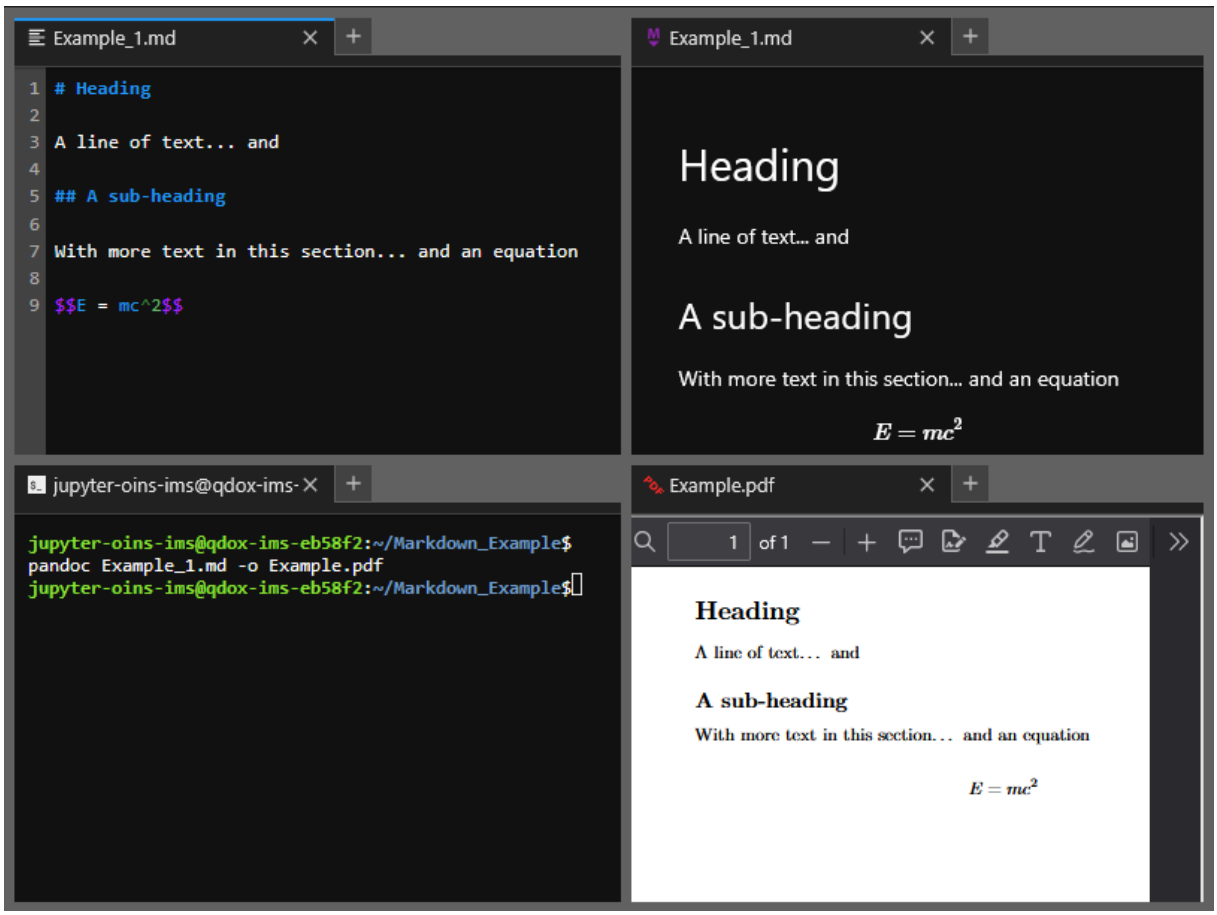

Figure 18: Generating a PDF document from Markdown input from within the Jupyter environment.

4.1.2 Pandoc document conversion

The QDO measurement server also comes pre-installed with Pandoc [22]. Pandoc can transform documents from one format to another. For example, it is able to take Markdown input and generate a Portable Document Format (PDF) output, as shown in figure 18. Formatting of this output can be controlled by templates, more details on this will be provided in section 4.3.

4.1.3 Internal document references

Often one would like to reference one section (or figure, or table, or equation) within a document from elsewhere. The QDO measurement server come pre-installed with pandoc-crossref [23], a Pandoc filter for cross-referencing.

