Unravelling the Activity of a Novel Plant Growth Modulator

DANIEL JAMES BRUCE

DURHAM UNIVERSITY

DOCTORAL THESIS

Unravelling the Activity of a Novel Plant Growth Modulator

Author:
Daniel BRUCE

Supervisor:
Dr. Ehmke POHL

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in the

Pohl Group
Department of Biosciences

"You live and learn. At any rate, you live."

Douglas Adams

Abstract

The coordination of plant growth is achieved through a range of hormonal signalling processes that are of great academic and commercial interest. The first half of this project explores the mechanism of eW5, a novel naphthalenesulfonamide synthetic modulator of plant growth, through a series of biophysical and genetic studies. The growth-enhancing effect of eW5 in *Arabidopsis thaliana* and its structural similarity with pyrabactin fed the hypothesis that eW5 interacted with proteins in the PYR1-like (PYL) family, which regulate abscisic acid signalling. Thermal shift assay, surface plasmon resonance and microscale thermophoresis experiments with ew5 and PYL proteins provided evidence of weak interactions between eW5 and PYR1, PYL1, PYL2 and PYL5. A forward genetic study was performed with EMS-mutagenised *A. thaliana* to identify genes responsible for a seedling hypocotyl elongation response to eW5 that identified nine genes of interest linked to plant development.

The second half of the thesis focuses on method development for techniques useful both within the project and beyond. Hypocotyl elongation assays often involve a significant time investment to accurately measure hypocotyl lengths. A data processing pipeline was devised to automate data analysis. An R-CNN was trained to identify regions containing hypocotyls in MS-agar Petri dish images of *A. thaliana* seedlings. The resulting model demonstrated poor accuracy in determining the position of hypocotyls in crowded images.

Finally, contributions toward a novel algorithm for BioSAXS data analysis were made. Geometric parameters, including curvature, torsion and contacting residues were analysed for each protein structure in a survey of high-resolution crystallographic data from the Protein Data Bank to provide a distribution of probable curve geometries during model refinement. A web server for data submission and automatic preprocessing was created to simplify the process of utilising the algorithm for inexperienced users.

Acknowledgements

This project would not have been possible without help from my brilliant family, friends and colleagues.

Thanks to Ehmke Pohl, Marc Knight and Patrick Steel for keeping me right in all aspects of the project. Your knowledge and enthusiasm fuelled so much of this work; thank you for your support throughout and thank you for your patience during an exceptionally protracted write-up process. Thank you to the BBSRC, whose funding made this project possible

Thanks to Fieka Sukiran, whose work laid the foundation of my PhD and whose advice was invaluable and infallible in the early days of it. Thanks to everyone in CG001 for reminding me how to do synthetic chemistry. Particular thanks must go to Jonathan Reuven, without whom I'm pretty sure I would have given up hope of ever completing the work for chapter 4. Thanks to Ted Farmer for providing the seeds used in Chapter 4. Thanks to Wenbin Wei, whose analysis of the NGS data explored in Chapter 4 saved so many headaches. Thanks to everyone in Lab 19 for putting up with us and our plants while providing excellent advice throughout the *Arabidopsis* work. Chris Prior and Arron Bale remain oracles of all things mathematical, thanks very much for your help with Chapter 6. Thanks to Pete Stokes and Dave Parker in mass spec along with Alan Kenwright and the NMR team for their analytical services. Thanks to Alex Eagan and Newcastle University for allowing use of their MST facility. Thanks to Toby Breckon for an illuminating conversation that ignited the work that became Chapter 5. Thanks to everyone in CG231 (and Joy!) for keeping me suitably distracted with tea breaks, crosswords and brilliant advice.

Mum, Dad, Megan and Ellie, thank you for your wonderful support over the years and for feigning interest very convincingly while I bore you to death talking about proteins. Ness, you've been a rock throughout hugely difficult times and I couldn't think of anybody I'd rather share a PhD journey with.

Contents

List of Figures

List of Tables

Chapter 1

Thesis Introduction

1.1 Chemical Modulators of Plant Growth

Plants rely on a complex network of regulatory responses to thrive in nature. From the moment a seed touches down on the substrate it is destined to grow on, it must work in harmony with its environment to enact the processes necessary for its survival, not just in the right order for its own development but at the correct time for the seasonal, weather and habitat changes it relies on. Far from the classical opinion of plants as stationary and unchanging organisms, within each plant there exists a dynamic and powerful array of signalling responses which allow plants to correctly respond to the 'when', 'what' and 'how' questions of survival; a chemical intelligence that maintains the plant kingdom's position as the dominant biomass on planet earth.

The ability to harness agrochemicals represents one of the greatest technological leaps forward in agricultural history. Chemical agents, from pesticides to fertilisers, underpin vast reserves of the world's food supply and allow the production of crops in a way that is cheaper, better-yielding and more consistent than has ever been historically possible. Continued innovation within the agrochemical space is vital to support a growing global population by allowing crops to be produced on previously untenable land and in greater quantities.

This thesis explores the activity of eW5, a synthetic ligand that has been shown to enhance radicle and hypocotyl growth in *Arabidopsis* samples [1]. This introduction provides a brief overview of industrially-important modulators of plant growth and also gives a general introduction to techniques that can be used in the search for novel agrochemical agents.

1.1.1 Commercial Uses of Plant Growth Modulators

On the largest scale, the agrochemicals used to enhance the growth of plants (and subsequent crop yield) commercially are those used in more traditional fertilisers, such as ammonium nitrate, which provide plants with an abundance of nutrients and allow crops to be produced in poorer soils. However, a second class of plant growth enhancers modulate plant growth by interacting with some of the signalling pathways that control the metabolic and physiological processes that plants use to adapt to their environment. The introduction to Chapter 4 provides individual introductions to important growth-mediating phytohormones. While chemical fertilisers are a key player in modern agriculture, they are not the focus of this section.

One of the most prominent uses of phytohormone-analogue plant growth enhancers is in seed priming. Seed priming is a pre-germinative technique that involves exposing seeds to conditions that help synchronise the germination time and speed of a whole crop. Traditional, or 'on-farm', seed priming can be as simple as soaking seeds in water for a time that is generally optimised by species, then drying them back to their original moisture content before sowing. This can be particularly useful in dry-land regions and works by increasing the rate of early germination, allowing seedlings to quickly develop roots and shoots to take advantage of difficult-to-access water and light. In reducing the time spent during this early phase of plant development, the risk of new plants succumbing to weeds, pests and diseases in one of their most vulnerable developmental stages is also reduced. Simple, water-only seed priming can dramatically increase the survivability and overall yield of vulnerable harvests in resource-poor regions, with some farmers reporting as much as a 206 % increase in crop yield that is consistently observed after switching to on-farm priming.[2]

Hormonal priming increases the complexity of the priming process by exposing the seeds to phytohormones or their analogues. Phytohormones are generally delivered as an aerated solute medium. Several studies show the efficacy of hormonal priming on different crop species, with the priming process generally providing increased growth, root biomass and overall drought-tolerance. A range of different phytohormones are usable within hormonal priming treatments, including gibberellic acid (GA), benzyladenine (BA) and even abscisic acid (ABA).[3–5]

It is, perhaps, counterintuitive to also consider the uses of general inhibitors of plant growth in an agricultural setting. Despite the importance of crop yields, tactical use of compounds which interact with growth signalling pathways in an often inhibitory manner can also be useful in a variety of settings. A major industrial use of ABA and its analogues is in fruit production, where exogenous ABA can be used to influence ethylene release and accelerate the ripening process of commercially-important fruits.[6]

With an ever-increasing global population and environmental instability all but certain in the future, the ability to influence the growth of crops in a predictable manner is an important field of study. The discovery of new compounds that can activate and disrupt phytohormone signalling pathways is critical to the long-term sustainability of population growth.

1.2 Pathways to Novel Agrochemicals

There is an interesting parallel between research into new agrochemicals and new medicines. They share a surprisingly large number of experimental workflows, they feed into two of the most significant modern industries and, despite the many differences between an operating theatre and a farmer's field, both underpin technologies that are vital for human life. They also share many similar problems; issues such as side-effects, be they unintended consequences of medication or environmental disasters caused by reckless agrochemical use. The need for powerful and targeted novel agrochemicals has given rise to a triumvirate of experimental areas over time that range from traditional whole-organism studies to powerful *in silico* simulations, all of which drive experimental progress. This thesis explores a few of these techniques (the list below is by no means exhaustive) and aims to show how lab-based experimentation can be augmented with novel computational efforts.

1.2.1 *In Vitro* Approaches

A combination of biochemical and phenotypic approaches can be useful in elucidating a molecular mode of action and any physiological changes associated. Biochemical techniques can be useful when a target protein is established and can be used to investigate binding interactions. In the case of this project, multiple PYL proteins were overexpressed and any potential binding interactions were probed using the biophysical techniques discussed below.

Molecular biology techniques allow for the heterologous expression of target proteins at widely affordable prices. The use of recombinant proteins can provide an experimenter with the opportunity to study specific interactions between target molecules and a protein of interest using well-established biophysical techniques. While this is generally the most preferable approach to quantifying the affinities of protein-ligand interactions, it is important to remember the crosstalk that phytohormonal signalling pathways display when considering the targets of molecules of interest in an isolated setting.

Biophysical Techniques

Chapter 3 explores the relationship between eW5 and a family of ABA-binding proteins, the PYL family, through a variety of biophysical approaches. A basic introduction to each technique is also provided within the chapter.

Thermal Shift Assays The Thermal Shift Assay (TSA) is an established technique for studying protein-ligand interactions. Protein-ligand complexes typically have a higher degree of stabilising internal interactions compared to free protein. This usually translates to an increase in the thermal denaturation temperature of a protein with increasing ligand concentration.[7–9]

Surface Plasmon Resonance Surface Plasmon Resonance (SPR) is a popular technique that allows the investigation of protein-protein and protein-ligand interactions in an unlabelled form. SPR measures changes in refractive index at the surface of a metal sensor chip. Changes in the environment or conformation of a protein molecule bound to the sensor chip, such as those caused by multimer association or by small-molecule ligand binding, can cause detectable changes in refractive indices and provide an SPR signal that allows quantifiable measurement of interaction affinity and, with the correct setup, binding kinetics.[10]

Microscale Thermophoresis Microscale Thermophoresis (MST) provides insights regarding a molecule's size, charge and hydration shell. It relies on thermophoretic movement, the directed motion of molecules along a thermal gradient. A molecules thermophoretic affinity is affected by its size as well as its affinity to its surrounding medium, meaning that protein-protein interactions or ligand binding events can be analysed through this technique. Protein molecules studied through MST require fluorescent labelling, which can be achieved through a variety of readily-available commercial kits for ease.[11, 12]

1.2.2 *In Silico* Approaches

Modelling Approaches

For rational molecule design approaches, acquiring a high-quality structural model of a target protein (or, at least, a target binding domain) is often seen as a the most important first step of a project. For most proteins, this would historically involve acquiring a high-resolution data set from x-ray crystallography. While this is still generally the most desirable starting point in a structural biology project, crystallography requires significant efforts to optimise crystallisation conditions, and for the protein to be crystallisable in the first place. Alternative structural techniques such as biological small angle x-ray scattering (BioSAXS), Cryo-EM and NMR are also available to provide protein models. Chapter 6 explores a novel technique for analysing BioSAXS data. Each of these techniques requires a degree of computation to build a suitable model to match experimental data. The optimisation process of model building benefits from having a physically-realistic starting point that is as close to the true structure as possible. Structural biologists have benefited from a range of *in silico* approaches to protein structure prediction that can be used as part of modelling workflows.

