Development of Automatic Temperature Compensation Software for OFS Data

by Malcolm Scott

Doctoral Training Centre
Photonic Systems Development

Supervisor: Kenichi Soga, 2009/2010

I hereby declare that, except where specifically indicated, the work submitted herein is my own original work.

Malcolm Scott
24th May, 2010

This report contains 9289 words.

Development of Automatic Temperature Compensation Software for OFS Data

Malcolm Scott

Technical Abstract

This report describes the design and successful implementation of a standalone software application to aid geotechnical researchers and structural civil engineers in the processing and analysis of data from recently-developed Optical Fibre Sensing (OFS) techniques: photonic distributed strain sensors in which sensing optical fibres are embedded in a concrete structure during construction, or otherwise fixed to the structure after construction. The aim in installation of these sensors is to determine how strain in a structure changes over time, during successive stages of construction and as the structure ages.

These sensors make use of the technique of Brillouin optical time-domain reflectometry (BOTDR) – the measurement of the Brillouin backscattering effect in optical fibres, which is affected by changes in both local mechanical strain and temperature.

Isolation and removal of the effect of temperature from the data, in order to produce data on mechanical strain only, is important before the data can be analysed. The current preferred method involves the use of a separate fibre installed alongside the primary sensing fibre which is subjected only to changes in temperature and not to strain; I provide an overview of this technique in comparison with alternatives, and derive a formula for temperature compensation using the results of Mohamad (2008):

$$\varepsilon_M \;=\; \varepsilon_{z\_\text{strain}} - \left(\frac{\alpha_a + \alpha_{\text{concrete}}}{\alpha_a + \alpha_{n\_\text{temperature}}} \right) \varepsilon_{z\_\text{temperature}}$$

where ε_M is the pure mechanical strain, $\varepsilon_{z\_\text{strain}}$ is the measured total apparent strain on the strain-sensing fibre, $\varepsilon_{z\_\text{temperature}}$ is the measured strain on the temperature-sensing fibre, α_a is a constant termed the "temperature-induced apparent strain" which is calculated to be $19.47 \times 10^{-6}\ \text{K}^{-1}$, α_{concrete} is the thermal expansion coefficient of the concrete in which the primary fibre is embedded, and $\alpha_{n\_\text{temperature}}$ is the thermal expansion coefficient of the temperature-sensing fibre (a property of the type of cable used, for example $4.2 \times 10^{-6}\ \text{K}^{-1}$ for the commonly-used Unitube cable).

The developed software goes beyond simply performing a temperature compensation calculation; the user is guided, via a user-friendly graphical interface, from the initial import of raw data sets all the way through to the initial analysis of changes in strain over time. Data sets are plotted at high quality throughout the process as they are created, and the graphs and processed data can be exported for use in publishable material.

The software has been well received by researchers in the Geotechnical Research Group of the University of Cambridge Department of Engineering.

Contents

List of Figures

CHAPTER 1

Introduction

This project report concerns the development of software to assist in the processing and analysis of detailed data on the strain within man-made structures. The data is gathered from Optical Fibre Sensing (OFS) apparatus – a pair of optical fibres with different properties embedded in or attached to the structure, together with equipment to perform Brillouin Optical Time-Domain Reflectometry (BOTDR) using those fibres. Two different fibres are used in order to isolate strain from thermal effects; the method of calculating strain given BOTDR responses from the two fibres is given in Section 2.3 (page 11).

Before describing the theory behind this, I will first give a high-level view of the application of this technique in the field.

1.1 Strain Sensing in Civil Engineering

The ability to measure the strain of a structure can be important for a variety of reasons. For old structures, it can be useful in evaluating their safety, in particular during or after a change to the surrounding environment; Fujihashi et al. [1] for example describe a strain sensing system (based on optical fibre sensing) installed to monitor the effects of a new subway tunnel on an existing older telecommunications tunnel, and Mohamad [2, §4.1] describes a similar setup to determine the effect on an old, brick-lined Thameslink tunnel of a new railway tunnel being drilled beneath it through London clay (see Figure 1.1). For newly-built structures, it can be instructive to track how strain increases as construction work continues and the load on the structure or on piles increases, as discussed by Ohno et al. [3].

The traditional method of measuring strain in civil engineering is the vibrating wire

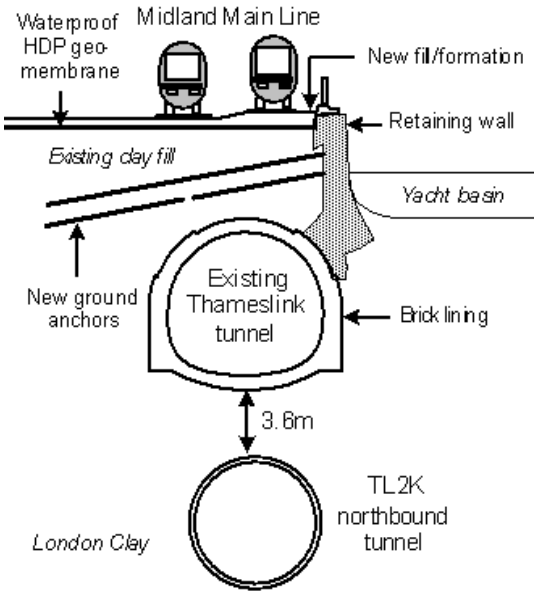

Figure 1.1: Vertical cross-section (not to scale) of old Thameslink tunnel instrumented with BOTDR distributed strain sensors (reproduced from Mohamad [2, §4.1])

strain gauge (VWSG). This device makes use of the effect of additional strain on the resonant frequency of a length of wire under tension. VWSGs are known to be highly accurate and stable over a long period of time, but their weakness is that a single VWSG can only return strain data for a single point; typically a structure so instrumented will have a limited number of VWSGs as each must be installed individually, so strain data from that structure will have low spatial resolution.

Brillouin Optical Time-Domain Reflectometry (BOTDR) is a more recent technique which has several advantages over VWSGs. Most significantly, BOTDR provides a *distributed* strain sensor: optical fibres are installed along the length (or height) of the structure of interest, and a detailed strain profile can be obtained for the length of the fibres – the whole optical fibre itself is the sensor. Figure 1.2, showing strain data from two minipiles instrumented with both VWSGs (four per pile) and BOTDR, illustrates the immediate advantage of BOTDR in this regard, and also attests strong agreement between VWSG and BOTDR. Mair [4] describes the full set of advantages of the use of BOTDR for strain measurement thus:

(a) It has shown good comparison with vibrating-wire strain gauge measurement in piles, and has already been used successfully for a number of piling projects.

(b) It has provided valuable strain data in the Thameslink masonry tunnel during construction of a new tunnel beneath.

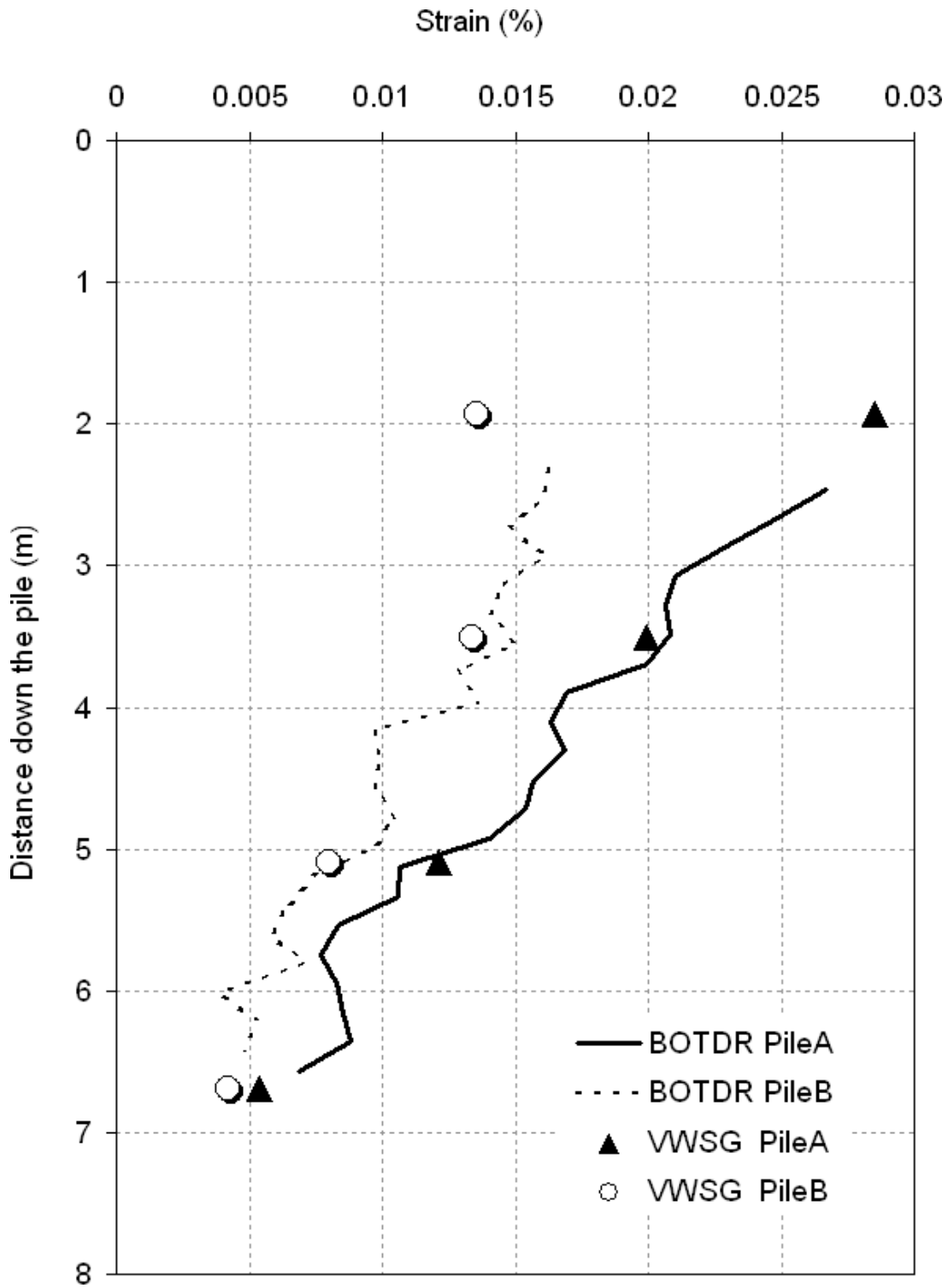

Figure 1.2: Comparison of BOTDR and VWSG strain output for two minipiles (reproduced from Bennett et al. [5])

Figure 1.3: Addenbrooke's Access Road bridge during construction; strain-sensing fibres are embedded in some of the cast concrete beams visible and also in several of the piles supporting the far abutment and the near pier.