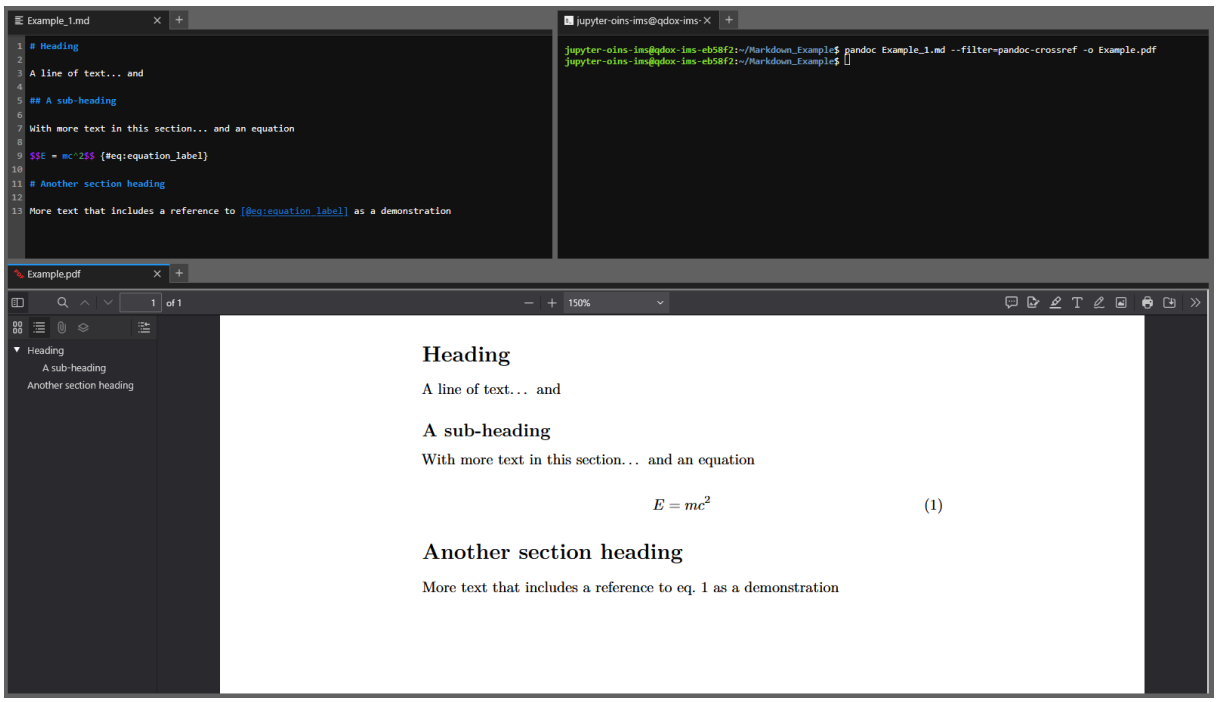

Figure 19: Generating a PDF document with internal references to an equation from within Jupyter environment.

In the Markdown text, labels are applied to elements which can then be referenced in other parts of the document: the syntax for this labelling/referencing is described in detail in [23]. The document is then built with the application of the pandoc-crossref filter, figure 19.

4.1.4 Equations

Writing equations in Markdown is the same as in LaTeX. A good overview in given in [24].

As is shown in figure 19 the equation is tagged `{#eq:equation_label}` and then this tag is referenced from elsewhere in the document using `[@eq:equation_label]`.

4.1.5 Figures

Figures (images) are inserted with the standard Markdown syntax, and the reference tag appended, e.g.

```
![Figure caption.](figure_file.name){#fig:figure_label}
```

The figure can then be referenced elsewhere in the document with `[@fig:figure_label]`.

4.1.6 Tables

Tables are inserted with the standard Markdown syntax, and the reference tag appended, e.g. this Markdown

```
| right aligned | centred | left aligned |
|----:|:---:|:----|
| Abc | 123 | xyz |
: An example table... {#tbl:example_table}
```

would generate table 1, which could be referenced with `[@tbl:example_table]`.

right aligned	centred	left aligned
Abc	123	xyz

Table 1: An example table, including its cross-reference details, generated from the Markdown text shown above.

4.1.7 Document sections

To link to a section of a document, a reference tag is applied to the end of the section title, e.g.

```
## A sub-title {#sec:section_label}
```

This reference can then be used from other sections of the document using `[@sec:section_label]`.