AI in Bioscience

Artificial intelligence represents one of the largest paradigm shifts in modern science and technology. It has permeated through almost every academic and industrial field; in its various guises it can be used in tasks as diverse as translating natural language to Python code, summarising lengthy texts into synopses and even predicting the structure of a protein from its primary sequence. While general-use AI still feels many years or decades away, AI continues to evolve on two fronts: increasing power as models and computing power improves and increasing utility as the widespread adoption of AI technologies allow for novel and creative use cases. The work undertaken for this project was completed during a time that watched AI progress from a decreasingly niche academic interest, with strong industry players such as DeepMind ready to take centre stage with projects such as AlphaGo, all the way to a mainstay of daily life. Biology has not been isolated from this widespread adoption of AI and has seen several important use cases develop and ingrain themselves within the field.

Alphafold[13] is the most widely adopted AI-based protein structure prediction program at the time of writing. Along with RosettaFold[14], its model is constructed around neural networks, a deep-learning architecture that is loosely based on the setup of human neural networks. Chapter 5 provides an short introduction to neural networks and describes some of the tasks they can accomplish in the field of image analysis. After winning CASP 2020, the team behind Alphafold have gone from strength to strength, improving the program across several releases with the most recent being Alphafold 3[15]. The widespread adoption of AI-generated modelling techniques has been expedited by their low barrier of access. The AlphaFold Protein Structure Database[16] provides over 200 million freely-available, pregenerated models and their associated confidence metrics. These models have shown good accuracy with a range of protein families when used to generate hypothetical Bio-SAXS curves and compared with experimentally-obtained results using an ensemble-based technique.[17]

The ability to access increasingly accurate protein structure predictions is a useful tool in the practising structural biologist's toolkit. However, there are limitations to the conclusions that can be drawn from predicted models alone. Machine learning techniques, however complex the architecture of their neural networks, ultimately rely on an established dataset to guide the training of their predictive models and measure their resulting quality. In the field of structural biology, there is therefore an unavoidable bias to the training dataset: models can only be reliably trained on proteins with experimentally-obtainable structures of good quality. There is therefore, perhaps, a danger worth bearing in mind that AI-driven models could confidently provide a model that is, while physically sensible, incorrect. This can be especially true in protein families and motifs that are underrepresented in experimental databases such as the Protein Data Bank[18] or that possess intrinsically disordered regions.

Despite these dangers, the rise of AI as a tool within structural biology has undoubtedly been a success and has opened up the possibility of modelling and making predictions about proteins that are not amenable to current experimental techniques. AI-generated modelling has generally replaced homology modelling approaches in the generation of starting models that are to be optimised in experimental workflows (for example, in crystallographic molecular replacement). At present, AI represents a powerful complement to experimental work in structural biology. Whether this balance will continue into the future is an interesting question, but the 'human-in-the-loop' is still of vital importance today within structural biology, especially so in the case of unique and novel protein morphologies.

1.2.3 *In Vivo* Approaches

Chapter 4 explores eW5 activity in *Arabidopsis thaliana* from a whole-organism perspective, combining phenotypic response experiments with next-generation DNA sequencing.

Phenotypic screening is a useful way to investigate the overall effect a molecule has on an organism. In contrast to traditional drug-discovery, whole-organism phenotypic screening is more readily accessible in plant sciences due to the lack of necessary ethical considerations. As such, this technique can be used to quickly and widely assess the impact of a library of compounds on the growth of a plant to efficiently pinpoint lead compounds. In the case of this project, iterative phenotypic screening across a mutant population was used to identify genes associated with the mode of action of eW5.

One downside of plant-based phenotypic screening and subsequent genetic studies is the amount of time-consuming labwork and data analysis that it entails for the experimenter. Chapter 5 explores the use of machine learning image analysis techniques to supplement the manual work involved in phenotypic profiling.

Chapter 2

Experimental Chapter

2.1 Biophysical Studies of PYL Family Proteins with eW5

2.1.1 Ligand Synthesis and Characterisation

All reagents used for chemical synthesis were purchased from Merck (Sigma Aldrich). NMR spectra were recorded with either a Varian VNMRS-600 instrument with operating frequencies of 600.130 MHz for ^{1}H and 150.903 MHz for ^{13}C NMR and Varian VNMRS-700 with operating frequencies of 700.130 MHz for ^{13}C and 176.048 MHz for ^{1}H NMR. NMR spectra were referenced relative to $CDCl_3$ (δ_H 7.26 ppm, δ_C 77.16 ppm), DMSO-d_6 (δ_H 2.50 ppm, δ_C 39.52 ppm). Chemical shifts are reported in parts per million (ppm), coupling constants (J) in hertz (Hz) and multiplicity as singlet (s), doublet (d), triplet (t), quartet (q), multiplet (m) or a combination thereof.

Infrared spectra were collected using a Perkin Elmer Frontier FTIR with Universal ATR Sampling Accessory with assignments reported in wavenumbers (cm^{-1}).

Melting point analyses were performed with Thermo Scientific Electrothermal IA9100 Digital Melting Point apparatus.

Thin layer chromatography was performed using Merck F254 silica gel 60 aluminium sheets pre-coated with silica gel.

Liquid chromatography mass spectrometry (LCMS) was performed on a Waters TQD mass spectrometer ESI-LC water (0.1 % formic acid): MeCN, flow rate 0.6 ml min^{-1} with a UPLC BEH C18 1.7 µm (2.1 mm x 50 mm) column. Gas chromatography mass spectrometry (GCMS) was carried out on a Shimadzu QP2010-Ultra with a temperature gradient 50 °C – 300 °C and a hold time of 5 min, using a Rxi-17Sil MS (0.15 µm x 10 m x 0.15 mm) column.

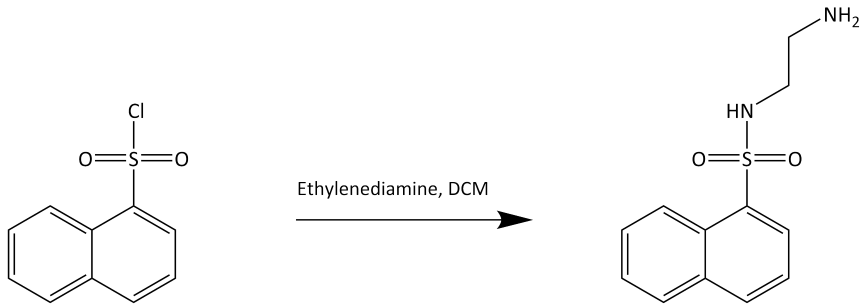

FIGURE 2.1: eW5 synthetic route.

eW5 Synthesis

Napthalene-1-sulfonyl chloride (0.51 g, 2.2 mmol) was dissolved in 25 ml of dry dichloromethane. Ethylenediamine (3.0 ml, 44.8 mmol, 20 equivalents) was added dropwise, then the reaction mixture was stirred at room temperature for 1 h. After TLC analysis (3:2 hexane:ethyl acetate) suggested complete consumption of the starting material, 10 ml of water was added to the reaction mixture and the product was extracted in dichloromethane (3 x 10 ml). After removal of dichloromethane *in vacuo*, the crude product was dissolved in 15 ml of methanol then filtered. The pH of the mother liquor was adjusted pH 2 with concentrated HCl and then the solvent was removed *in vacuo*. following recrystallisation from methanol-ether, the title compound was obtained as white crystals (0.45 g, 1.5 mmol, 71 %) melting point 176-177 °C. ν max (ATR) 3405(N–H [salt]). δH (700 MHz, Chloroform-d) 8.68 – 8.61 (m, 1H, Ar-H), 8.26 (dd, J = 7.3, 1.3 Hz, 1H, Ar-H), 8.06 (d, J = 8.2 Hz, 1H, Ar-H), 7.96 – 7.89 (m, 1H, Ar-H), 7.67 (ddd, J = 8.5, 6.9, 1.4 Hz, 1H, Ar-H), 7.60 (ddd, J = 8.1, 6.9, 1.1 Hz, 1H, Ar-H), 7.53 (dd, J = 8.2, 7.3 Hz, 1H, Ar-H), 2.90 (t, J = 7.0, 2H, CH_2), 2.68 (t, J = 7.0 Hz, 2H, CH_2). δC (151 MHz, Chloroform-d) 134.5(Ar-C), 134.3(Ar-C), 134.2(Ar-C), 129.7(Ar-C), 129.1(Ar-C), 128.4(Ar-C), 128.19(Ar-C), 126.9(Ar-C), 124.3(Ar-C), 124.1(Ar-C), 45.4(CH_2), 40.7(CH_2). m/z (LCMS ES^+) rt 1.65; m/z 251.94 $[M+H]^+$.