(c) The measurement of a continuous strain profile is a big advantage over measurements at discrete locations.

(d) The low cost of installation is attractive.

The photonic theory which underpins BOTDR is explained in Chapter 2 (page 9).

1.2 Case Study: Addenbrooke's Access Road Bridge

During this work, my ongoing case study and source of sample data has been provided by a recent civil engineering project in Cambridge. The Addenbrooke's Access Road was commissioned by Cambridgeshire County Council to provide improved access to the expanding Addenbrooke's Hospital; the project is described on the website

Figure 1.4: BOTDR fibres emerging from the ends of precast beams; the black fibres are for temperature compensation

Figure 1.5: Pile cage instrumented with BOTDR fibres undergoing careful installation into its borehole

of Road Traffic Technology [6]. Construction began in 2008 and is now close to completion. The road crosses the railway south of Cambridge station on a bridge (shown under construction in Figure 1.3), and some parts of this bridge were instrumented with BOTDR distributed strain sensors by members of the University of Cambridge Department of Engineering in collaboration with the constructors, Jackson Civil Engineering, Carillion Piling and Tarmac Limited. The bridge is formed of three spans, supported on two piers and two abutments; each pier and each abutment rests on several piles. Instrumented with BOTDR optical fibres are:

(a) **Beams in the span between the west abutment and the west pier:** The beams were precast off-site by Tarmac Limited and fibres were loosely taped to the internal steel frame immediately before casting. See Figure 1.4.

(b) **Piles in the west pier:** The fibres were attached to the pile cage immediately before being lowered into the boreholes. See Figure 1.5.

(c) **Piles in the west abutment:** As above.

It was always expected that some of the installed fibres would not survive the harsh environment of a construction site – besides the concrete pouring and casting process, the pile fibres had to endure the removal by crane of the excess concrete surrounding the fibres at the head of the pile which caused the fibres to be dragged through a hole in a lump of concrete – and indeed several fibres were deemed unusable[1]. However, the fibres embedded in two beams and seven piles (three in the west pier and four in the west abutment) remained at least partially serviceable.

In all cases, the fibre pairs form a loop along one side of the beam or pile to the inaccessible end (e.g. the bottom of the pile) where they turn around within the concrete and return to the accessible end along the other side. This has three advantages:

(a) The data sets from the two sides may be separated and averaged to reduce noise in the data and to obtain an estimate for the axial compressive strain.

(b) Changes in strain on either side of the beam or pile can be compared, which could provide further information to indicate for example whether the structure is bending or twisting.

(c) Two ends of the fibre are available from which readings can be taken. If the fibre(s) are broken within the beam or pile – for example at the bottom of the pile due to the pouring of concrete combined with the relatively narrow bend radius, as happened in several cases – readings can be taken from both ends of the fibre and stitched together afterwards. (If the fibre has multiple breaks, the data from between the breaks will however be irretrievably lost.)

This bridge was instrumented so that the changes in strain over a time span of multiple years, within the piles in particular, could be recorded and analysed with reference to the soil composition and other environmental factors. That task falls to various of my colleagues – Tina Schwamb, Echo Ouyang, Andy Leung and Te Janmonta, all under the supervision of Professor Kenichi Soga – and it is they who have highlighted the need for a software tool to aid in this analysis.

1.3 Current Data Analysis Techniques

The data sets resulting from a single run of BOTDR along the length of a fibre are large. The BOTDR equipment – in our case, a Yokogawa AQ8603, as pictured in Figure 1.6 –

[1] To their credit, the contractors did attempt to mend one snapped fibre with electrical tape.

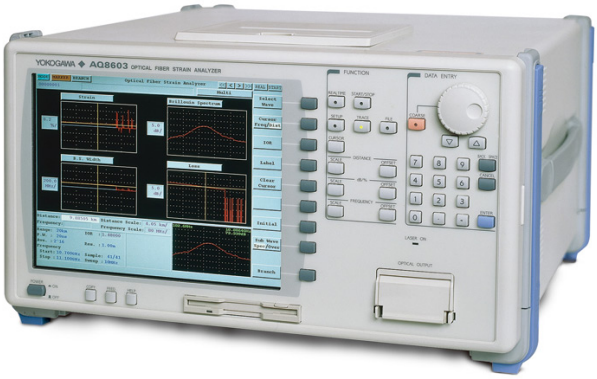

Figure 1.6: Yokogawa AQ8603 BOTDR strain analyser

contains some built-in software to perform the initial processing of Brillouin backscatter frequency response and convert to apparent strain, given various parameters such as fibre refractive index, frequency range and desired output distance resolution. The output from this process is a strain profile containing – depending on the resolution and the model of analyser – on the order of 10,000 to 100,000 data points for a single fibre.

As described above, there are nine usable sensing fibres in the Addenbrooke's bridge and a further nine temperature fibres. Due to breakage, some of these fibres have readings taken from both ends. Usually each reading is repeated up to three times in order to minimise errors. As a result, a single day's measurements is likely to total several hundreds of thousands of data points.

Since the objective of this study is to investigate changes in strain over time, this process must be repeated over the course of several years. This is a manual process since the equipment must be brought to the site each time; over the first 18 months of the bridge's life, measurements have been taken on seven different occasions, leading to a total number of data points well in excess of two million spread across 164 files. Managing, processing and analysing this data is a challenge.

Currently, analysis typically takes place in a vast spreadsheet, usually in Microsoft Excel, aided in some cases by Matlab programs. Alignment of data sets – due to different lengths of strain and temperature fibres, sometimes involving changes in length between readings due to re-splicing – is performed manually; discarding meaningless data beyond the end of the fibre, which appears just as noise in the output, is also done manually. As the data is multidimensional – two sets of linear measurements, repeated multiple times for each fibre on each day, and repeated over several days – an unwieldy number of separate spreadsheets is maintained with a certain amount of

manual copy-and-paste. Multiple of my colleagues have remarked that the process is very time-consuming, and also varies from person to person as different corners are cut to save time and spreadsheet space.

The results from this semi-manual analysis are good, but much time which could have been spent investigating geomechanical implications is instead wasted on the mechanics of the spreadsheet itself.

It is clear that the process described above would benefit considerably from automation. I set about to implement a piece of software, targeted at direct use by both academics and civil engineers, to achieve this. The first step was to capture the requirements for the software, a process I describe in Section 3.1 (page 17), but I will first cover the theory of BOTDR distributed strain sensing and temperature compensation.

CHAPTER 2

Theoretical Background

In this chapter I will provide an overview of the principles which underpin Brillouin-scattering-based distributed optical fibre strain sensing. For a more in-depth explanation, Mohamad [2] provides an invaluable reference.

2.1 Optical Scattering Effects

Optical fibre sensing is based on the measurement of scattering effects. The underlying principle is that any transparent medium, such as the glass of an optical fibre, will to some degree influence light passing through it: the fraction of photons which interact directly with the medium rather than passing straight through (which is very small in a transparent medium, but nevertheless non-zero) may be *scattered*, or absorbed and re-emitted, via a number of different processes. The re-emitted photon may have the same wavelength as the absorbed photon (in the case of Rayleigh scattering), or the wavelength may be changed slightly (Brillouin or Raman scattering). With Brillouin and Raman scattering, increases in wavelength due to scattering are Stokes components, whereas decreases in wavelength are Anti-Stokes components; since these are symmetrical, usually only one or other component is measured.

Photons which undergo scattering may end up travelling in any direction. In an optical fibre what matters most is whether they travel away from the light source, i.e. in the same direction as the incident photons along the fibre, or back towards the source – *backscattering*. Either direction of scattering can be used to measure properties of the fibre; in a distributed sensor it is usually more convenient to work with backscattered light as it only requires a connection to one end of the fibre. However, the amplitude of backscattered light is very low compared with the input light, so a high-powered

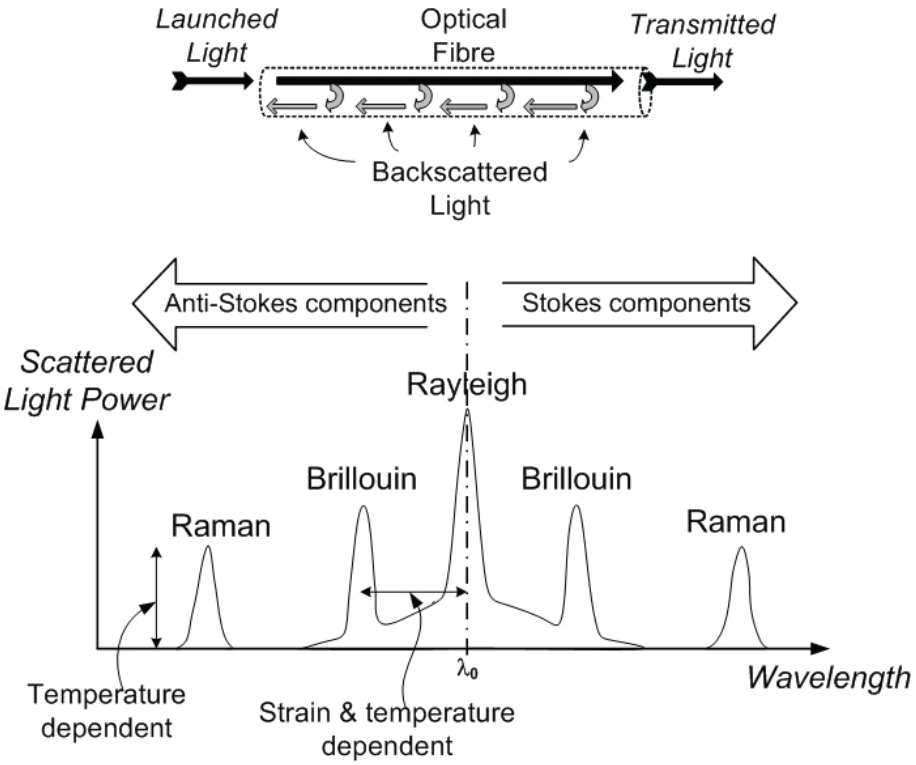

Figure 2.1: Types of backscattered light (reproduced from Mohamad [2, §2.3])

pulse laser is needed along with a highly-sensitive receiver.