4.2 Including references to literature

In scientific writing in particular, referencing the literature is critical. There are many tools available for handling external references, often the citation information is compiled into a bibliography file. The Pandoc `citeproc` filter can be used to process citations based on this information. Here we provide an example of using BibTeX [25].

The first step is to compile the citation information you require. Most journals provide this *via* their webpages, an example is given in figures 20, 21.

Figure 20: Exporting a reference from the webpages of a journal.

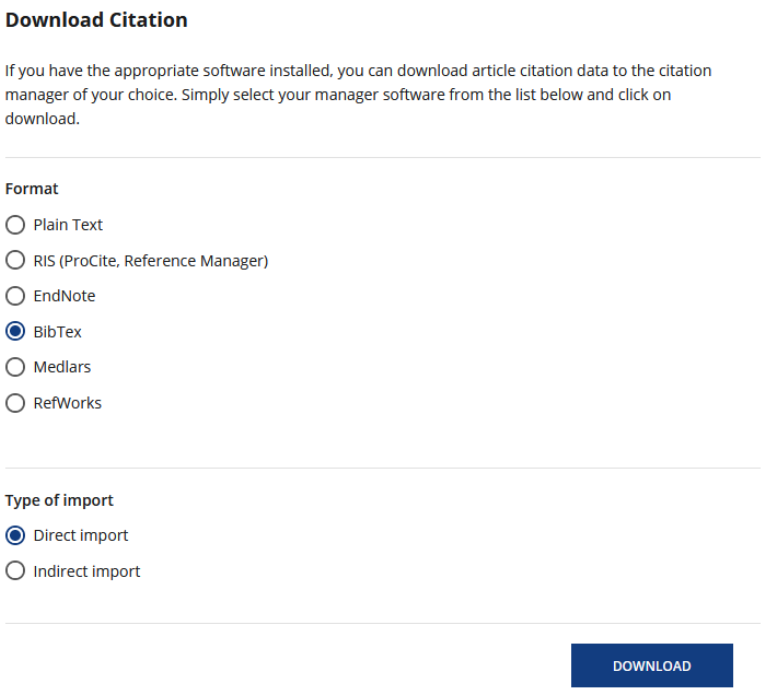

Figure 21: Selecting BibTex as the citation format.

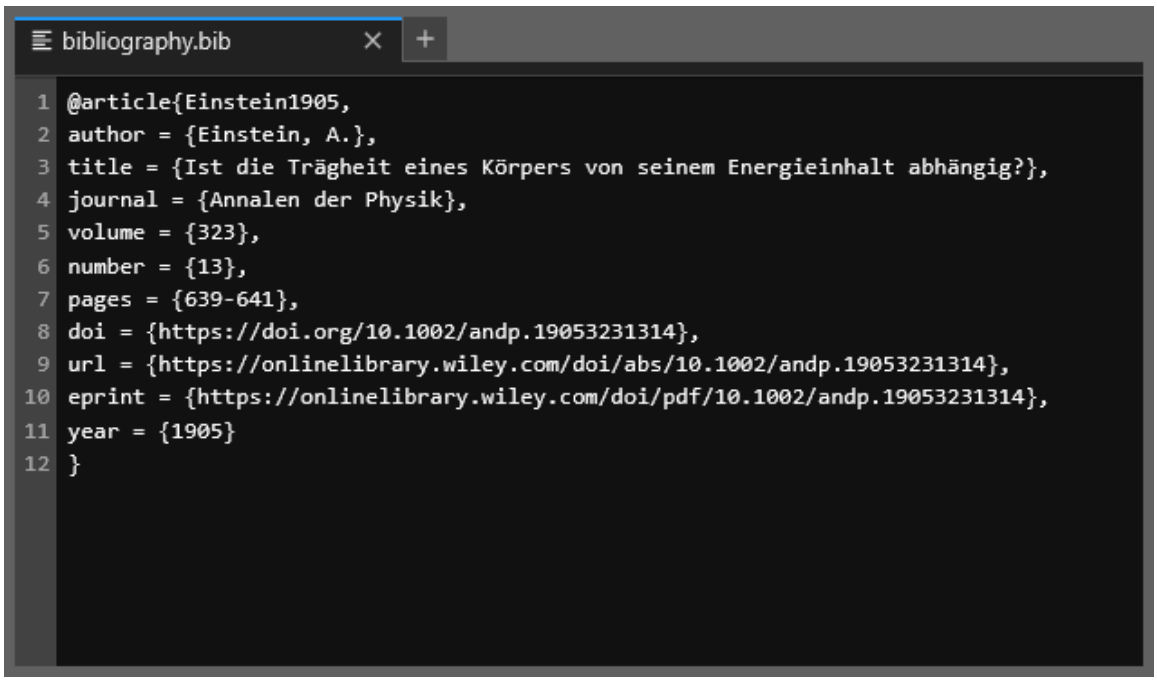

Figure 22: The contents of the bibliography.bib file.