DB1 Synthesis

Napthalene-1-sulfonyl chloride (0.25 g, 1.1 mmol) was dissolved in 10 ml of dry dichloromethane. Propylamine (1.8 ml, 20.0 mmol, 20 equivalents) was added dropwise, then the reaction mixture was stirred at room temperature for 20 h. After GCMS analysis suggested complete consumption of the starting material, 10 ml

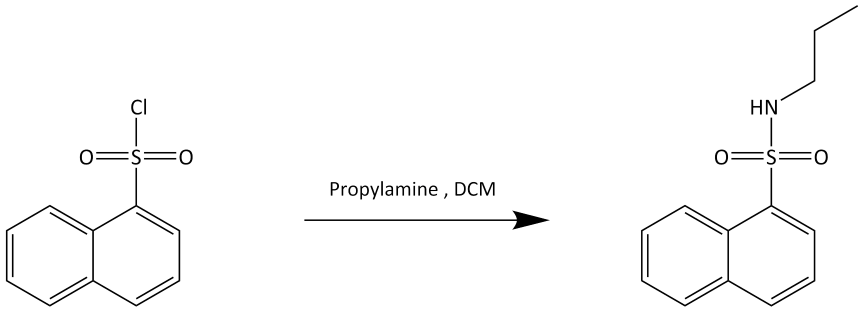

FIGURE 2.2: DB1 synthetic route.

of water was added to the reaction mixture and the product was extracted in 12 dichloromethane (3 x 10 ml). After the removal of dichloromethane in vacuo, the product was placed under high-vacuum for 24 h and isolated as a pale-yellow solid without further purification (0.25 g, 1.0 mmol, 91 %) melting point 68-69 °C. ν max (ATR) 3295(N–H). δH (400 MHz, Chloroform-d) 8.67 (dq, J = 8.6, 0.9 Hz, 1H), 8.29 (dd, J = 7.3, 1.3 Hz, 1H), 8.10 (dd, J = 8.3, 1.2 Hz, 1H), 8.02 – 7.95 (m, 1H), 7.75 – 7.53 (m, 3H), 4.56 (s, 1H), 2.89 (td, J = 7.1, 6.2 Hz, 2H), 1.43 (h, J = 7.3 Hz, 2H), 0.79 (t, J = 7.4 Hz, 3H). δC (176 MHz, Chloroform-d) 134.67(Ar-C), 134.26(Ar-C), 134.16(Ar-C), 129.62(Ar-C), 129.10(Ar-C), 128.30(Ar-C), 128.16(Ar-C), 126.81(Ar-C), 124.32(Ar-C), 124.13(Ar-C), 45.04(CH_2), 22.92(CH_2), 10.97(CH_3). m/z (GCMS ES^+) rt 6.29; m/z 249.10 $[M+H]^+$.

2.1.2 Protein Expression

PYL Family Expression

The transgenes for PYL1-PYL10 were provided by the Marc Knight Laboratory in pET-28 a (+) plasmids conferring kanamycin resistance and introducing a hexahistidine (His_6) tag. Plasmids were extracted from previously transformed DH5α *Escherichia coli* provided by a colleague using an Illustra plasmidPrep Mini Spin Kit (GE Healthcare) and its standard accompanying protocol. *E. coli* strain BL21 (DE3) cells (New England Biolabs) were transformed through heat shock with the extracted plasmids. Colonies were grown overnight on agar inoculated with 50 µg ml^{-1} kanamycin. Plasmids subsequently isolated from the positive colonies were verified by DNA sequencing before protein expression was performed.

Overnight cultures (25 ml) of transformed BL21 (DE3) cells in LB media (Melford Biolaboratories) containing 50 µg ml^{-1} kanamycin were inoculated into 1 l LB media containing 50 µg ml^{-1} kanamycin and incubated at 37 °C. When the OD_{600} of the cell cultures reached 0.6-0.8, expression was induced via the addition of isopropyl β-D-1-thiogalactopyranoside (IPTG) to a final concentration of 100 µM. Cells were then grown for an additional 20 h at 20 °C and harvested by centrifugation for 30 min at 4000 x *g* (Beckman Coulter Avanti J-20 XP). Cell pellets that were not immediately used for protein purification were stored at -80 °C.

PYR1 Expression

Initial experiments with PYR1 used a construct provided by the Marc Knight Laboratory that was tagged with both an N-terminal Strep-II tag and a C-terminal His_6 tag to provide multiple options for purification.[1] The transgene was inserted into a pET-24 a (+) plasmid conferring kanamycin resistance. Plasmid isolation, expression line transformation, protein expression and cell lysis were performed as described for PYR1.

PYR1 Purification

Cell pellets were resuspended in Buffer A (50 mM NaH_2PO_4, 300 mM NaCl, pH 8.0) containing 10 mM imidazole and sonicated (Bandelin SONOPLUS 2450) on ice five times for 30 s with 30 s resting intervals. The lysate was then centrifuged at 48,000 x *g* for 50 min at 4 °C and the resulting supernatant was decanted and passed through a 22 µm filter membrane.

Purification of the lysate was performed using an ÄKTA™ Pure FPLC System (GE Healthcare). A 5 ml HisTrap™ Fast Flow Column (GE Healthcare) was equilibrated with Buffer A containing 30 mM imidazole before the filtered lysate was applied to it. The column was then washed with 10 column volumes of Buffer A containing 30 mM imidazole. Bound protein was then eluted over 10 column volumes with Buffer A containing 500 mM imidazole using a 0-100 % gradient elution programmed with UNICORN™ software (GE Healthcare).

Component	Resolving / µl	Stacking / µl
Buffer	2400.0	1200.0
40 % BisAcrylamide	1875.0	712.5
MilliQ H_2O	1575.0	1487.0
10 % (w/v) APS	62.0	31.3
TEMED	6.3	3.2

TABLE 2.1: SDS-PAGE gel recipe used throughout project

Eluted protein was then dialysed overnight at 4 °C into a buffer appropriate for downstream use (see corresponding subsections for each buffer composition) and stored at 4 °C or flash frozen in liquid nitrogen and stored at -80 °C.

SDS-PAGE Analysis

Protein samples were routinely analysed for purity using SDS-PAGE. Gels were generally made to 1 mm thickness with a final acrylamide concentration of 12.5 %. Protein samples were loaded with Invitrogen™ NuPAGE™ LDS Sample Buffer (4X) (Fisher Scientific) supplemented with 5 % β-mercaptoethanol (Fluorochem) to 1X NuPAGE™ LDS after heating the combined sample at 96 °C for approximately 4 min. SDS-PAGE experiments were run in a buffer containing 25 mM Tris, 250 mM glycine, 0.1 % (w/v) SDS, pH 8.3. 4X Resolving and 4X stacking buffers were obtained from Alfa Aesar, acrylamide:bis-acrylamide 29:1, 40 % solution was purchased from Fisher Bioreagents. Electrophoresis grade TEMED was purchased from MP Biomedicals and ammonium persulfate was purchased from Merck. SDS-PAGE gels were stained using Quick Coomassie Stain (Generon) overnight before destaining with ultrapure water and imaging using Bio-rad Molecular Imager Gel Doc XR+ with Image Lab (version 6.0.1) software.

ESI Mass Spectrometry

Proteins were prepared for mass spectrometry analysis by being dialysed into a 50 mM ammonium acetate buffer (pH 5.6) and concentrated to 1 $mg\,ml^{-1}$. An ESI-MS spectrum was obtained using a QToF Premier Mass Spectrometer (Waters Ltd) detecting positive ions. MS method: 9min Desalt pos V LM 800 2000 CV35. Protein data were processed using Masslynx 4.1 and deconvoluted using MaxEnt 1 to show the nominal neutral mass of the protein.

Mass Spectrometry Following Tryptic Digest

Following SDS-PAGE of HisTrap™-purified protein, bands corresponding to the expected mass of the protein were extracted and digested using trypsin and chymotrypsin. Digested fragments were compared against a database of *A. thaliana* and *E. coli* proteins to confirm protein identity.

2.1.3 Thermal Shift Assays

Protein Stability Screening Preparation

The Durham Salt Screen, The Durham pH Screen and The Durham Osmolyte Screen were purchased from Molecular Dimensions. 5000 X SYPRO® Orange (ThermoFisher) was added to 1 mg ml^{-1} PYR1 (30 mM Na_2PO_4, pH 7.2) to a final concentration of 4 X SYPRO® Orange. 10 µl of the resulting solution was added to each well of a 96 well plate containing 10 µl of the corresponding condition from one the Durham Screens. The plate was then sealed with StarSeal Advanced Polyolefin Film (StarLab) and centrifuged at 400 x *g* for 2 min at 20 °C.

Ligand Screening Preparation

5000 X SYPRO® Orange (ThermoFisher) was added to 1 mg ml^{-1} PYL-family protein (50 mM Na_2PO_4, 300 mM NaCl, pH 7.2) to a concentration of 4 X SYPRO® Orange. 100 mM stock solutions of all ligands were prepared in DMSO. Serial dilutions of the compounds in DMSO were created, then wells of a 96 well plate were filled with both solutions and diluted with additional buffer such that the final concentration of: PYR1 was 0.5 mg ml^{-1}, SYPRO® Orange was 2 X and DMSO was 5 % with a final sample volume of 20 µl per well. 96 well plates were then sealed with StarSeal Advanced Polyolefin Film (StarLab) and centrifuged at 400 x *g* for 2 min at 20 °C. Following centrifugation, the plates were incubated at room temperature for 30 min before the thermal shift assay (TSA) experiment commenced.

Performing a Thermal Shift Assay Experiment

TSA experiments were performed using either the 7500 Fast RT-PCR System (Applied Biosystems) or the CFX Connect (Bio-Rad). Fluorescence data was collected between 25-95 °C with a constant temperature gradient of 1 °C min^{-1}. After conversion into a suitable format for analysis, data from filter C of the 7500 Fast RT-PCR System (excitation 550 ± 11 nm, emission 586 ± 11 nm) or the FRET channel of the CFX Connect were analysed using NAMI software.[8, 9]

2.1.4 Surface Plasmon Resonance

Surface Plasmon Resonance (SPR) experiments were performed using a Biacore X100 (GE Healthcare) on either Biacore Sensor Chip CM5 or Biacore Sensor Chip NTA (GE Healthcare). Unless otherwise specified, Sensor Chip NTA experiments were run in HBS-P buffer (10 mM HEPES, 150 mM NaCl, 0.005 % (v/v) Surfactant P20, pH 7.4 with 3 mM EDTA) and Sensor Chip CM5 experiments were run in HBS-EP buffer (10 mM HEPES, 150 mM NaCl, 3 mM EDTA, 0.005 % (v/v) Surfactant P20, pH 7.4). Experiments were performed using "Kinetics/ Affinity" workflow protocols programmed with Biacore X100 Control Software (GE Healthcare).

Sensor Chip NTA

Protein Capture Optimisation Before each protein capture step, the Sensor Chip NTA surface was preconditioned with 0.5 mM $NiCl_2$. Protein exposure times were varied from 60-180 s and protein concentration was varied between 5 - 50 µg ml^{-1}.

Affinity Measurements Solutions of eW5 from DMSO stocks were made in HBS-P buffer (final DMSO concentration in samples 1 % (v/v)) with concentrations ranging from 20 nM - 1620 nM. Captured PYR1 was exposed to each solution with a contact time of 60 s and a dissociation time of 600 s. Regeneration between each exposure was performed with 350 mM EDTA. Before each protein capture step, the surface was preconditioned with 0.5 mM $NiCl_2$.