Rayleigh and Brillouin scattering, as previously mentioned, increase or decrease the wavelength of photons. The degree by which they do so depends upon properties of and conditions in the material. In Rayleigh scattering, the wavelength shift is a direct property of the molecular composition of the medium and does not change over time; however the amount of scattering – i.e. the power of the scattered light – depends upon the temperature of the material. Strain does not affect Rayleigh scattering. Brillouin scattering is more useful for distributed sensing as the change in wavelength is a function of strain on the material (and also of temperature).

These effects are summarised in Figure 2.1.

2.2 The Effect of Strain on Brillouin Scattering

Brillouin scattering arises because no medium is perfectly homogeneous. Slight imperfections in the glass of an optical fibre give rise to minute shifts in density along the length of the fibre, and these shifts can give rise to phonons when the material is

excited. The phonons which propagate axially along the fibre themselves cause slight variations of density – in this case, due to their wave nature, a periodic variation. This translates into a localised periodic component to the refractive index, which acts in a manner comparable to that of a Bragg grating.

A medium under compression or tension undergoes a change of density and of refractive index (Jones et al. [7]). As a consequence of this, Brillouin scattering is also affected. The Brillouin wavelength shift is a function of the phonons' propagation speed; it has been shown by Horiguchi et al. [8] that the shift expressed in frequency terms increases linearly with the longitudinal strain of an optical fibre.

Mohamad [2, equation 2.2] makes use of the results of Cotter [9] in order to derive an equation relating the frequency shift to longitudinal strain:

$$\nu_b(z) \quad = \quad \nu_{b0} + M\varepsilon(z) \tag{2.1}$$

where $\varepsilon(z)$ is the strain at position z along the fibre, $\nu_b(z)$ is the Brillouin frequency shift at position z, and ν_{b0} is the Brillouin frequency shift at zero strain, calculated to be approximately 11 GHz at a wavelength of 1550 nm.

Of course, it is not possible to measure the Brillouin frequency shift at a particular position directly, especially given access only to the ends of the fibre. However, the speed of light in the medium is by definition $\frac{c}{n}$ where c is the speed of light in a vacuum (3×10^8 m/s) and n is the refractive index of the medium; any variation of refractive index in the medium due to imperfections and phonon propagation is sufficiently small as to be negligible for this purpose. Therefore the technique of optical time-domain reflectometry (OTDR) can be used: a very brief pulse of laser light is input into the fibre, and return pulses are detected. The time between initial pulse and return pulse can be converted into a distance along the fibre.

The measurement of the frequency of Brillouin-scattered return pulses using OTDR is referred to as *Brillouin Optical Time-Domain Reflectometry*, BOTDR.

2.3 Temperature Compensation

Brillouin scattering is affected by strain on the optical fibre, as described above, but also by variation in temperature. It is important when analysing a structure to separate the effect of temperature on the fibre from the effect of actual mechanical strain within the structure; ignoring the effect of temperature on BOTDR has been shown by Smith et al. [10] to lead to substantial errors in strain data. For this reason, the strain data must undergo a temperature compensation process.

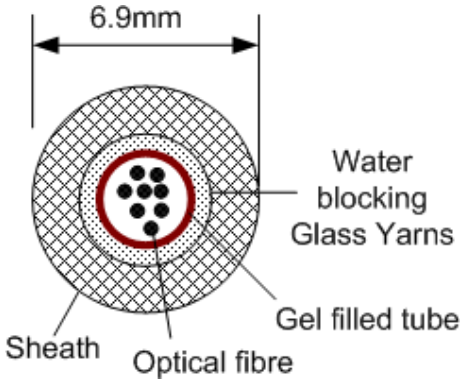

Figure 2.2: Cross section of Unitube fibre optic cable

2.3.1 Obtaining Temperature Data

Temperature compensation requires, in addition to the combined strain and temperature data set from BOTDR, a separate data set of temperature only. Ideally, the temperature data set should also be taken using a distributed sensor, along the same route covered by the BOTDR data set. Some earlier deployments instead used a point temperature sensor – e.g. a few coils of a single sensing fibre which were under no mechanical strain – and assumed uniform temperature, but these methods are not discussed further here as they are outdated.

There are two major methods for obtaining the temperature data set in a distributed manner.

Parallel Temperature-Only Fibre

In this method, BOTDR is performed on two parallel fibres, one of which is installed such that it is only affected by temperature and not mechanical strain.

The temperature-only fibre is contained within a loose tube filled with a gel which allows the fibre to slide relative to the outer sheath; in the case of the Addenbrooke's Access Road bridge, "Unitube" cable (pictured in Figure 2.2) was used. Due to the gel layer, this fibre is under no mechanical strain but does nevertheless respond to temperature; since it is mounted alongside and close to the main sensing fibre, the apparent strain due to temperature can be recorded along the same path as the total apparent strain due to both mechanical strain and temperature. Such data is relatively easy to work with as the absolute temperature need not be calculated; the data set from the temperature-only fibre can (with a simple multiplication factor, derived below) simply be subtracted from the data set taken from the strain-and-temperature fibre.

Raman OTDR

As described in Section 2.1 (page 9), the intensity of Raman scattering varies only with temperature and not with strain (and the frequency response is fixed, assuming a stable material). Thus, one can obtain both temperature-and-strain data and temperature-only data from the same fibre, using both Brillouin OTDR and Raman OTDR respectively.

The equipment required to perform both types of OTDR in parallel is very specialised; the only known commercially-available strain analyser capable of performing such independent temperature measurements is the Sensornet Distributed Temperature and Strain Sensor (Parker et al. [11]) which has other limitations and is not in use in this research group. However, it seems that analysers using this technique may become popular as it becomes more widely available, as it allows for immediate temperature compensation at the time of data recording, and also permits cheaper sensor deployments using only a single fibre.

As the Yokogawa equipment in use in this research group does not support Raman OTDR, my software was developed for the former case involving a separate temperature-sensing fibre.

2.3.2 Temperature Compensation Calculation

My method of temperature compensation is based on that presented by Mohamad [2, §5.2], with the equations rearranged to suit the parallel temperature-only fibre setup described above. For these purposes I will assume – as is the case with our Yokogawa analyser – that raw data sets take the form of *total apparent strain* readings, ε_z, taken along the length of the fibre.

For BOTDR in any optical fibre, total apparent strain ε_z is formed of the sum of two components: the actual *mechanical strain* ε_M and the unwanted additional influence, the *thermal strain* $\Delta\varepsilon_T$.

$$\varepsilon_z = \varepsilon_M + \Delta\varepsilon_T \tag{2.2}$$

Mohamad [2, equation 3.18] derives:

$$\Delta\varepsilon_T = (\alpha_a + \alpha_n)\Delta T \tag{2.3}$$

where ΔT is the change in temperature in kelvin (K), $\Delta\varepsilon_T$ is the resulting change in measured apparent strain, α_a is the *temperature-induced apparent strain* of the fibre itself

– a constant calculated to be 19.47×10^{-6} K^{-1} – and α_n is the net thermal expansion of the material surrounding the fibre.

This latter constant α_n merits careful consideration. Mohamad and Soga [12] explain that one must pick from a few different options when assigning α_n depending upon the cables in use and their method of installation. There are several materials surrounding the fibre which may expand and compress the fibre when heated; however one of these materials will likely have a dominant effect allowing others to be discounted. For example:

- For a low-cost single fibre in a plastic sheath, suspended in air, the only surrounding material which matters is the sheath (the effect of the air will be several orders of magnitude less and can be ignored). Therefore in this case, the thermal expansion coefficient of the sheath is required. This depends upon the type of fibre in use; a typical value measured by Mohamad [2, table 3.4] is $\alpha_n = 4.37 \times 10^{-6}$ K^{-1}.

- For a fibre embedded in a rigid sheath in concrete, although the sheath will have some effect, the effect of the concrete will be much greater and the effect of the sheath can – and indeed should, according to Mohamad and Soga [12], in order to avoid erroneous results – be ignored.

  Selecting a value for the coefficient of thermal expansion of concrete (α_{concrete}) is itself not simple as this depends upon the constitution of the concrete used. Values for α for concrete commonly used in the UK vary from about 8–13 K^{-1} according to Browne [13], although Sellevold and Bjontegaard [14] suggest that early-aged concretes may have a lower coefficient of thermal expansion due to higher water content.

  This scenario applies in the case of the strain cable data for Addenbrooke's Access Road bridge. For testing I used a nominal value of 10 K^{-1} as recommended by the standard EN1992-1-1.

- For Unitube cable embedded in concrete, as used in the Addenbrooke's Access Road bridge temperature cable, the problem actually becomes considerably simpler again. Due to the construction of Unitube cable (pictured in Figure 2.2) – more specifically due to the gel layer and the air gaps around and within the fibre bundle – the surrounding concrete does not directly compress the optical fibre. Mohamad [2, table 3.4] gives a value for Unitube cable of $\alpha_n = 4.2 \times 10^{-6}$ K^{-1}, notably similar to the value for a single-sheathed fibre suspended in air.

It is important to note that when separate temperature-sensing and strain-sensing fibres are in use, the value of α_n will almost certainly be different for the two fibres. As a result, the temperature data set cannot simply be subtracted from the strain data set.