The content of citation information is shown in the bibliography file in figure 22.

The cross-reference in the document then automatically generates the citation information from the bibliography file.

NB the `--citeproc` filter should be applied *after* `pandoc-crossref` on the command line, figure 23.

It can also be seen in figure 23 that the command line instructions to build the output can become quite lengthy. The is addressed in sections 4.2.1, 4.3.

4.2.1 Reference formatting

The way references are formatted in your document can be controlled. Pandoc understands the Citation Style Language (CSL) [26]. This allows different citation styles to be applied by passing in an appropriate CSL schema, figure 24.

Also shown in figure 24 is the use of a small “bash script” to build the output rather than needing to re-enter lengthy command line options.

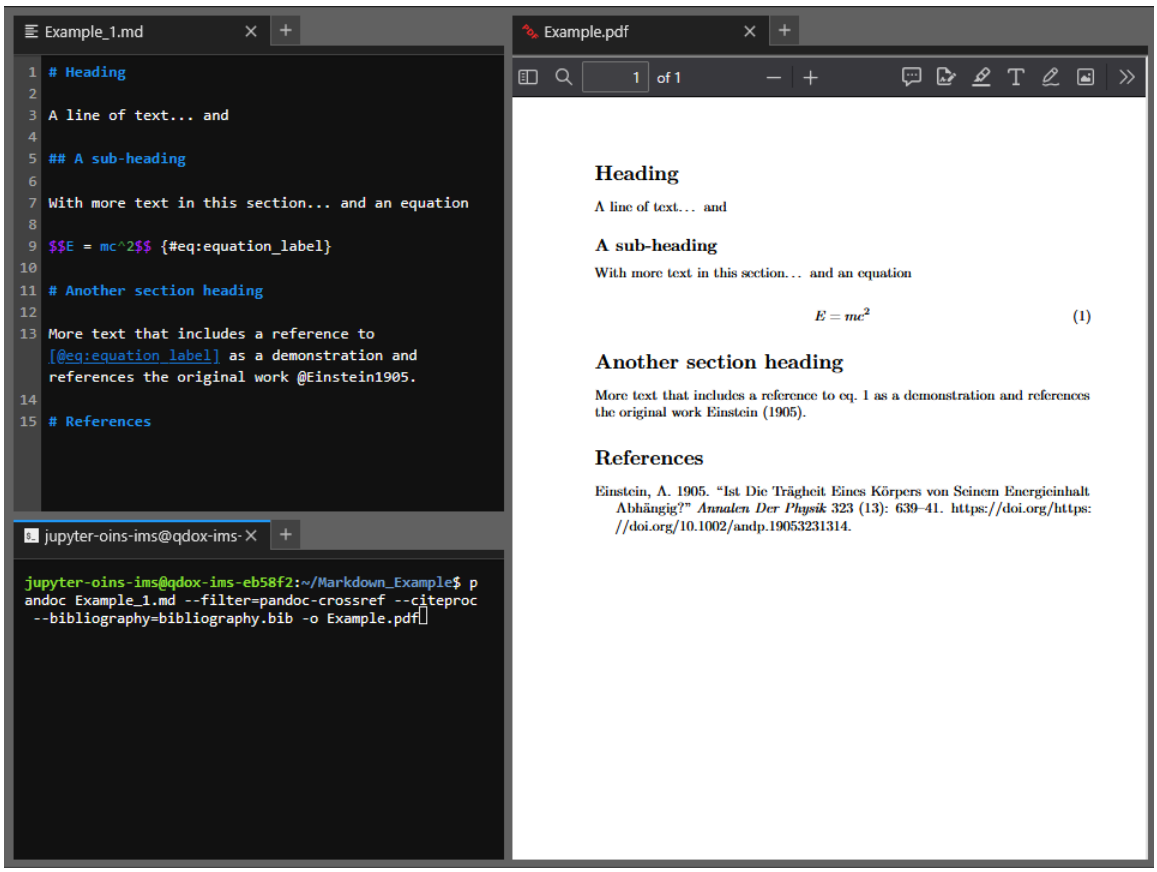

Figure 23: PDF file generated with an example reference.