Sensor Chip CM5

pH Scouting 200 µg ml^{-1} solutions of PYR1 were prepared (10 mM acetate; pH 4.0. 4.5, 5.0, 5.5 and 6.0). The SPR response produced by each solution was measured over a contact time of 60 s and the chip surface was regenerated using 50 mM NaOH between each cycle.

Amine Coupling The Sensor Chip CM5 Surface was activated using 100 mM N-hydroxysuccinimide (NHS) and 400 mM 1-ethyl-3-(3-dimethylaminopropyl) carbodiimide hydrochloride (EDC). 200 µg ml^{-1} PYR1 (10 mM acetate, pH 5.5) was then immobilised on the activated surface. The CM5 surface was then washed with 1 M ethanolamine to deactivate any unreacted NHS-esters.

Affinity Measurements Solutions of ABA and eW5 from DMSO stocks were made in HBS-EP buffer (final DMSO concentration in samples 1 % (v/v)) with concentrations ranging from 20 nM - 1620 nM. Immobilised PYR1 was sequentially exposed to each solution with a contact time of 60 s and a dissociation time of 600 s. Regeneration between each cycle was performed with 10 mM glycine, pH 2.0.

2.1.5 Microscale Thermophoresis (MST)

Fluorescent Labelling of Protein

Consumables used in the labelling of PYR1 were purchased as part of the Monolith NT™ Protein Labelling Kit RED – NHS (NanoTemper Technologies GmbH). 0.5 mg μl^{-1} PYR1 in PBS (50 mM NaH_2PO_4, 300 mM NaCl, pH 7.2) was mixed with NT-647-NHS fluorescent dye at a 1:3 molar ratio (protein-dye) and incubated for 30 min at room temperature in darkness. Conjugates were then isolated from unreacted dye using gel filtration chromatography and stored at -80 °C prior to use in MST experiments.[19]

MST Experiments

Monolith NT.115 Standard Capillaries were used throughout the MST experiments. Samples were created with serial dilutions of ligand (final sample compositions: 0.4

μg ml^{-1} NT-647-NHS-labelled protein, 1 % DMSO, 1 % Tween 20, 50 mM NaH_2PO_4, 300 mM NaCl, pH 7.2). Experiments were performed with 20 % laser excitation and varying MST power settings (typically 40 %).

2.2 An *Arabidopsis* Forward Genetic Study with eW5

2.2.1 Hypocotyl Assays

Seedling Preparation *Arabidopsis thaliana* wild-type seeds were from laboratory stocks of the Columbia (Col-0) and Landsberg erecta (Ler-0). A selection of EMS-mutagenised, hygromycin-resistant *A. thaliana* seeds were obtained from the Edward Farmer Lab. The sample set represented approximately 2260 individual lines, each with randomly-generated mutations. The mutant seeds were passed through a sieve to remove chaff, sterilised for 5 minutes in 70 % (v/v) ethanol, then scattered over agar plates containing Murashige and Skoog (MS) medium [20] (composition provided in Table 2.2) and 1.2 % (w/v) agar. The seeds were then stratified at 4 °C for 72 h, briefly aired out in a laminar flow hood to allow any surface water to evaporate, then sealed with micropore tape. The plates were set vertically and in partial shade by covering the setup using two layers of 80 gsm white paper (as discussed in previously published work on eW5 [1]) and placed in a Pericval (CU-36L5D, CLF plant climatics, Emersacker, Germany) with a photoperiod of 16/8 h at a light intensity of 150 μmol $m^2\,s^{-1}$ and a temperature of 20 ± 1 °C. After 120 h of growth in these conditions, the plates were, again, briefly aired out before being imaged using a flatbed scanner. Hypocotyls were measured using ImageJ [21] software.

eW5 Selection MS-agar plates were produced as described in Section 2.2.1, with a 'blank' control set and a set containing 100 mM eW5. Seedlings were grown and imaged as described in Section 2.2.1 and population hypocotyl lengths were compared.

2.2.2 Mutant Crossing

Mutants lines identified as being eW5-insensitive in hypocotyl assays were crossed with wild-type *A. thaliana* (the wild-type plant was the maternal plant in all crosses). Siliques (seed pods) were allowed to mature, then were harvested. The crossed seeds

Component	Concentration / mM
Ammonium nitrate	1.32×10^{2}
Boric acid	3.83×10^{-1}
Calcium chloride anhydrous	3.69×10^{1}
Cobalt chloride (II) hexahydrate	5.95×10^{-3}
Copper (II) sulfate pentahydrate	6.24×10^{-3}
Disodium edetate	1.25×10^{1}
Iron (II) sulfate heptahydrate	7.73
Magnesium sulfate	2.18×10^{1}
Manganese (II) sulfate monohydrate	2.86
Sodium molybdate dihydrate	6.05×10^{-2}
Potassium iodide	1.38×10^{-1}
Potassium nitrate	1.92×10^{2}
Potassium phosphate monobasic	2.31×10^{1}
Zinc sulfate heptahydrate	2.47

TABLE 2.2: Composition of Murashige and Skoog medium

were bulked and harvested, then validated by growing seedlings on MS-agar plates containing 40 µM hygromycin B to check that resistance conferred by the mutant parent was still present.

Before collecting material for Next-Generation Sequencing (NGS), a final hypocotyl assay with seedlings grown on eW5-MS-agar as described in Section 2.2.1 was performed and the 15 % of seedlings with the shortest hypocotyls (i.e. those showing an eW5-insensitive phenotype) were selected as material for NGS.

2.2.3 Next Generation Sequencing

DNA Extraction *A. thaliana* gDNA was extracted using the DNeasy® Plant Mini Kit (Qiagen). Samples were disrupted by grinding with a micropestle while chilled with liquid nitrogen, then subjected to the standard DNeasy® Plant Mini Kit protocol. Following extraction, DNA quality was assessed using agarose gel electrophoresis and the nanodrop feature of the DeNovix DS-11+ Spectrophotometer. DNA concentration was quantified using a Qubit Fluorometer (Invitrogen).

Next Generation Sequencing Analysis Next generation sequencing data were collected and downloaded from Novogene following Illumina sequencing. Initial processing of NGS data was performed by Wenbin Wei (Durham University). *Arabidopsis thaliana* genome in fasta format was downloaded from EnsemblPlant release

Sample	Control
A1_Bot	A12_Red
A1_Pink	A12_Red
A12_Red	A23_Green
A23_Green	A12_Red
T215	A12_Red
T238	A12_Red

TABLE 2.3: Controls used in mutect2 analysis.

52[22]. Reads quality was analysed using fastqc[23] and multiqc[24]. Reads were aligned to the genome sequence using bwa-mem[25]. Alignment statistics were generated using samtools[26] and combined using multiqc. GATK[27] (Genome Analysis Toolkit) version 4.2.4.1 was used to identify variants. Duplicated reads were removed using GATK tool MarkDuplicates. The resulting bam files and metrics files were analysed using samtools and multiqc to collect read statistics. Coverage statistics were generated using mosdepth[28] and summarized using multiqc. Variants that exist in one population but not in the control population (Table 2.3) were identified using GATK tool Mutect2 with –f1r2-tar-gz option. Artifacts of variant calls were filtered using GATK tools LearnReadOrientationModel and FilterMutectCalls. Variants that passed the filters were selected using GATK tool SelectVariants. Impact and effect of variants were predicted using software snpEff[29]. Variant information in the vcf files were extracted using software SnpSift[30].

2.3 Improving New Tools for Small Angle X-Ray Scattering Analysis

In order to provide realistic constraints for the optimisation routine, a selection of protein coordinate files were downloaded from the RCSB PDB [18]. High resolution structures obtained by X-ray crystallography were used instead of NMR models to simplify the data analysis process as NMR structures often contain multiple models that could be problematic to process. Chain length was limited between 30 and 300 residues to decrease the likelihood of obtaining multidomain proteins in the structure search and to ignore short, simple peptides.

2.3.1 Data Curation

The criteria for PDB sampling were:

- X-ray crystallography data
- Resolution < 2 Å
- $R_{Work} < 0.20$
- $R_{Free} < 0.25$
- Chain length 30 - 300 residues
- No ligands
- Minimum 2 alpha helices
- Remove redundancy by using representative structures at 70 % sequence identity (i.e. use a single structure to represent all lysozyme structures).

The search, performed 30-10-18, produced **2343** entries of which the coordinate (.pdb) files were downloaded for analysis. Many of the files downloaded from the RCSB PDB search contained multiple chains. A Python script utilising the PDB module of the BioPython [31] package was created to analyse the distribution of chain values obtained.

2.3.2 Dataset Cleaning

To ensure that every chain within the collected dataset conformed to the sampling criteria, some quality controls were imposed. In addition to reasserting the chain length constraints, an assessment of chain sequence similarity was assessed with the following process:

- Each model coordinate file was parsed with BioPython's PDBParser.
- The residue sequence of each chain in each model of the pdb file was stripped of water and cast to a BioPython Seq object.

- A list of all Seq objects in a model is iterated through, and a global pairwise alignment score was generated for each pair using BioPython's pairwise2 module. Alignments were scored with the following arguments:
  - Match parameter: 'x'. Identical characters have score of 1, otherwise 0.
  - Gap parameter: 'x'. No gap penalties.
- The lengths of both chains used in the pairwise global alignment are compared, and the longest chain is used to calculate the percentage similarity of each pair (i.e. 2 chains, 90 residues and 100 residues long, with an overlap of 80 residues would have a score of $(80/100) \times 100\% = 80\%$)
- For each model, a dictionary of chains added to that records a True/ False value based on uniqueness (False = sequence similarity is greater than assigned cutoff; chain is designated redundant)

2.3.3 Distance Mapping

Each pdb file was searched for contacting residues based on the CASP definition of a contacting residue pair [32]. Two residues are defined to be in contact if the Euclidean distance between their Cβ atoms is less than 8.0 Å (Cα for glycine). Contacts are classified as short-, medium- or long-range if they separated by 6–11, 12–23 or at least 24 residues in sequence space, respectively. The contacts with a separation of less than 6 residues were not considered as they typically correspond to contacts within secondary structure elements.

Since the SAXS data analysis routine considers Cα traces, distance mapping calculations were performed with reference to Cα atoms instead of Cβ atoms.

2.3.4 Secondary Structure Identification

Biopython possesses a DSSP [33, 34] interface to access DSSP output within a Python environment. DSSP assigns each residue a secondary structure from a choice of 8 secondary structure elements (SSEs). These 8 classes were reclassified as 3 SSEs: Helices, Strands and Linkers to simplify downstream analysis.