So, combining equations 2.2 and 2.3 and taking the two sensing fibres, termed for the sake of brevity "strain" and "temperature" respectively (but noting that the "strain" cable is affected by both strain *and* temperature):

$$\varepsilon_{z\_\text{strain}} = \varepsilon_{M\_\text{strain}} + (\alpha_a + \alpha_{n\_\text{strain}})\Delta T \tag{2.4}$$

$$\varepsilon_{z\_\text{temperature}} = \varepsilon_{M\_\text{temperature}} + (\alpha_a + \alpha_{n\_\text{temperature}})\Delta T \tag{2.5}$$

By design, $\varepsilon_{M\_\text{temperature}} = 0$: that is to say, the temperature cable is unaffected by mechanical strain. As previously discussed, $\alpha_{n\_\text{strain}} = \alpha_{\text{concrete}}$, whereas $\alpha_{n\_\text{temperature}}$ is a property of the temperature cable.

We can now solve for the mechanical strain $\varepsilon_M = \varepsilon_{M\_\text{strain}}$ in terms of the recorded data sets $\varepsilon_{z\_\text{strain}}$ and $\varepsilon_{z\_\text{temperature}}$. From equation 2.4:

$$\varepsilon_M = \varepsilon_{z\_\text{strain}} - (\alpha_a + \alpha_{\text{concrete}})\Delta T \tag{2.6}$$

From equation 2.5:

$$\Delta T = \frac{\varepsilon_{z\_\text{temperature}}}{\alpha_a + \alpha_{n\_\text{temperature}}} \tag{2.7}$$

Substituting back into equation 2.6:

$$\varepsilon_M = \varepsilon_{z\_\text{strain}} - \left(\frac{\alpha_a + \alpha_{\text{concrete}}}{\alpha_a + \alpha_{n\_\text{temperature}}}\right)\varepsilon_{z\_\text{temperature}} \tag{2.8}$$

This is the formula required for temperature compensation of BOTDR data with a separate temperature-only fibre.

The factor $\frac{\alpha_a + \alpha_{\text{concrete}}}{\alpha_a + \alpha_{n\_\text{temperature}}}$ is the key to this temperature compensation process. This ratio of the combined temperature expansion coefficient of the strain cable to the combined temperature expansion coefficient of the temperature cable is termed R_α by Mohamad [2] and, where a separate temperature cable is used, is expected to be greater than 1.

CHAPTER 3

Software Design and Implementation

3.1 Requirements Capture

The first step in any software development project is to determine the requirements for the software in collaboration with the end users. The primary targeted end users of the strain analysis software are academics undertaking geotechnical or civil engineering research using distributed strain sensors to investigate the behaviour of structures[1].

I met with several PhD students in the Geotechnical Research Group of the University of Cambridge Department of Engineering who were undertaking such work under the supervision of Professor Kenichi Soga, some of whom were nearing completion of their respective projects and could teach me about their current analysis techniques, whilst others were just beginning their research and may themselves later use the software I was to develop.

I also visited the Addenbrooke's Access Road bridge with some of these students to watch the data acquisition process and learn about the causes of imperfections in data, for example changes in length and loss characteristics of fibres due to repair work – cutting off damaged fibre and using fusion splicing to attach a new connector – which was necessitated by vandalism of the site.

As a result of these meetings I drew up the following set of requirements for processes which the software must perform. I aimed to make this software useful in the general case, rather than assuming specifics of the Addenbrooke's Access Road bridge sensor deployment; the only deployment characteristic I assume is the existence of separate strain and temperature cables.

[1]The software is also expected to be useful by non-academic civil engineers, but due to time constraints I did not primarily target such users during this project and did not seek to determine their requirements.

3.1.1 Parsing of Raw Data from BOTDR System

The raw data from the Yokogawa AQ8603 optical fibre strain analyser used by this research group can be extracted from the analyser in two forms. First is the "`.str`" format which is the native file format of the analyser; every run produces a `.str` file and this contains all data recorded by the analyser during that run, including the full set of Brillouin frequency responses. However, this format is proprietary, binary and undocumented as it does not appear to be intended for use in non-Yokogawa software; reverse engineering it would be a significant task by itself.

However, the analyser or software provided by Yokogawa can read a `.str` file and export data sets of various types – in particular the calculated strain, but see Section 3.3.1 (page 26) for further details – in a more useful, text-based format. It was thought at the start of the project that these could only be exported by the analyser itself, which was inconvenient as there is only one analyser in the research group and it is in popular demand; however we discussed this issue with Yokogawa and were provided with a piece of Windows software capable of performing this conversion. It is therefore reasonable to expect the input to my software to be one or more text files exported from the original `.str` file, rather than the `.str` file itself.

However, these text files are still sparsely documented and the format must undergo a small amount of reverse engineering before it can be used.

3.1.2 Initial Cleansing of Raw Data

The raw data requires a certain amount of processing before it can be analysed; this process has previously been done manually but can be automated by my software.

Different researchers have cleansed the data in different ways; the following procedure is a superset of the various processes of all the researchers I talked to.

(i) Discard bogus data beyond the end of the fibre

The analyser does not know the exact length of the fibre, nor does it make any attempt to estimate this. It simply detects that beyond a certain point (i.e. the end of the fibre) the loss is unsurprisingly very high, and amplifies the nonexistent signal from that section of the fibre with a very high gain. Recorded strain data from beyond the end of the fibre is therefore entirely bogus, consisting only of highly-amplified noise, and can to an extent be detected by software using heuristics. However, manual correction must be possible as any such heuristic-based algorithm will not be perfect, especially

where there genuinely is high attenuation in the fibre caused by other factors such as damage or poor splicing.

(ii) Align and average repeat readings

Typically up to three data sets are taken for each fibre pair on each day of measurement. It is not necessarily the case that the fibre is the same length or that the useful data starts at the same offset in each case – sometimes a fibre is broken, cut and re-spliced between readings, especially if the first reading indicates that the fibre may have been poorly spliced (which would lead to high loss and noisy data).

A cross-correlation algorithm can be used to automatically align repeat readings before they are averaged.

(iii) Mark beginning and end of useful portion of data, after lead-in fibre

Even when the data has been trimmed so that only valid strain data from the full length of the fibre is present, not all of the data is useful. In almost any distributed fibre-based sensor, a portion of the fibre will be present solely to connect the structure of interest to the analyser. In the case of the Addenbrooke's Access Road bridge, this consisted both of fibre in fixed ducts attached to the structure – leading from a manhole down to the pile, or from an access cabinet up the abutment wall to the beams – and a length of fibre in the open air connected to the analyser, usually including a coil of several metres of spare fibre. Whilst the strain in this fibre will have been recorded by the analyser, it is not of interest; the strain in the open-air portion of fibre is expected to change frequently as it is moved around between readings, and is expected to contain at least one spike in the strain data due to a fusion splicing point connecting the sensing fibre to a short lead with an ST connector suitable for plugging into the analyser.

In more complex deployments, such as that at the Singapore Circle Line as described by Mohamad [2, §4.2], there could be a kilometre or more of lead-in fibre.

It will not be possible in the general case for the software to detect the position of the structure of interest, so the user must provide this information manually. However, where repeat readings are taken, there is no need for the user to select this for each reading separately.

(iv) Correct for erroneous calibration of analyser

Unfortunately, several existing data sets on the Addenbrooke's Access Road bridge, and presumably on other sites as well, were recorded with erroneous analyser settings

as those people who took the readings were unaware of at least one value which should be set, usually the refractive index (referred to as "IOR", for Index Of Refraction, on-screen on the analyser and in Yokogawa software).

The analyser documentation from Yokogawa states that one must specify the refractive index of the sensing fibre on the analyser before taking any readings, as this affects the conversion from return pulse time to distance. Luckily, correcting for this in software is possible since the refractive index setting used by the analyser is stored with the data; the software can prompt the user for the actual refractive index of the sensing fibre and apply a constant correction factor to all length values.

It is also possible to correct this using the Yokogawa software utility whilst converting from `.str` to text format.

(v) Alongside each strain data set, maintain a temperature data set

As above, for brevity the term "strain data set" refers to data from the fibre subject to both mechanical and thermal strain, and the term "temperature data set" refers to the data from the (Unitube or similar) fibre which is isolated from mechanical strain and subject only to thermal strain.

Every strain data set should have a corresponding temperature data set. There may be a different number of repeats of the strain data sets and temperature data sets, but since the repeat readings should be averaged before temperature compensation this should not present a problem.

(vi) Align temperature data with strain data

In the general case, this is not possible to automate as there is little correlation between temperature and strain data sets, so the user must be involved in order to align these data sets before temperature compensation can take place.

It would be possible to automate this only if further assumptions are made about the characteristics of the deployment – for example, the Addenbrooke's Access Road bridge deployment uses a "there and back" fibre loop, so the data is roughly symmetrical; this could be used to detect the far end of the loops of both fibres and align based on the knowledge that both fibres take exactly the same route. However I believe that making such assumptions would have been an unwise design decision as the software should also be useful in cases where these assumptions do not hold. Having the user align the data sets manually, for example using a simple drag-and-drop process, is unlikely to be considered overly arduous.

(vii) Compute and store temperature-compensated strain data

By this point, the software is ready to perform the temperature compensation algorithm described in Section 2.3 (page 11).

Once temperature compensation has been performed, the data should be stored for comparison with equivalent data sets taken at other times.

Deployment-specific features

A few additional features were considered to be useful in the specific case of the Addenbrooke's Access Road bridge, such as:

- Reverse data sets when taken from the opposite end of a fibre loop
- Stitch together data sets taken from each of a fibre loop, where the loop is broken part way along, in order to produce a more complete data set
- Optionally, split a data set from a "there and back" fibre loop in half and treat the two halves as multiple data sets for the same beam or pile

The decision was made to place these beyond the scope of this software in an effort to maintain the general-purpose nature of the software.

3.1.3 Management and Storage of Data Sets

It is expected that a user of this software will have an existing repository of raw strain data sets that he or she wishes to analyse. The user will start by importing this data and performing temperature compensation, then storing it for future use. Further data sets from the same site or new sites may be added later as BOTDR measurements are taken.

In order to avoid the user having to repeatedly go through the initial data processing stages described above every time he or she wishes to perform data analysis for a particular site, it is necessary to permit the user to maintain a repository of preprocessed data sets – data which has already been averaged and compensated for temperature, and which covers one or more instrumented structures at various times throughout the time frame of interest.