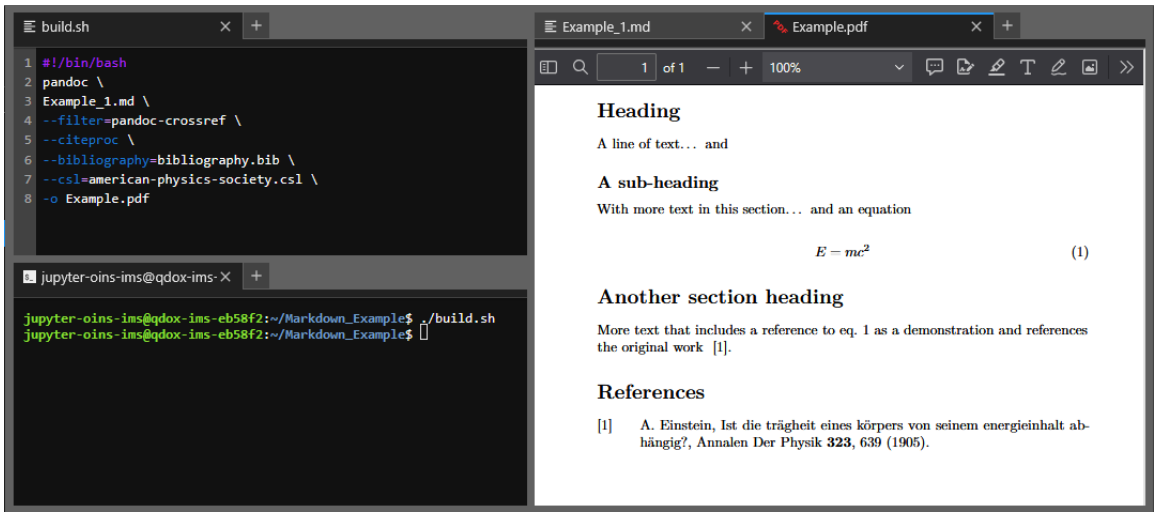

Figure 24: PDF file generated from a build script file with a particular reference formatting style.

4.2.2 bibutils

It may be the case that citation information is only available in a format other than BibTex (in the example below the RIS format, from Research Information Systems). The QDO measurement server has the `bibutils` [27] tool-set pre-installed.

These tools allow conversion between file formats, in listing 5 several commands are linked *via* pipelines:

- `ris2xml` first converts the `ris` format into an intermediate `xml` representation
- the output above is piped into `xml2bib` that converts the `xml` to `bib` format
- the output above is piped into `tee`, which appends the text to the bibliography file.

Listing 5 Command line tools pipelined to transform bibliography file formats.

```
ris2xml vK2020.ris | xml2bib | tee -a bibliography.bib
```

4.3 Generating output

As was discussed in section 4.2 the command line for building output can become long; in addition to this information, Pandoc can also process meta-data associated with the document. This metadata can be supplied, as YAML [28], in three ways:

- On the command line
- Embedded directly into the document in a YAML header
- Using a Pandoc “defaults” file

The command line option quickly becomes impractical. In figure 25 a metadata block can be seen embedded as the start of the document. In addition to information about the document (such as the author) options relating to the presentation of the document (such as the two-column layout) are also included.