DSSP Code	SSE	Simplified Structure
H	Alpha helix (4-12)	H
B	Isolated beta-bridge residue	S
E	Strand	S
G	3-10 Helix	H
I	Pi helix	H
T	Turn	L
S	Bend	L
-	No SSE classification	-

TABLE 2.4: Secondary structure simplifications used

By iterating through the chains marked as valid in Subsection 2.3.2 using Biopython's DSSP module, the secondary structures can be mapped to each chain's primary sequence. Since the calculation of curvature and torsion values in Subsection 2.3.5 characterises segments of Cα chains that are four atoms long, each subsection was assigned an overall SSE based on the model SSEs of its constituent residues. When assigning the modal SSE in the case of a tie, linkers are given the highest priority and helices the second highest. If all 4 SSEs are present in one chunk (including no classification), an assignment of None ('-') is given instead.

2.3.5 Curvature and Torsion Analysis

Curvature Calculation

The **curvature** (κ) of three points, c_1, c_2, c_3, is given by:

$$\kappa(c_1, c_2, c_3) = \frac{2|sin(\theta_{123})|}{||c_1 - c_2||}$$

Where $\theta_{1,2,3}$ is the angle between vectors $c_1 - c_2$ and $c_2 - c_3$. This can be extended to sections of a discrete curve containing four points by using midpoints. For a curve subsection, m_i, comprising four points:

$$m_i = \frac{c_{i+1} + c_i}{2}$$

Thus for a section (c_1, c_2, c_3, c_4) we can define:

$$\kappa(c_1, c_2, c_3, c_4) \equiv \kappa(m_1, m_2, m_3)$$

Torsion Calculation

Torsion (τ) is a measure of curve non-planarity. For a curve subsection of four points (c_1, c_2, c_3, c_4), two planes can be defined for (c_1, c_2, c_3) and (c_2, c_3, c_4) with plane normals n_1 and n_2, respectively. Plane normals are unit vectors given by:

$$N = (c_{i+1} - c_i) \times (c_{i+2} - c_{i+1}),$$

$$n = \frac{N}{||N||}$$

If the curve is planar, n_1 and n_2 will be equal. Otherwise, they will differ. The angle between n_1 and n_2 can be used as a measure of curve subsection non-planarity. From this, the curve torsion can be calculated using:

$$\tau(c_1, c_2, c_3, c_4) = \frac{2}{l} sin(\frac{\theta_n}{2}),$$

$$l = \frac{||c_2 - c_1|| + ||c_3 - c_2|| + ||c_4 - c_3||}{3}$$

Where θ_n is the angle between n_1 and n_2 This definition takes into account the length of the curve as well.

For each protein, curvature and torsion values were calculated for every subsection comprising four Cα atoms and recorded against the modal secondary structure element of each subsection.

2.4 Applying Machine Learning Methods to Hypocotyl Elongation Analysis

A pipeline was developed to automatically collect and analyse data from images of *A. thaliana* hypocotyl elongation assays. The pipeline involved ascertaining the location of Petri dishes in an image using a circle Hough Transform, detecting hypocotyls within the boundaries of each Petri dish using a machine learning model built around a recurrent convolutional neural network (RCNN), then measuring the length of detected hypocotyls. Images of hypocotyl elongation assays prepared as described in Subsection 2.2.1 were used in the training and validation of the RCNN. Hypocotyls were annotated with bounding boxes using ImageJ [21] software.

2.4.1 Hough Transform

Image manipulation and circle Hough transformation were performed using the Python3 implementation of OpenCV, cv2 [35]. Master images containing between one and six Petri dishes arranged in up to three rows of two were converted from three-channel blue-green-red to single-channel grayscale. A Gaussian box blur with a kernel size of 5 was applied to the images. A circle Hough transform was then performed on the image using the HoughCircles function of OpenCV with an accumulator resolution ratio of 1:1, minimum distance between circles of 1200 px, Canny edge threshold of 200 and circle centre accumulator threshold of 50. Circles detected by the circle Hough transform were also constrained to a radius size of 1750-1950 px.

2.4.2 Faster-RCNN Generation

The PyTorch implementation[36] of the Faster-RCNN neural network[37] was trained using a manually-labelled image set of petri dishes comprising 4323 hypocotyls with annotated bounding box information.

Chapter 3

Biophysical Studies of PYL Family Proteins with eW5

3.1 Introduction

The sessile nature of plants demands biochemical ingenuity to cope with the difficulties of a life rooted in one place. An array of phytohormones are produced by plants to adapt their growth, development and physiology in response to environmental stressors. One of the most important phytohormones in higher plants is (+)-abscisic acid (ABA), an apocarotenoid that influences a diverse set of transcriptional and physiological responses.[38]

Significant advances in the dissection of ABA signalling pathways have been made in the last decade, revealing an assembly of protein families that are involved in the core ABA pathway. A family of proteins thought to be at the apex of the core ABA pathway includes pyrabactin resistance 1 (PYR1) and a set of 13 PYR1-like (PYL) proteins,[39] also known as regulatory components of ABA receptors (RCARs).[40]

Elucidation of the key players in the core ABA signalling pathway was facilitated largely by the discovery of pyrabactin, a synthetic, structurally dissimilar ABA agonist that can be used to sidestep genetic redundancy within the pathway through a chemical genetic approach.[39, 41] The ability to control the action of ABA receptors is desirable in an agricultural setting, as it could increase the survivability of plants in drought conditions through receptor activation or improve the growth of plants in optimal conditions through receptor repression. ABA itself is unsuitable

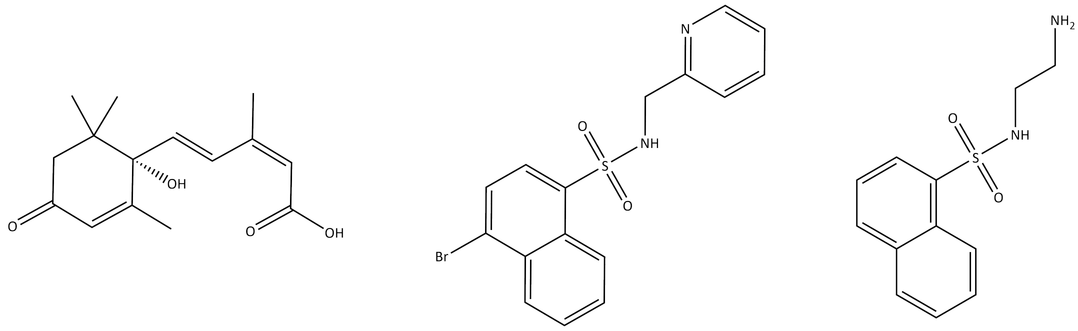

FIGURE 3.1: Left to right: Skeletal formulae of ABA, pyrabactin and eW5

for commercial agrochemical use as it is metabolically unstable[42, 43] and prone to photodegradation.[44] However, pyrabactin and its derivatives present an easily-synthesised and stable family of compounds that could become an attractive source of novel drought-tolerance agents.

This chapter explores the current knowledge of the core ABA signalling pathway and attempts to explain the effects of eW5, a potential novel binding partner of the PYR/PYL/RCAR family of *Arabidopsis* ABA receptors (henceforth referred to as the PYL family).

3.1.1 ABA, Pyrabactin and eW5

Since its discovery in the 1960s, ABA has been identified as a signalling compound that coordinates a diverse range of biological processes.[45–47] PYLs bind to ABA with a strong preference over its non-physiological stereoisomers.[40, 48] Overall ABA affinity and the extent to which ABA elicits receptor activity varies within the PYL family; some PYLs, such as PYL10, are capable of inhibiting PP2C activity even in the absence of ABA.[49]

A subset of the PYL family have also been shown to bind to a structurally-unrelated, synthetic ligand called pyrabactin; some in an agonistic manner and others in an antagonistic manner. Despite the differences in ligand activity, crystallographic studies indicate that both ABA and pyrabactin bind at the same site.[39, 50, 51]

eW5 is a synthetic ligand that has been shown to enhance radicle and hypocotyl growth in *Arabidopsis* samples [1]. Due to its effect on growth and its structural similarity to pyrabactin, eW5 was predicted to interact with the PYL family in some way.

3.1.2 The Core ABA Pathway

Current models of the core ABA pathway suggest that the signal cascade is initiated by the binding of ABA to members of the PYL family, a set of 14 soluble receptor proteins.[40, 52–54] In the presence of ABA, these proteins inhibit a key set of negative regulators in the ABA pathway, the clade A type 2C protein phosphatase (PP2C) family.[55, 56] Without inhibition by PYLs, PP2Cs bind and dephosphorylate SNF1-related protein kinases (SnRK2s), a family of autophosphorylating proteins responsible for the activation of further downstream targets (see Figure 3.2).[50, 57–59]

ABA-responsive transcription is achieved via the activation of ABA-responsive promoter elements (ABREs) by ABRE binding factors (ABFs), basic leucine zipper transcription factors that are phosphorylated by SnRK2s.[50, 60–63]

The archetypal ABA-mediated physiological process is the closure of stomata to conserve water in drought conditions. Each stoma is surrounded by a pair of guard cells that are responsible for opening and closing the pore. The aperture of stomatal pores is regulated by the movement of ions across guard cell membranes, which is typically coordinated by ion channel activation by SnRK2s or ion channel deactivation by PP2Cs. Guard cell ion channels are also regulated by calcium-dependant kinases, which present another layer of control that works in tandem with the direct phosphorylation and dephosphorylation of ion channels by SnRK2s and PP2Cs, respectively.[64–66]

3.1.3 The PYL Family

Early efforts at dissecting the core ABA pathway through a genetic knockout approach were complicated by functional redundancy within the PYL family.[39] Overall sequence identity between the 14 members of the family varies between 36 % and 81 %, but all members to share a steroidogenic acute regulatory-related lipid transfer (START) domain, characterised by an α/β fold flanked by a 'half-pipe' incomplete