The implementation of this data set repository could take a variety of forms depending on the implementation of the software – which, at this point in the design process, had not yet been decided – but should not be overly restrictive or impose

unwanted methodologies on the users. It should be possible to easily share data sets between multiple users.

It is also necessary to allow the user to export compensated data sets as files for use in other applications – for example, a standard Comma-Separated Values (CSV) format for import into a spreadsheet package.

3.1.4 Presentation of Individual Data Sets

Graphical visualisation of data sets is of course important, although it should be noted that the shape of an individual data set is generally understood to be of little use on its own: it depends more upon initial conditions of installation such as the degree of pretensioning of the fibre and the concrete pouring or casting process, whereas the purpose of strain monitoring is usually to monitor *changes* to the strain within a structure over time with respect to a baseline data set. Nevertheless displaying previews of individual data sets – or averaged data sets across repeat readings on a single occasion – allows the user to see at a glance the quality of data (the degree of noise, the presence of fibre breaks, etc.) and to see the locations of obvious features such as the points at which the fibre enters concrete, and is necessary for the manual alignment step.

3.1.5 Presentation of Change in Strain over Time

This is the ultimate objective, and the final output of this application.

One averaged data set for a given sensing fibre pair – probably the first to be taken, or an average of the first several for increased accuracy – would be selected by the user as the baseline or reference data set. Subsequent data sets for the same fibre pair can then be compared to this reference data set.

Before this can be achieved, however, a couple of final data processing steps may first need to be performed:

(i) Noise filtering

The error inherent in BOTDR strain measurements using current analysers is not insignificant (in the case of the Yokogawa AQ8603, $\pm 30\ \mu\varepsilon$) in comparison with the absolute strain values recorded, and especially in comparison to the potentially very small differences between consecutive data sets, so the data must be filtered to attempt to eliminate noise.

Current researchers such as Mohamad [2, §3.3] and the PhD students with whom I discussed the software requirements have in general used the Savitzky-Golay high degree polynomial filtering method described by Savitzky and Golay [15] whereby a polynomial of specified order is locally fitted to the data within a specified kernel, thus effectively removing high-frequency components.

It is important, in order to avoid surprising artefacts after subtraction, that all data sets are filtered using the same method with the same parameters; as a result, noise filtering should be done at this late stage rather than earlier in the process.

(ii) Resampling

If the data sets to be compared are of differing resolutions, subsequent data sets must be resampled to match the reference data set before they can be subtracted. This may arise due to a change in settings of the strain analyser – either of the resolution directly, or of the refractive index setting which will require compensation by adjusting the resolution (as discussed above).

(iii) Alignment

Once again, the data sets must be aligned. This can take place automatically: since the profiles are expected to be very similar in shape, a cross-correlation algorithm can be used to perfectly align subsequent data sets with the reference data set.

Once the data is aligned, of matching resolutions and relatively noise-free, the reference data can be subtracted from each subsequent data set point-by-point.

The resulting "delta strain" data sets should be plotted graphically; the graph must be exportable from the software in high quality (ideally a vector format) for use in publishable material. The delta strain data sets should also be exportable in a numeric format for use in other applications as before.

3.2 Implementation Choices

Given this set of requirements, the next major task in the production of this piece of software was to decide on the manner in which it would be implemented on a technical level. Several choices needed to be made.

3.2.1 Application Model

The predominant model for software during the last couple of decades has been to distribute the software to end users, in either compiled or source form, where it is run on local computers. However, recently there has been a trend towards "cloud computing" or "software as a service" (which actually bears many similarities to the terminal-based computing model of the 1970s!): that is to say, software accessed remotely via the internet and presented to the user usually via a web browser.

Neither option can be immediately discounted, and these options were carefully evaluated. Cloud computing has many advantages, most significant being its ability to work on any computer platform and operating system able to run a recent web browser, and its ease of use in not requiring installation on end users' computers. Whilst it does require maintenance of servers, there are well-regarded groups within the University[2] which can provide well-specified server facilities for free.

However, the data storage model usually inherent in cloud computing applications has major problems for both software developers and end users. Data is usually stored on the server rather than on end users' computers, which means that the software must handle authentication and authorisation to keep different users' data separate. Users may also dislike this model as their data is out of their control: they must trust the service provider to maintain their data safely and securely, and to continue to provide access to it and to the application indefinitely.

For this application, it was decided that the more traditional computing model of local installation on end users' computers was more suitable, although ideally the application would not mandate the use of one particular operating system. This leads onto the choice of programming language below.

Data storage would, as a result, be in regular files – one per sensor, containing the differences between several temperature-compensated data sets – on the user's computer. These can be manually transferred to another user's computer if sharing is required, for example by email.

3.2.2 Programming Language

Comparison of programming languages forms an entire research discipline within Computer Science; however, pragmatically, for non-Computer Science research software which may fall to unspecified other academics to maintain in the future one must

[2]Such as the Student-Run Computing Facility: `http://www.srcf.ucam.org/`

choose from a fairly small set of well-known and well-documented languages. The choice is further constrained by specific requirements of the software to be written:

- cross-platform support (ability to run on different operating systems);
- graphical user interface support;
- rapid prototyping support;
- high level (the performance benefits of a low-level implementation would be minimal as the application is not expected to be performance-critical);
- availability of good numerical processing routines or libraries;
- object orientation (for easier handling of a hierarchical structure of data sets);
- standalone operation (the software may ultimately be used in a non-academic environment).

Previous work to automate parts of the data analysis process beyond what is possible in a spreadsheet, in particular the work by Echo Ouyang (one of the PhD students with whom I discussed the requirements for this software) have been based on Matlab due to its excellent facilities for numerical processing and analysis. However Matlab falls short on a few of the other requirements, especially the target of producing a standalone, cross-platform application.

The language I settled upon is Python[3]. This is a freely-available, open source language which can be run on all major operating systems and which has a good community, excellent documentation and vast array of freely-available libraries to provide extra functionality. It is well-known amongst computer scientists and the open source development community for the clarity and readability of its code, as confirmed by The Right Tool [16]. Although not previously used in this research group, it has a good reputation for scientific and numeric computing, particularly due to library suites SciPy and NumPy[4]. A further library, Matplotlib[5], also provides Matlab-like graph plotting functionality including the ability to export high-quality graphs in several formats.

For the graphical user interface, I chose the GTK+ toolkit, which is available in Python in the form of the PyGTK library[6]. This toolkit will work on almost any operating system, displaying consistently with other software in the environment. GTK+

[3] `http://www.python.org/`
[4] `http://www.scipy.org/`
[5] `http://matplotlib.sourceforge.net`
[6] `http://www.gtk.org/; http://www.pygtk.org/`

also has the benefit of the Glade[7] rapid application development (RAD) tool from the GNOME project, which allows functional and consistent graphical user interfaces to be laid out and then used directly in code.

3.3 Software Engineering Techniques

Due to the necessity of relying heavily on feedback from researchers more familiar than I with distributed strain sensing and civil engineering in general, I adopted a rapid form of the Spiral Development Model due to Boehm [17] (illustrated for the general case in Figure 3.1). This model focuses on producing prototypes early and often, each of which receives feedback in order to enhance the specification for the next cycle.

In practice, this meant that at my second and each subsequent meeting held every few weeks with my supervisor and my PhD student colleagues I would present a new prototype, which although it borrowed some code from the previous prototype was largely new. The user interface in particular was redesigned several times based on the feedback of my colleagues and on my own ideas for more intuitive designs.

This development model was very successful for this project and facilitated the creation of what I believe to be a high-quality piece of software.

3.3.1 Adaptations of the Specification during Development

The Spiral Model acknowledges that it is rarely possible to produce a complete specification before writing any code, and allows for the specification to be rewritten during the early development cycles. The complete specification given in Section 3.1 (page 17) was the state after a few cycles of Spiral development, after which most of the specifics had been finalised. A few required changes – and additional desired features – arose later on in the development process and were incorporated into the specification during the next Spiral cycle. Two changes of particular interest are given here.

Use of loss data

Recall from Section 3.1.1 (page 18) that the Yokogawa AQ8603 analyser saves its data to files in its proprietary `.str` format, from which can be extracted more useful text-based data files of various kinds. Initially it was thought that only calculated strain data was available in text format; however the Windows utility software obtained from