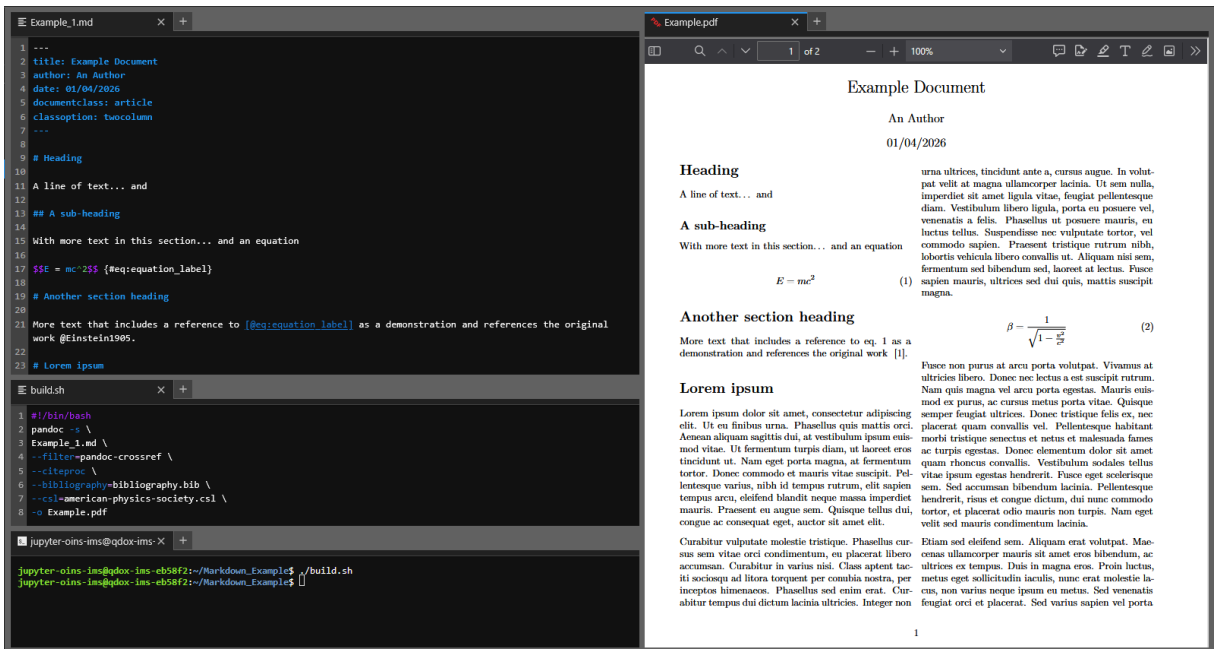

Figure 25: An example of a PDF document formatted as a two-column article.

The embedded YAML is usually fine for short documents with a limited number of options. For more complex applications the Pandoc defaults file [22] will be the correct approach.

4.3.1 Exporting plots from notebooks

A key advantage of working in the Jupyter environment is that the data acquisition, data analysis and presentation of results are all controlled through a single interface. The plots resulting from the analysis steps presented in section 3 can be easily exported from the Jupyter notebook to be included in the document. The code snippet in listing 6 shows a plot being generated and then exported (as a PDF in this case) to a location on the measurement-server file system. This image file can then be imported into the document as described in section 4.1.5.

The key advantage here is that should the data analysis in the notebook be updated, or the plot modified in some way, once the document is rebuilt any changes to the generated images are automatically incorporated into the final output.

Listing 6 Example code that produces and exports a plot to a PDF file.

```
plt.rc('text', usetex=True)
plt.rc('text.latex', preamble=r'\usepackage{amssymb}')

fig, ax = plt.subplots(1, 1, figsize=(6,6))

...

ax.legend(shadow=True)
ax.grid()
ax.set_xlabel(r'${B_{\perp}}^{-1}$ (T$^{-1}$)')
ax.set_ylabel(r'R$_S$ ($\Omega \square^{-1}$)')

plt.savefig('../img/raw_data_with_background_fits.pdf',
            bbox_inches='tight',dpi=300)
```

4.3.2 Templates and formatting

When generating standalone documents (command line `-s` option), Pandoc applies document templates. Customised templates can be used should you wish to format your document in a specific way, or for a specific journal. Details of customising templates can be found in the Pandoc documentation [22].

> All of the data that were acquired and presented in this document; all of the associated analysis of those data; and all of the typesetting and formatting of this document has been performed within the Jupyter environment provides with the QDO measurement server, using exactly the processes described above.

References

[1] K. von Klitzing et al., 40 years of the quantum Hall effect, Nature Reviews Physics **2**, 397 (2020).

[2] P. Liu, J. R. Williams, and J. J. Cha, Topological nanomaterials, Nature Reviews Materials **4**, 479 (2019).

[3] A. Berdyugin, Transport Properties of Novel van der Waals Materials, PhD thesis, The University of Manchester, 2020.

[4] K. von Klitzing, G. Dorda, and M. Pepper, New method for high-accuracy determination of the fine-structure constant based on quantized Hall resistance, Phys. Rev. Lett. **45**, 494 (1980).