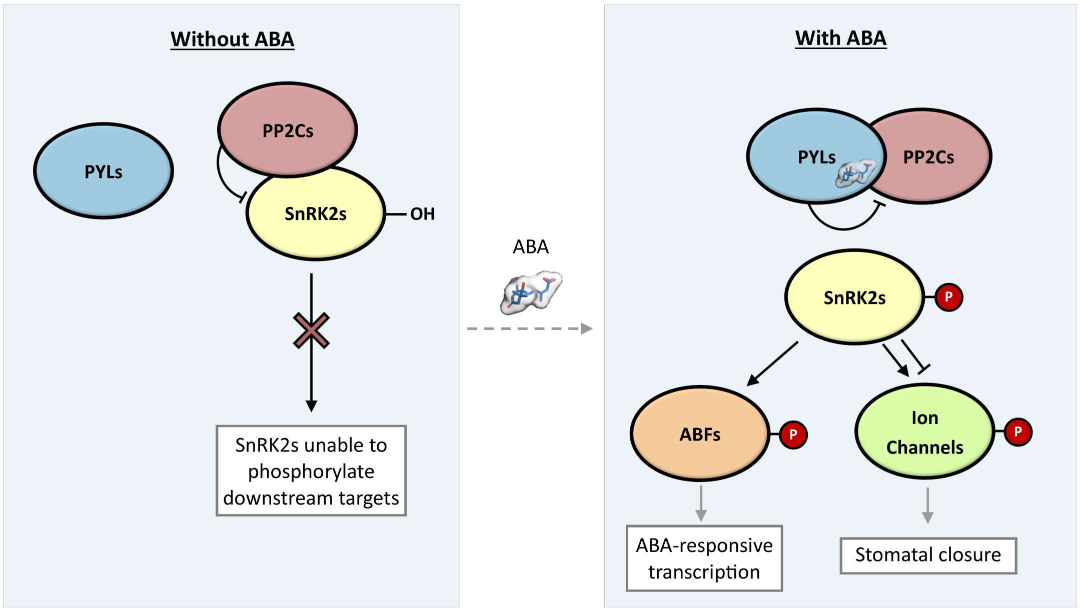

FIGURE 3.2: Simplified overview of the core ABA pathway. In the absence of ABA, PYLs are generally unable to inhibit the action of PP2C-family proteins, meaning SnRK2s do not phorphorylate downstream targets of the pathway. In the presence of ABA, PYLs can complex with PP2Cs, inhibiting their activity. This allows SnRK2s to phosphorylate downstream targets, causing ABA-related physiological and transcriptional responses.

β-barrel.[67] PYLs also possess a well-conserved ABA-binding pocket, illustrated by the multiple sequence alignment presented in Figure 3.3.

The PYL-PP2C interaction crucial to ABA signal transduction is governed by the conformation of two highly-conserved loops that surround the PYL ABA binding pocket and are described as the 'gate' and 'latch' in a PYL-ABA-PP2C ternary complex. Crystal structures of apo and ABA-bound PYR1 and PYL2 suggest that the binding of ABA has no significant effect on the latch region (residues 119-121). However, a large conformational change is observed in the gate region (residues 89-93), which moves 3-9 Å to preclude solvent access to the binding cavity (see Figure 3.4).[53, 72] These conformational changes create an interface that facilitates PP2C association as part of a ternary complex (see Figure 3.5).[72]

Complexation with a PP2C can also confer additional stability to the PYL-ABA pair. The indole ring of Trp385 of HAB1, a PP2C, was observed to be inserted between the gate and latch regions of PYL2 in the PYL2-ABA-HAB1 ternary complex, reinforcing the architecture of the ABA-binding pocket (see Figure 3.5). The orientation of HAB1's Trp385 also allows the formation of a water-mediated hydrogen

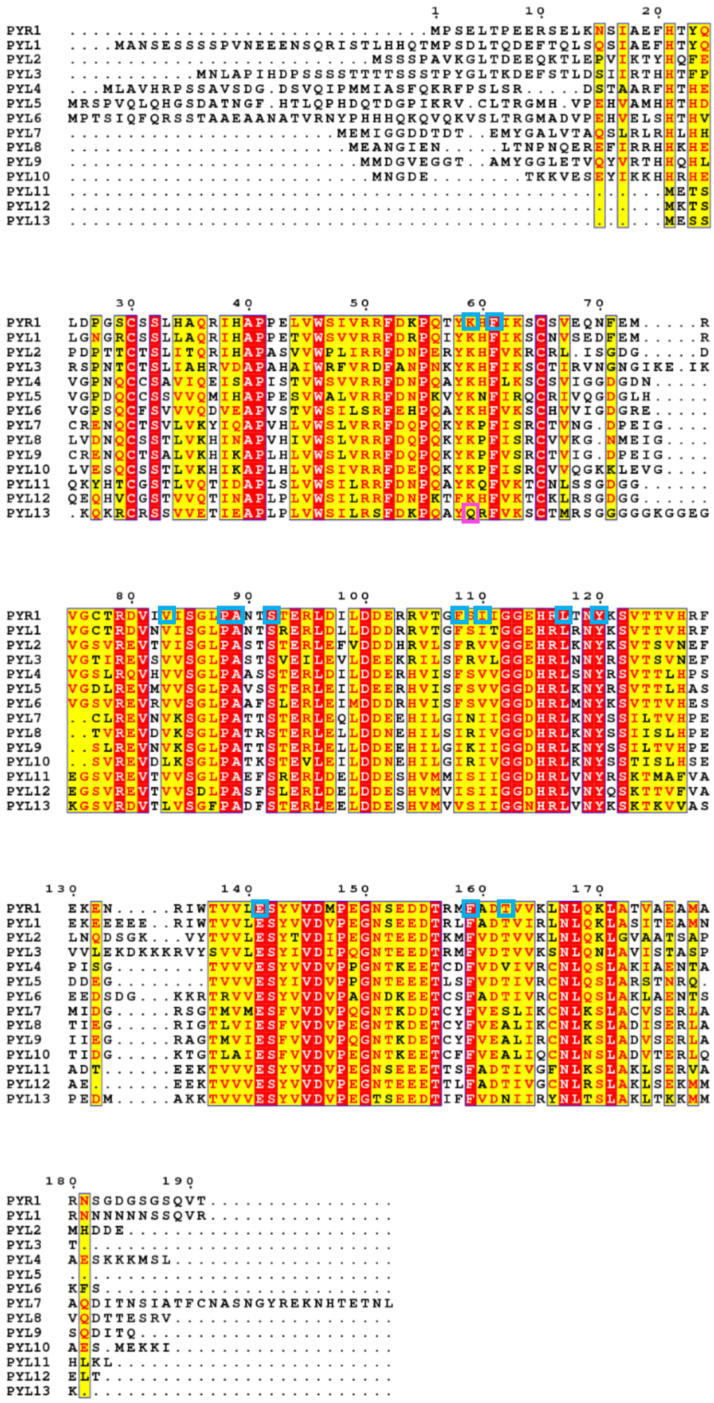

FIGURE 3.3: Multiple sequence alignment of the PYL family performed using Clustal Omega.[68–70] Residues of PYR1 that are within 5 Å of ABA in the holoprotein are highlighted with a blue box. The conserved ABA-binding site Lys → Gly mutation that renders PYL13 ABA-insensitive is also highlighted with a pink box. Figure produced using Espript.[71]

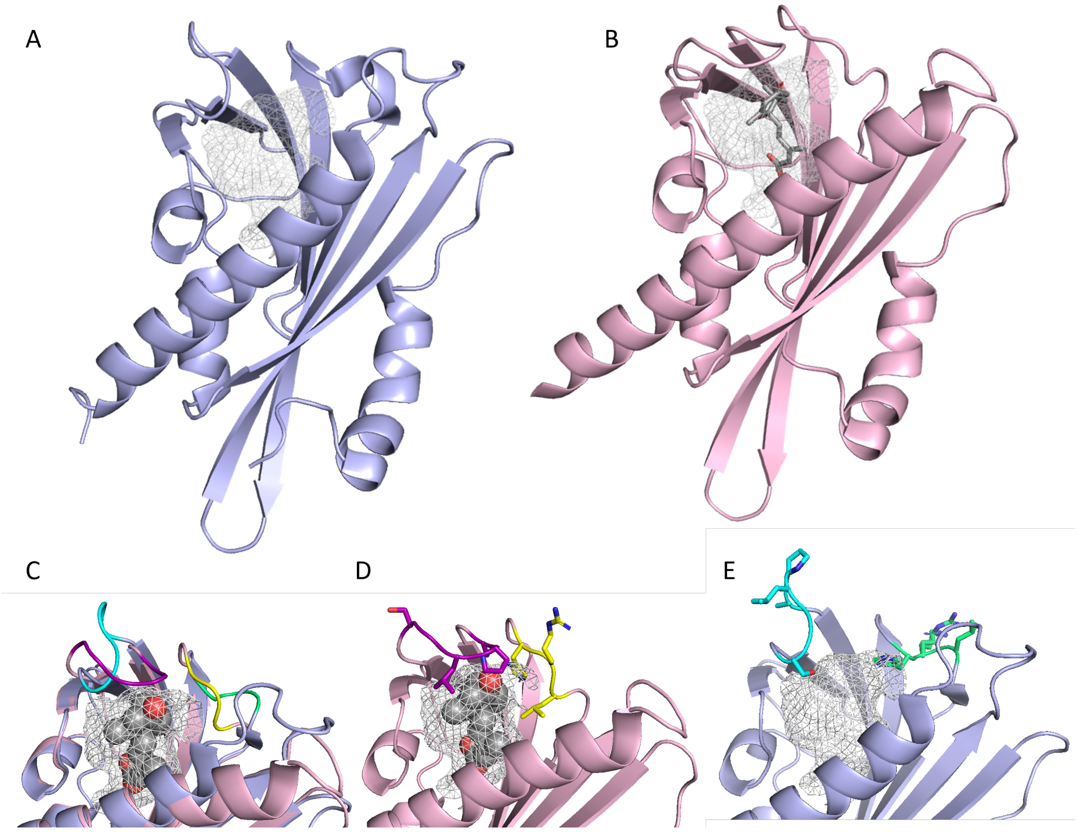

FIGURE 3.4: A) Ribbon structure of apo-PYR1 with the ABA binding pocket shown as a white mesh B) Ribbon structure of PYR1-ABA complex with bound ABA shown C) Superimposed ribbon structures of apo-PYR1 (lilac) and ABA-bound PYR1 (pink). The 'gate' loop is coloured yellow in holo-PYR1 and green in apo-PYR1, and the 'latch' loop is coloured purple in holo-PYR1 and cyan in apo-PYR1. D) ABA binding pocket of holo-PYR1, with side chains of the gate and latch loops shown. E) ABA binding pocket of apo-PYR1, with side chains of the gate and latch loops shown. Both structures from PDBID: 3K3K (crystallised as a dimer comprised of apo and holo units).[53]

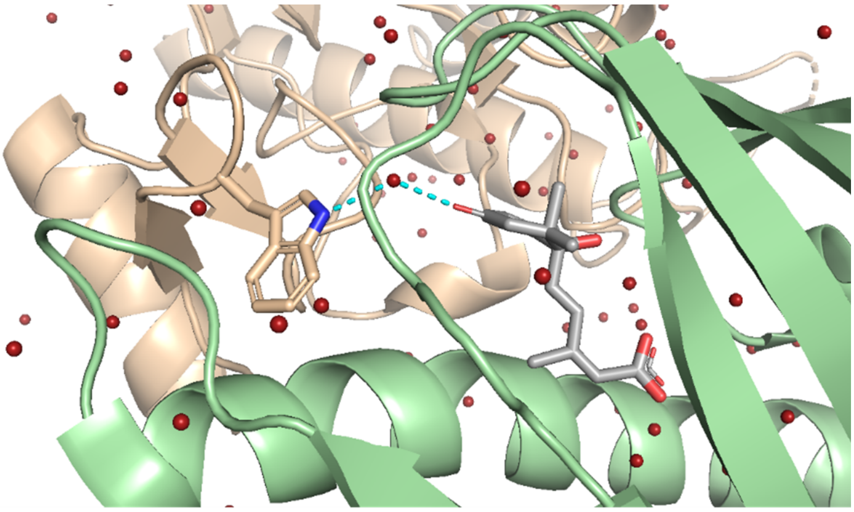

FIGURE 3.5: Ribbon representation of the PYL2-ABA-HAB1 complex (PDBID: 3KB3). HAB1 is coloured orange and PYL2 is coloured green. The water-mediated hydrogen bond between Trp385 of HAB1 (side chain shown) and the carbonyl of ABA is represented as cyan dashed lines.[72]

bond with the carbonyl of ABA, further strengthening the complex.[72] Given the high degree of structural conservation observed within the PYLs, it is hypothesised that the 'gate-latch-lock' mechanism is common amongst all members of the family that complex with ABA.