[7] `http://glade.gnome.org/`

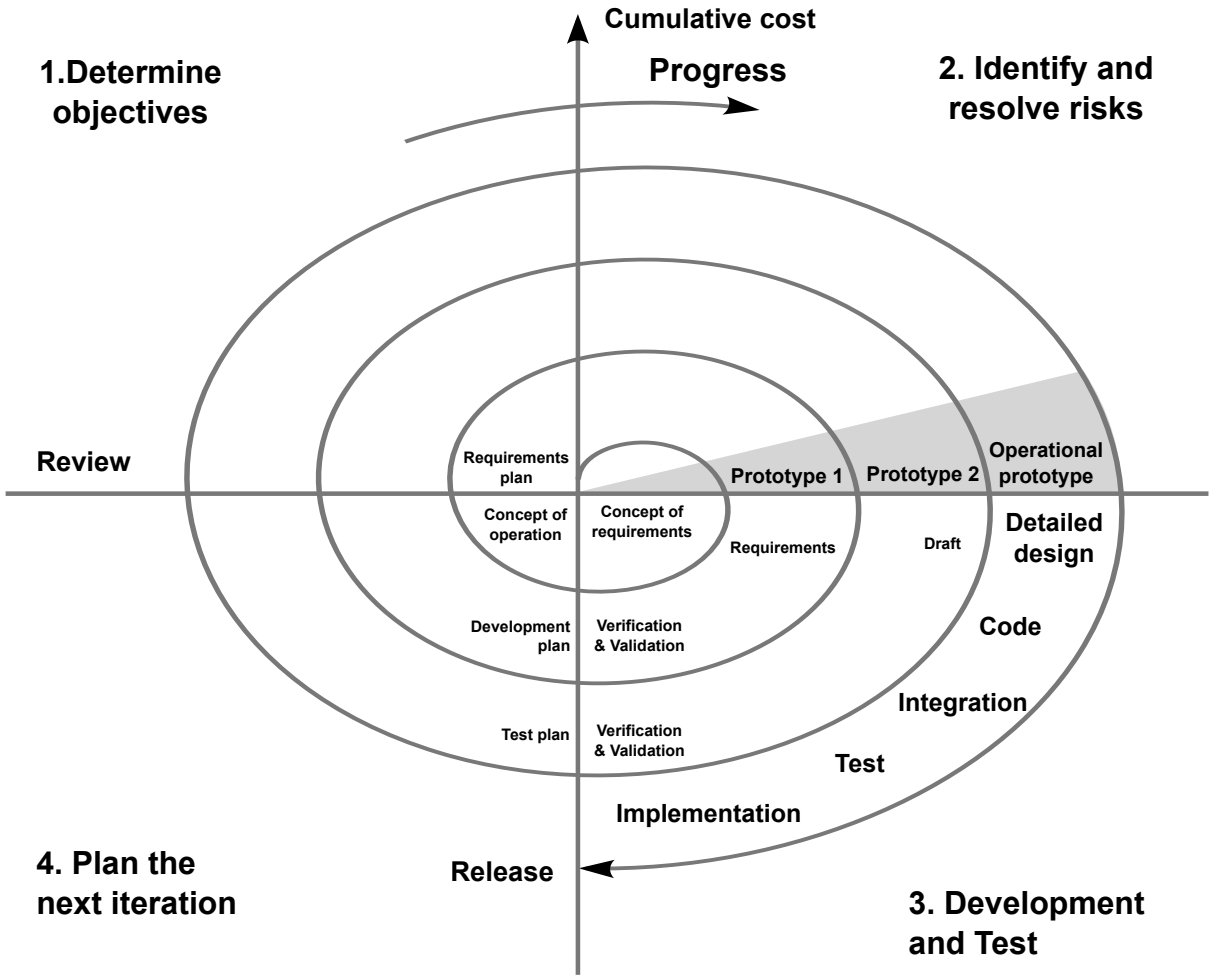

Figure 3.1: Illustration of Boehm's Spiral Development Model

Yokogawa proved to be capable of extracting data of other kinds which could be made use of. The full set of text-based data file types available are:

Strain – as previously described;

Loss – the measured power of the received Brillouin backscattered light as a function of distance (N.B.: despite being termed "loss" this is actually stored as the opposite of loss: a high value in this data set is an indication of low attenuation);

Brillouin spectrum width – expected in general to be the same as the transmitted pulse spectrum width, but could be affected by non-linear effects in the fibre or the accidental detection of light from sources other than Brillouin scattering;

Brillouin spectrum for a particular distance – used by the analyser to calculate a single strain data point, and not especially useful for external software;

Backscatter loss for a particular frequency – similar to Loss, above, but considers the measured power at a specified single frequency only rather than across all frequencies, which is again not especially useful for external software.

Besides the strain data, of particular interest is the loss data, which can be used to estimate attenuation within the fibre and thus provide an estimate of the quality of strain data points. Using the loss data it is possible to accurately locate the end of the fibre, as beyond that point the attenuation is infinite; more generally, high-attenuation data points can be filtered out as they are likely to be imprecise if not bogus. It also facilitates more intelligent averaging of repeat data sets: the repeats may have differing loss profiles due for example to re-splicing or cleaning of dirt from the fibre connector, so low-attenuation data points can be weighted higher when averaging repeated data sets.

As a brief aside, I will discuss the implementation of this feature making use of a unique feature of the NumPy library. First, the loss data points are converted into an array of "confidence" values between 0 and 1. They are then combined with the strain data points in the form of a NumPy *MaskedArray*:

```
self.straindata = numpy.ma.MaskedArray(
        self.straindata,
        mask=numpy.ma.masked_less(self.confidence, 0.1).mask
    )
```

This causes data points with a confidence of less than 0.1 to be marked as masked, which will mean in general that they are considered to be nonexistent. A MaskedArray can be used wherever a regular array can be, for example when performing cross-correlation or when taking a weighted average; the fact that some data points may be masked is taken into account automatically by the library at every stage.

High-resolution alignment

My colleague Tina Schwamb commented during the development process that the distance offset between two data sets may not be an integral number of data points. The output resolution of the Yokogawa AQ8603 analyser is 5 cm; if during re-splicing

2.5 cm is removed from the fibre, cross-correlation of the discrete data may produce slightly inaccurate results. Due to the very small differences in strain involved, this could have an unwanted effect on the output data.

This could be avoided by having the software resample all the data sets to a much higher resolution before the cross-correlation process.

The intended flow of data through the software in the final design is summarised graphically in Figures 3.2 and 3.3.

3.4 Software Architecture

I architected this software to take full advantage of the object orientation capabilities provided by Python. Object orientation was particularly beneficial for this project due to the complex hierarchy of different types of data – the software simultaneously worked with everything from individual raw data files all the way to an aggregated collection of differences between averaged, temperature-compensated data sets, with several stages in between. Each type of data set or collection of data sets was represented by a class. The full list of classes involved in the data model, from the bottom upwards, is as follows.

StrainAnalyserDatafile – a set of strain data points imported directly from the text files output by the analyser or Yokogawa's utility, along with the corresponding loss data points (if provided) converted into a confidence value;

StrainAnalyserDataset – a collection of StrainAnalyserDatafiles representing repeat readings from a single fibre on a single day;

StrainAnalyserAveragedDataset – a single data set calculated from the weighted average of all StrainAnalyserDatafiles in a given StrainAnalyserDataset;

StrainAnalyserTemperatureCompensatedDataset – a strain data set (StrainAnalyserAveragedDataset) which has undergone temperature compensation using a second StrainAnalyserAveragedDataset from the temperature fibre;

StrainAnalyserDatasetCollection – a collection of StrainAnalyserTemperatureCompensatedDatasets representing several different days' readings for a single distributed strain sensor, one of which is marked as the reference data set;