[5] D. C. Tsui, H. L. Stormer, and A. C. Gossard, Two-dimensional magnetotransport in the extreme quantum limit, Phys. Rev. Lett. **48**, 1559 (1982).

[6] J. H. Davies, *The Physics of Low-Dimensional Semiconductors* (Cambridge University Press, 1998).

[7] T. Ando, A. B. Fowler, and F. Stern, Electronic properties of two-dimensional systems, Rev. Mod. Phys. **54**, 437 (1982).

[8] I. Lifshitz and A. M. Kosevich, Theory of magnetic susceptibility in metals at low temperatures, Sov. Phys. JETP **2**, 636 (1956).

[9] N. Suresh Kumar, K. C. B. Naidu, P. Banerjee, T. Anil Babu, and B. Venkata Shiva Reddy, A review on metamaterials for device applications, Crystals **11**, (2021).

[10] L. J. van der Pauw, A method of measuring the specific resistivity and Hall effect of discs of arbitrary shape, Philips Research Reports **13**, 1 (1958).

[11] J. David and M. Buehler, A numerical analysis of various cross sheet resistor test structures, Solid-State Electronics **20**, 539 (1977).

[12] QCoDeS data acquisition framework, (https://microsoft.github.io/Qcodes/).

[13] DECS↔VISA, a wrapper to provide VISA-like communication to the Quantum Design - Oxford DECS platform., (https://github.com/decs-visa/_decsvisa).

[14] C. E. Becker, Transport Properties of Modulation-Doped $\mathrm{Si/SiGe}$ Quantum Well Structures, PhD thesis, University College London, 2002.

[15] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes: The Art of Scientific Computing* (Cambridge University Press, 2007).

[16] L. Wang, M. Yin, A. Khan, S. Muhtadi, F. Asif, E. S. Choi, and T. Datta, Scatterings and quantum effects in $\mathrm{(Al,In)N/GaN}$ heterostructures for high-power and high-frequency electronics, Phys. Rev. Appl. **9**, 024006 (2018).

[17] H. Cao, J. Tian, I. Miotkowski, T. Shen, J. Hu, S. Qiao, and Y. P. Chen, Quantized Hall effect and Shubnikov–de Haas oscillations in highly doped $\mathrm{Bi_2Se_3}$: Evidence for layered transport of bulk carriers, Phys. Rev. Lett. **108**, 216803 (2012).

[18] I. M. Lifshitz and A. Kosevich, Theory of magnetic susceptibility in metals at low temperature, J. Exp. Theor. Phys. **2**, 636 (1956).

[19] L. M. Roth, Semiclassical theory of magnetic energy levels and magnetic susceptibility of Bloch electrons, Phys. Rev. **145**, 434 (1966).
[20] LaTex, a document preparation system, (https://www.latex-project.org/).
[21] Markdown, syntax basics, (https://daringfireball.net/projects/markdown/basics).
[22] Pandoc, a universial document converter, (https://pandoc.org).
[23] pandoc-crossref, a pandoc filter for cross-references, (https://lierdakil.github.io/pandoc-crossref/).
[24] LaTeX/Mathematics Wikibook, a wikimedia project, (https://en.wikibooks.org/wiki/LaTeX/Mathematics).
[25] BibTeX, a tool and a file format used to process lists of references, (https://www.bibtex.org/).
[26] CSL, the citation style language, (https://citationstyles.org/).
[27] bibutils, a program set that inter-converts between various bibliography formats, (https://linux.die.net/man/1/bibutils).
[28] YAML, a human-readable data serialization language, (https://yaml.org).

About TeslatronPT Plus

TeslatronPT Plus is an open-architecture low-temperature measurement system that builds on Quantum Design Oxford's proven cryogen-free low-temperature and superconducting magnet technology. It offers fully automated control of temperature and magnetic field parameters, with integrated data logging within our software environment. The system seamlessly integrates with measurement instrumentation, such as that from Lake Shore (the M81 & M91) and other third-party devices, so is easily configurable across experimental setups.

With open-source Python measurement scripts **Teslatron**PT Plus simplifies instrument management, enabling automated measurement sequences and secure data handling. Combining advanced hardware with powerful software tools, it delivers a simple, flexible, and future-proof solution for electrical transport measurements at low temperatures and in high magnetic-fields.

Find out more about Quantum Design Oxford - **Teslatron**PT Plus