The action of pyrabactin on PYLs is notably different to ABA in several ways. While ABA is a pan-agonist of the PYLs (with the exception of the ABA-independent PYL13),[73] pyrabactin is an agonist for some PYLs and an antagonist for others.[39] This antagonism can be explained by the existence of inactive pyrabactin binding modes with several of the PYLs.[74]

Like ABA, pyrabactin interacts agonistically with PYR1 by promoting gate loop closure. Pyrabactin adopts a compact conformation with the pyridyl ring nestled between its naphthalene ring and the aromatic side chain of Tyr120. With pyrabactin in this binding pose, the closed conformation of PYR1's gate loop is stabilised through contact with the bromine and naphthyl regions of pyrabactin.[74] A similar binding pose is adopted in PYL1, which also weakly binds agonistically with pyrabactin. In the case of PYL1, π-stacking interactions align the pyrabactin naphthyl and pyridyl regions with Tyr147, again stabilising interaction between the gate loop and the bromine and naphthyl regions of pyrabactin.[51]

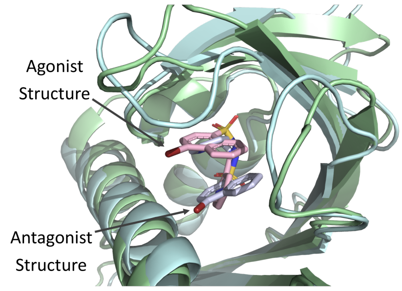

FIGURE 3.6: Overlay of pyrabactin bound to PYL1 (cyan/ pink, PDB ID 3NEF) and PYL2, (green/ blue, PDB ID 3NMH), illustrating the 180° rotation of pyrabactin when comparing its agonist and antagonist binding modes.

Antagonistic pyrabactin activity is caused by stable pyrabactin binding that does not promote gate closure. Pyrabactin binds to PYL2 in an antagonistic manner, with a pose that appears to be rotated 180° from the productive binding mode. In the case of PYL2, the naphthyl group stacks in the middle of the pyrabactin pyridyl region and Tyr124 of PYL2. This orients the bromine of pyrabactin in the opposite direction to the gate region of PYL2, preventing their interaction and disfavouring gate loop closure.[51] It is worth noting that the favoured orientation of pyrabactin (productive or non-productive biding mode) can be changed by the selective mutation of only one residue in the ABA-binding site. For example, an I137V mutation converts PYL1 to a pyrabactin-inhibited receptor by forcing pyrabactin to adopt the non-productive binding pose due to steric pressure on its naphthyl region.[51]

Study of the PYL family's role in ABA signalling has revealed a complex system of control that must be deconvoluted to explain the effect of eW5 on growth. This project aimed to express and purify PYL family proteins and test their potential interactions with eW5 using a selection of biophysical techniques.

3.2 Results

3.2.1 Protein Production

PYR1 Production

The PYR1 construct used was expressed with an N-terminal Strep-II tag and a C-terminal His6 tag (see Figure 3.7). Expression tests with the original transformed cells provided by a colleague showed poor expression (the cells were later determined to be a DH5α strain), so the plasmid was extracted and used to transform BL21 (DE3) cells. The resulting cells displayed significantly better expression (see Figure 3.8).

FIGURE 3.7: PYR1 construct used in initial experiments.

IMAC purification using standard HisTrap conditions with a gradient elution typically yielded approximately 70 mg of protein per litre of culture. SDS-PAGE analysis of eluted product suggested minimal contamination from co-eluted proteins (see Figure 3.8), so no further purification of the product was performed before biophysical testing.

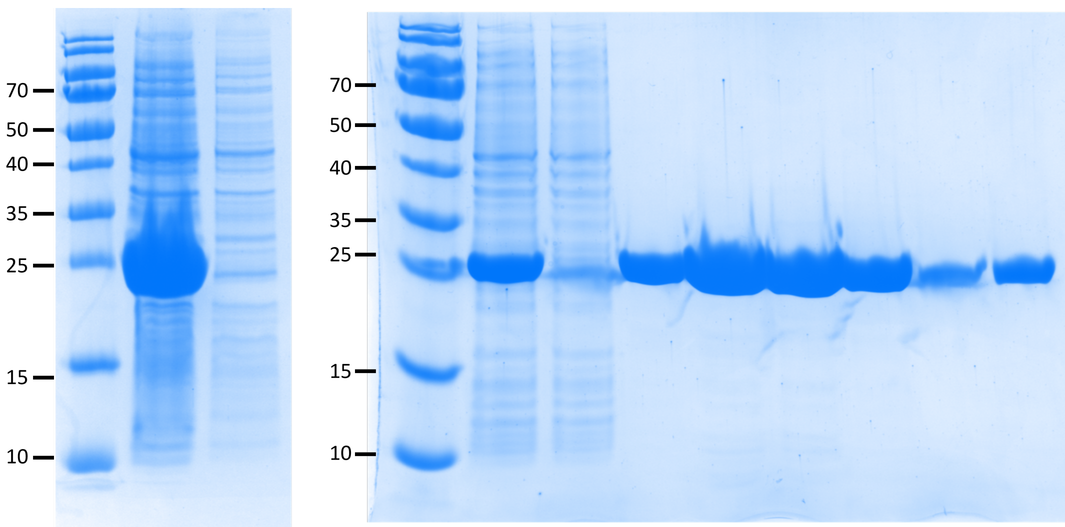

FIGURE 3.8: Left: SDS-PAGE gel of PYR1 (expected relative mass 25.38 kDa) expression test. From left to right: ladder, post-induction, pre-induction. Right: SDS-PAGE gel of PYR1 IMAC purification. From left to right: ladder, lysate before application to HisTrap FF column, HisTrap FF column flow-through, 6 collected fractions. Approximate relative masses in kDa are given to the left of the ladder references.

PYL Family Production

Additional members of the PYL family were expressed as described in Section 2.1.2.

Expression Tests Before attempting purification, the expression and solubility of additional PYL family proteins were tested. Figures 3.9, 3.10 and 3.11 show the results of expression tests for PYL1-PYL5, as well as PYR1 in a PET28-a(+) vector to compare with expression levels of the original.

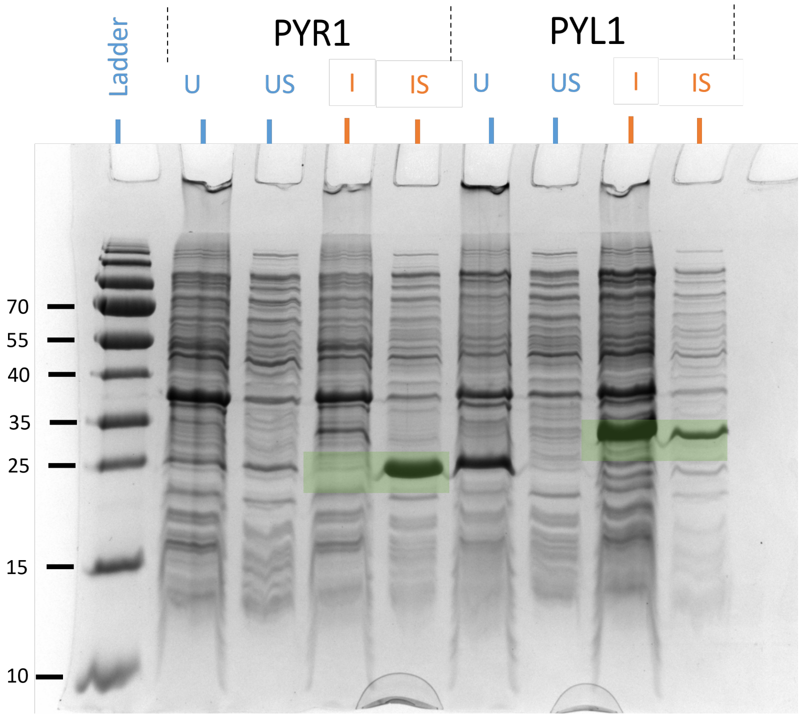

FIGURE 3.9: Expression tests of PYR1 (expected relative mass 25.38 kDa) and PYL1 (expected relative mass 26.27 kDa). Approximate relative masses in kDa are given to the left of the ladder reference. The expected positions of bands corresponding to each protein are highlighted in the lanes corresponding to induced fractions. Soluble (S) and insoluble fractions of induced (I) and uninduced (U) cell lysate were collected, with each of the four corresponding fractions labelled accordingly.

Figure 3.9 suggests good expression of soluble PYR1 and PYL1 from the PET28-a(+) plasmid, but the yields of PYR1 are reduced compared to what was obtained in original trials. PYR1 production was continued to the original specifications.