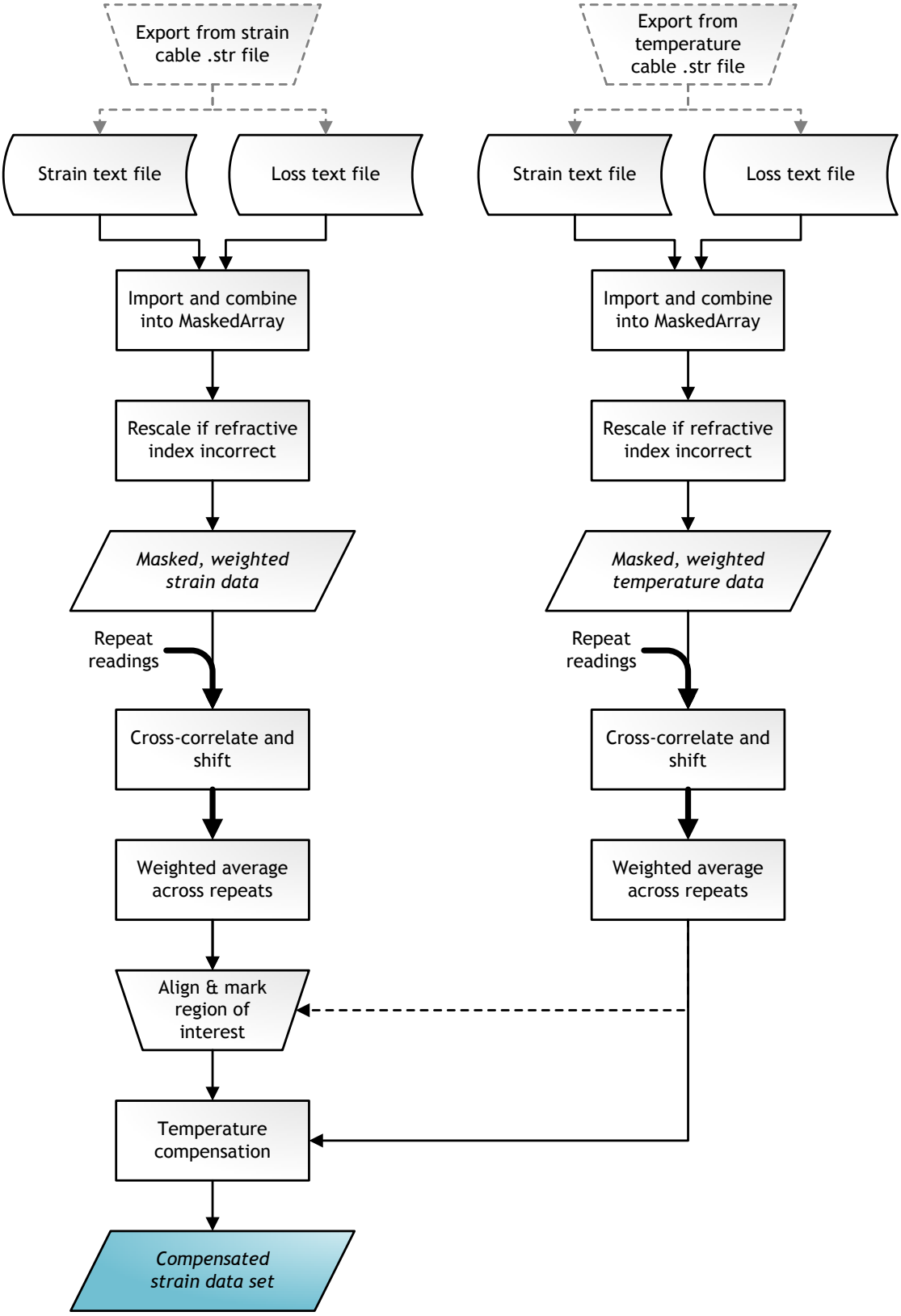

Figure 3.2: Flow of data (i): initial import to temperature compensation

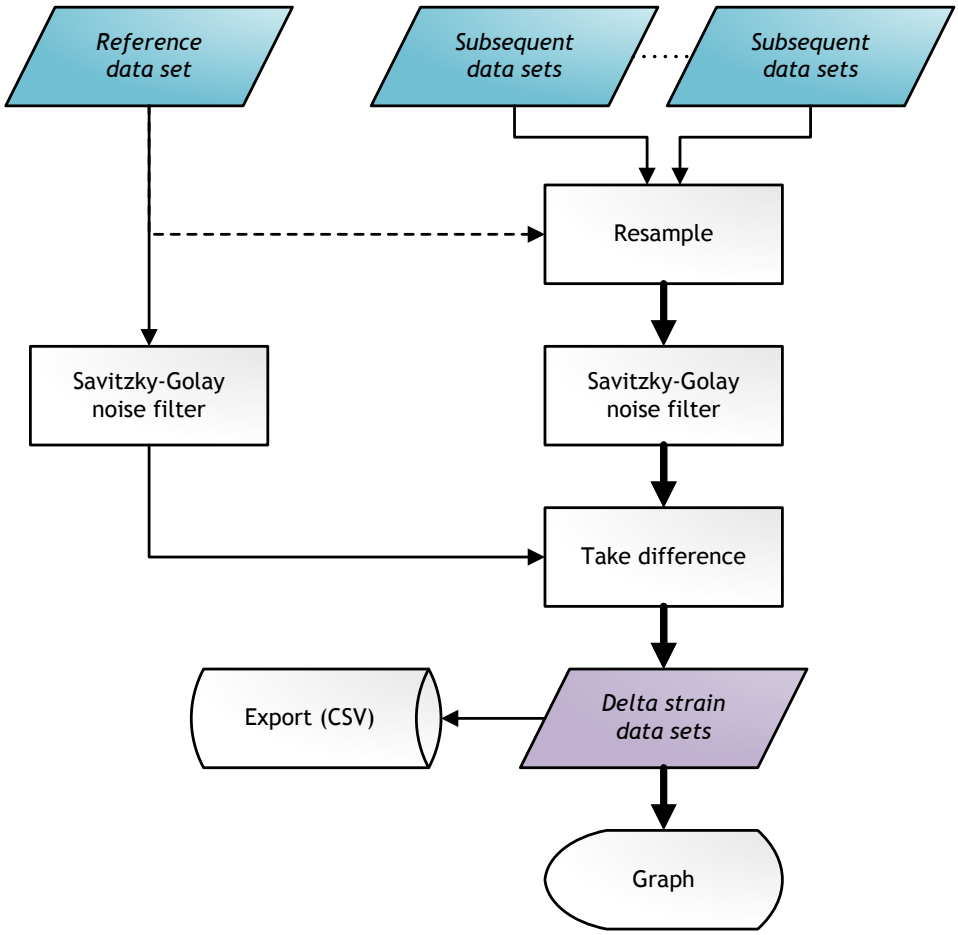

Figure 3.3: Flow of data (ii): aggregation and comparison of compensated data sets

StrainAnalyserDiffDataset – the difference between two StrainAnalyserTemperatureCompensatedDatasets (the reference data set and one subsequent data set), subtracted pairwise;

StrainAnalyserDiffDatasetCollection – a collection of StrainAnalyserDiffDatasets representing delta strain profiles compared with the reference.

This class hierarchy is shown in relation to the rest of the application in Figure 3.5.

I also aimed to keep the code for the graphical user interface as separate from and as loosely-bound to the underlying hierarchy of data structures as possible. Furthermore, within the graphical user interface, the graphing code was also kept in separate classes. This allowed changes to be made to an individual component or relationship, such as the data processing code, in relative isolation.

This class hierarchy follows the standard "Model-View-Controller" (MVC) architectural pattern in software engineering, originally proposed by Reenskaug [18] and il-

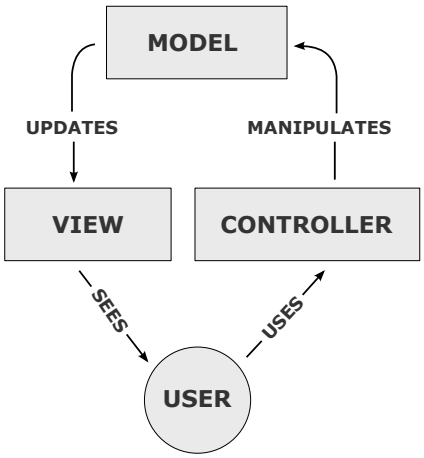

Figure 3.4: The Model-View-Controller software architectural pattern

lustrated in Figure 3.4; the "model" stores the data sets and associated state, the "view" handles presentation to the user and the "controller" handles updates of the model.

MVC applies both to the interaction between data model and user interface as a whole, and to smaller components of the software considered in isolation. For example, the GTK+ library uses its own form of MVC to connect lists in a graphical user interface to collections of data storage and processing objects: the model is named gtk.ListStore, and the view and controller rôles are both handled by the interface widget gtk.TreeView. I implemented my own classes to extend gtk.ListStore, since more complex behaviour than that provided by the library was required; my custom classes could still be connected up as the model for a gtk.TreeView. These connections can be considered a individual uses of MVC on the small scale, or as components of application-wide MVC.

Likewise, I implemented graphing classes (making use of the Matplotlib library) as individual views on various objects within the data model. This allowed live preview graphs to be implemented in order to assist the user. In a couple of cases the graphs also contained simple controllers where graph-based input was required, such as when manually aligning strain and temperature data.

3.5 Summary

The software has been developed in accordance with well-regarded software engineering practices and using a modern tool set, and operates as specified, reducing the task of processing BOTDR sensor data to a few simple steps for the user.

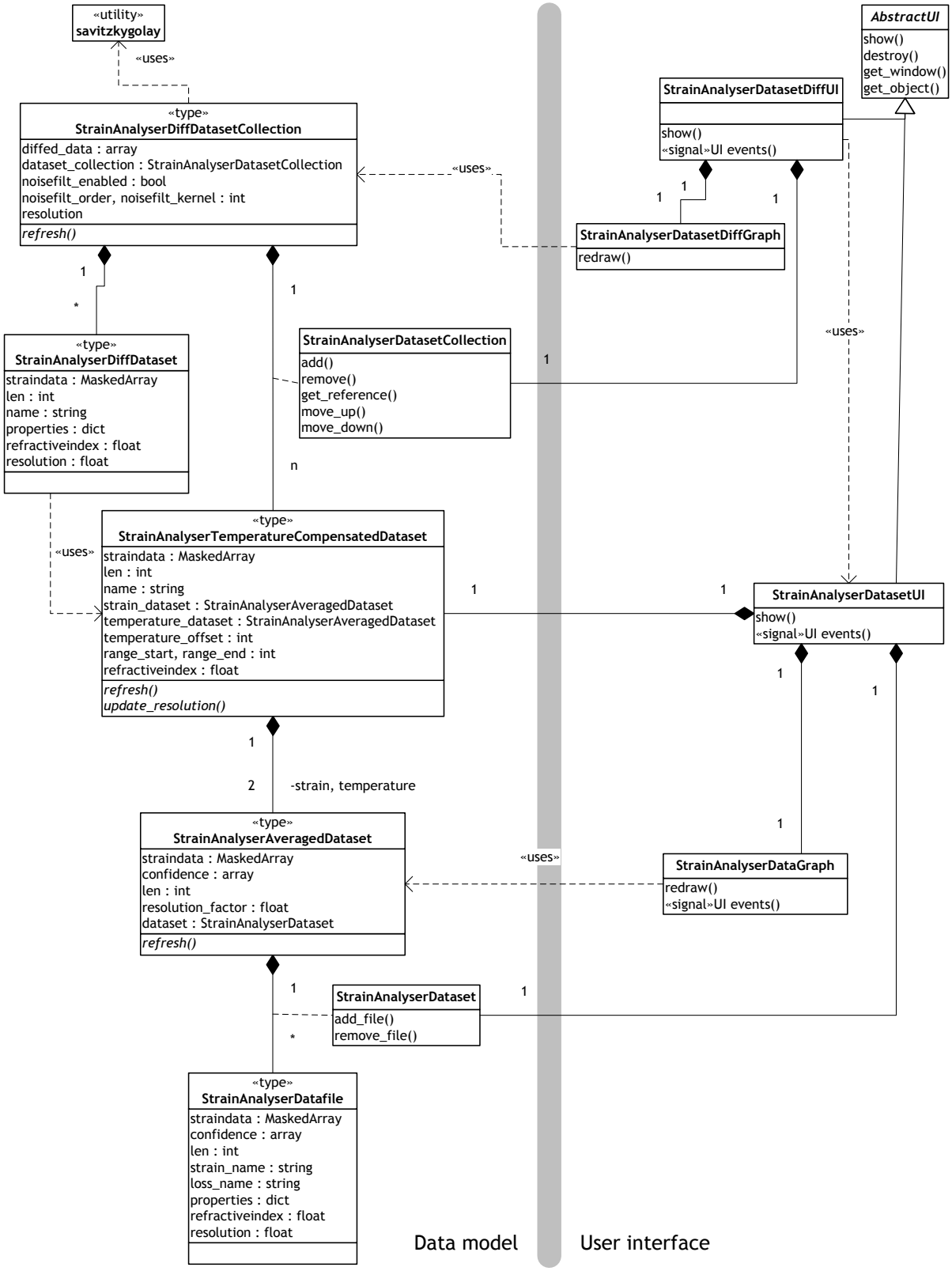

Figure 3.5: Unified Modelling Language (UML) object diagram

CHAPTER 4

Evaluation and Conclusion

This report has described the successful design and implementation of a novel software application to aid geotechnical researchers – and, ultimately, structural civil engineers – in the visualisation, processing and analysis of data from recently-developed photonic distributed strain sensing equipment. The software previously available as supplied by the sensor manufacturer Yokogawa was considered to be inadequate and has previously been supplemented with cumbersome spreadsheet-based analysis; the application I have developed goes significantly further than Yokogawa's software, and indeed further than some researchers' manual processes.

A full evaluation of the utility of this software to researchers in this field has not been possible given the time frame of this project. However, the software is intended to go into active use over the coming year, and some of the initial users were continuously involved throughout the specification and development process, providing invaluable feedback on each successive prototype, so it is very likely that the software meets their needs.

I believe the application is easy-to-use; it has been developed according to a modern set of user interface standards (via the Glade tool) and the "look and feel" of the software has been very well received. A screenshot is shown in Figure 4.1.

4.1 Future Work

Research into the use of "smart structures" with integral distributed strain sensors based upon Brillouin optical time-domain reflectometry is still very much active, and I expect deployment methods and analysis techniques to be adapted and improved over the coming years – indeed I hope that the software I have developed will be instrumen-

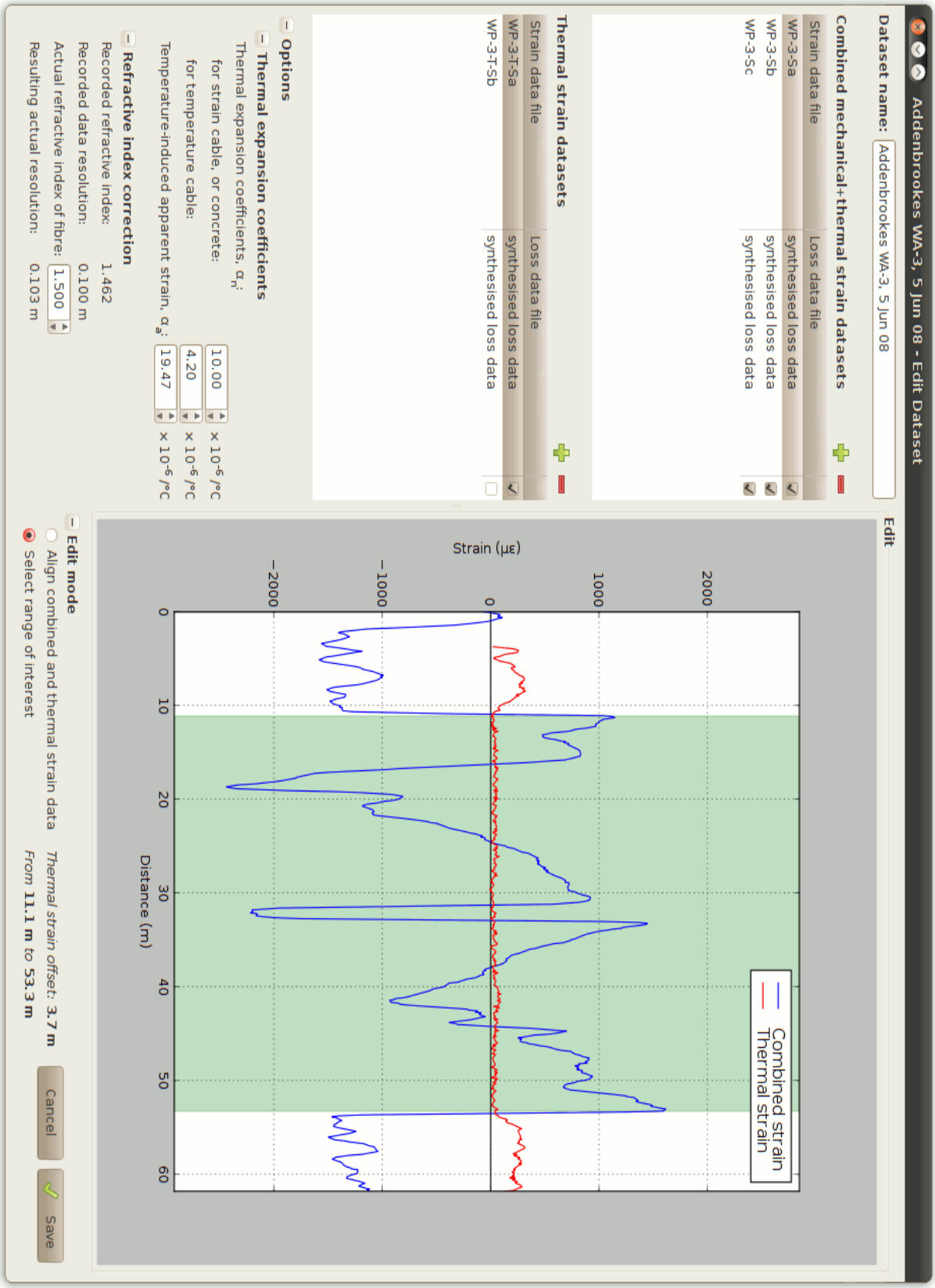

Figure 4.1: Screenshot of data set creation and editing interface

tal in the refinement of analytical techniques for distributed strain data. It is therefore likely that this software tool will have to be adapted and extended at some point to keep up with new methods and unanticipated scenarios. In particular, although every effort has been made to keep the application general-purpose, it has only undergone testing on the data from one deployment – the Addenbrooke's Access Road bridge – and other deployments already exist with significantly different characteristics, so the next step will likely be to investigate how well the software performs on other data sources.

I will be releasing this application under an open source license online, at:

`http://strainanalyser.malc.org.uk/`

I do not intend to abandon the project, and intend to work with Professor Soga's research group as the software enters use in order to make minor adjustments or perform bug fixes as needed, and of course the code will be available for others to use and adapt. The initial development effort would have been wasted if the code is subsequently left to rot, and I will not allow this to happen.

4.2 Acknowledgements

I am grateful to my supervisor Professor Kenichi Soga for his feedback throughout this project and for his support despite my lack of knowledge of civil engineering. His other students – Tina Schwamb, Echo Ouyang, Andy Leung and Te Janmonta – have also been extremely helpful throughout as well as being friendly and welcoming colleagues.

Hishan Mohamad, a previous student of Professor Soga, authored a thesis [2] which has been an invaluable and accessible reference for me during this project, and I am very appreciative of the care and attention which went into it.

Finally, I would like to thank all those involved in running the Doctoral Training Centre in Photonic Systems Development for giving me the opportunity to work in this field, and Dr. Cyril Renaud in particular for his tireless work behind the scenes to ensure that this first year of the new DTC was a success.

Bibliography

[1] K. Fujihashi, K. Kurihara, K. Hirayama, and S. Toyoda. Monitoring system based on optical fiber sensing technology for tunnel structures and other infrastructure. *Sensing Issues in Civil Structural Health Monitoring*, pages 185–195, 2005.

[2] H. Mohamad. *Distributed Optical Fibre Strain Sensing of Geotechnical Structures*. PhD thesis, University of Cambridge Department of Engineering, 2008.

[3] H. Ohno, H. Naruse, T. Kurashima, A. Nobiki, Y. Uchiyama, and Y. Kusakabe. Application of Brillouin scattering-based distributed optical fibre strain sensor to actual concrete piles. *IEICE Transactions on Electronics*, (E85):945–951, 2002.

[4] R. J. Mair. Tunnelling and geotechnics: new horizons. *Géotechnique*, 58(9): 695–736, 2008.

[5] P. J. Bennett, A. Klar, T. E. B. Vorster, C. K. Choy, H. Mohamad, K. Soga, R. J. Mair, P. D. Tester, and R. Fernie. Distributed optical fibre strain sensing in piles. In *Reuse of Foundations for Urban Sites: Proceedings of International Conference*, volume EP73, pages 105–114. IHS BRE Press, 2006.

[6] Road Traffic Technology. Addenbrooke's access road, Cambridgeshire, 2010. Online; retrieved 20 May 2010; available at `http://www.roadtraffic-technology.com/projects/addenbrookes/`.

[7] S. C. Jones, M. C. Robinson, and Y. M. Gupta. Ordinary refractive index of sapphire in uniaxial tension and compression along the c axis. *Journal of Applied Physics*, 93(2):1023 –1031, 2003.

[8] T. Horiguchi, T. Kurashima, and M. Tateda. Tensile strain dependence of Brillouin frequency shift in silica optical fiber. *IEEE Photonics Technology Letters*, (1):107–108, 1989.

[9] D. Cotter. Stimulated Brillouin scattering in monomode optical fiber. *Journal of Optical Communications*, (4):10–19, 1983.

[10] J. Smith, A. Brown, M. DeMerchant, and X. Bao. Simultaneous strain and temperature measurement using a Brillouin scatting based distributed sensor. In *SPIE Sensory Phenomena and Measurement Instrumentation for Smart Structures and Materials*, volume 3670, pages 366–372, 1999.

[11] T. R. Parker, M. Farhadiroushan, V. A. Handerek, and A. J. Rogers. A fully distributed simultaneous strain and temperature sensor using spontaneous Brillouin scatter. *IEEE Photonics Technology Letters*, 9:979–981, 1997.

[12] H. Mohamad and K. Soga. Thermal strain sensing of optical cables using Brillouin optical time domain reflectometry. To appear, 2010.

[13] R. D. Browne. Thermal movement in concrete. *Concrete*, 6:51–53, 1972.

[14] E. J. Sellevold and O. Bjontegaard. Coefficient of thermal expansion of cement paste and concrete: mechanisms of moisture interaction. *Materials and Structures*, 39:809–815, 2006.

[15] A. Savitzky and M. J. E. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry*, 36(8):1627–1639, 1964.

[16] The Right Tool. What programming tool is right for which job? – Python, 2010. Online tool by D. R. MacIver; retrieved 22 May 2010; available at `http://therighttool.hammerprinciple.com/languages/python`.

[17] B. Boehm. A spiral model of software development and enhancement. *SIGSOFT Software Engineering Notes*, 11(4):14–24, 1986. ISSN 0163-5948. doi: http://doi.acm.org/10.1145/12944.12948.

[18] T. Reenskaug. Thing-model-view-editor: an example from a planningsystem. Xerox PARC technical note, 1979.

Appendix A

Risk Assessment

Since this project was almost entirely computer-based, I was subjected to no unusual risks beyond those related to long-term computer use, which – having spent a considerable proportion of my working life using computers – I am used to mitigating against as a matter of habit.

The BOTDR analyser equipment has mechanisms in place to prevent accidental escape of laser light so was considered safe when used by someone familiar with the safety procedures. At no point did I use the analyser myself.

One site visit to the Addenbrooke's Access Road bridge was undertaken early during the project. At this point the bridge had already been completed and no construction work was ongoing on the site; all construction site hazards had been removed prior to my visit. However access to the site was still under the control of the contractors and so safety clothing and a hard hat was required and worn. The only risk I was subjected to was discomfort due to the poorly-fitting safety footwear with which I was provided.