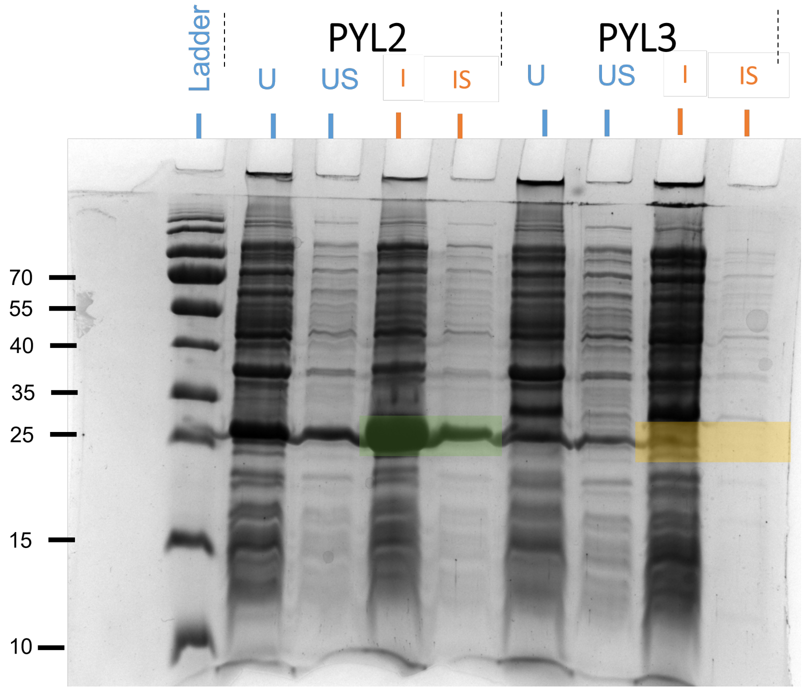

FIGURE 3.10: Expression tests of PYL2 (expected relative mass 22.18 kDa) and PYL3 (expected relative mass 24.19 kDa). Approximate relative masses in kDa are given to the left of the ladder reference. The expected positions of bands corresponding to each protein are highlighted in the lanes corresponding to induced fractions. Soluble (S) and insoluble fractions of induced (I) and uninduced (U) cell lysate were collected, with each of the four corresponding fractions labelled accordingly.

PYL2 showed good expression with a reasonable fraction of protein present in the soluble fraction following lysis. PYL2 showed significant expression prior to induction as well. PYL3 showed poor overall expression and solubility in initial trials with BL21(DE3) cells (NEB); Figure 3.10 displays the scarcity of soluble protein at expected migration values on an SDS-PAGE gel. An alternative strain, BL21-CodonPlus (Agilent) was explored to express workable amounts of PYL3. The change in strain proved successful, yielding enough protein to successfully purify using IMAC chromatography (Figure 3.14).

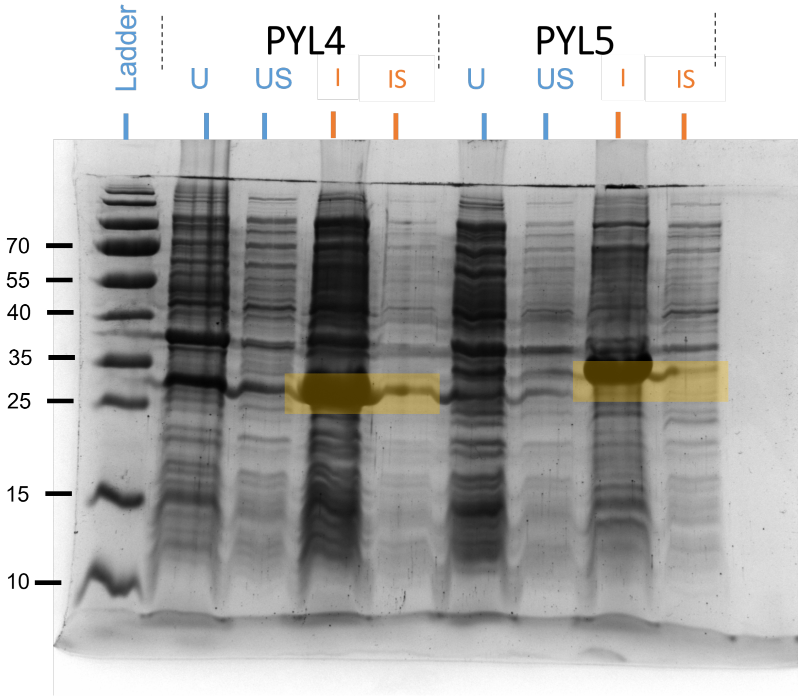

FIGURE 3.11: Expression tests of PYL4 (expected relative mass 23.34 kDa) and PYL5 (expected relative mass 23.57 kDa). Approximate relative masses in kDa are given to the left of the ladder reference. The expected positions of bands corresponding to each protein are highlighted in the lanes corresponding to induced fractions. Soluble (S) and insoluble fractions of induced (I) and uninduced (U) cell lysate were collected, with each of the four corresponding fractions labelled accordingly.

PYL4 and PYL5 showed good expression levels, but the majority of the protein following induction was present in the insoluble fraction of the lysed cells. Further optimisation of expression temperatures showed an increased yield of soluble protein at lower induction temperatures.

Purification Figures 3.12-3.16 show SDS-PAGE gels of PYL family proteins following IMAC purification. PYL1 showed excellent expression levels of soluble protein, which was successfully purified using a HisTrap column. The displayed gel shows a significant loss of protein in the flowthrough fraction. Subsequent purifications used a larger HisTrap column volume to increase the binding capacity.

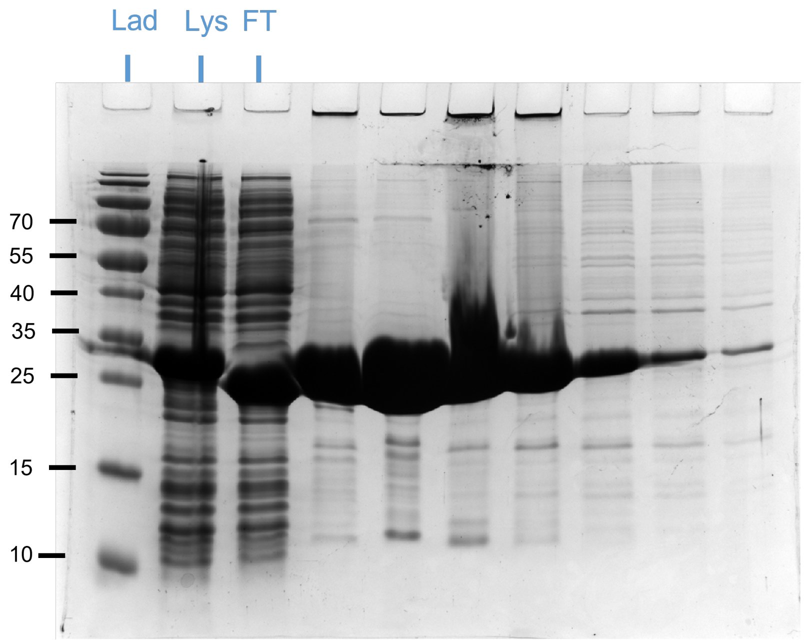

FIGURE 3.12: SDS-PAGE gel of PYL1 (expected relative mass 26.27 kDa). Left to right: reference ladder, cell lysate, column flowthrough, collected fractions.

PYL2 showed good expression levels of soluble protein, which was successfully purified using a HisTrap column.

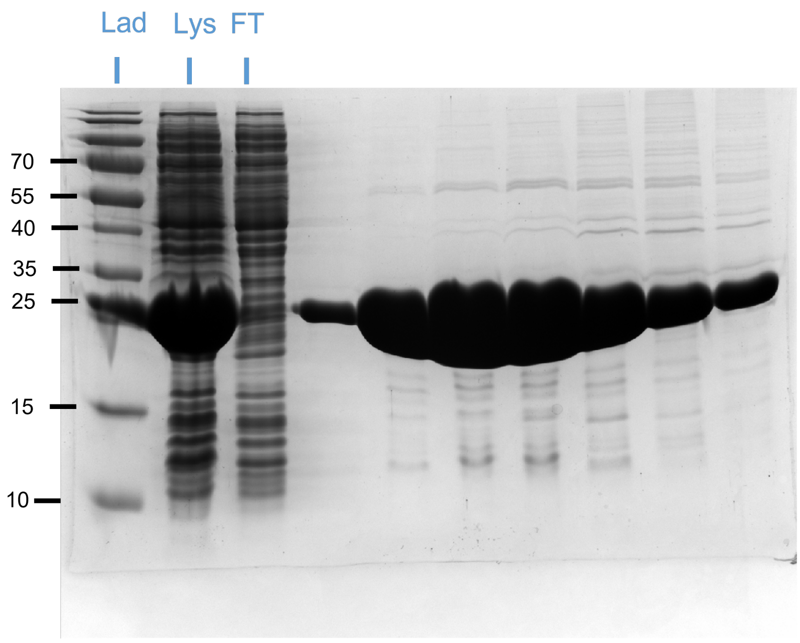

FIGURE 3.13: SDS-PAGE gel of PYL2 (expected relative mass 22.18 kDa). Left to right: reference ladder, cell lysate, column flowthrough, collected fractions.

PYL3 showed good expression with BL21-CodonPlus cells. The gel also appears to show a band corresponding to dimeric PYL3 at approximately 50 kDa.

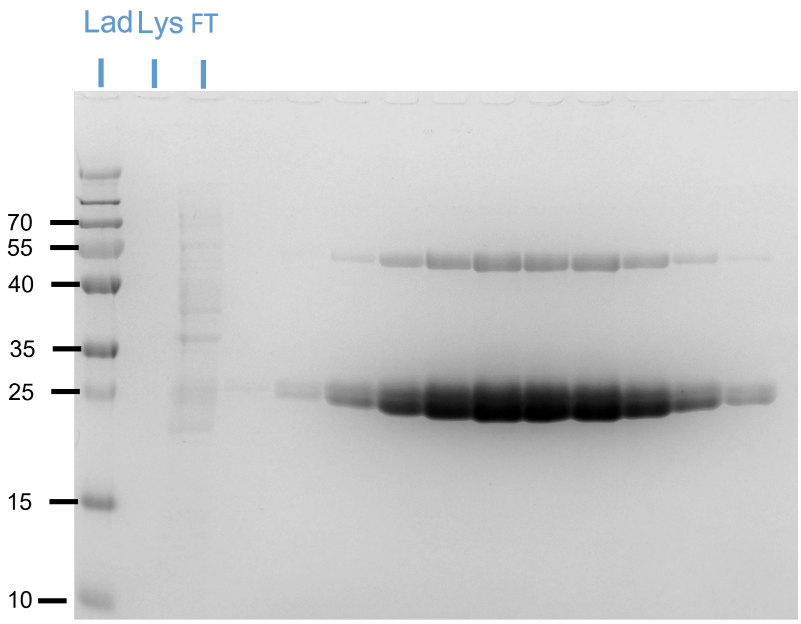

FIGURE 3.14: SDS-PAGE gel of PYL3 (expected relative mass 24.19 kDa). Left to right: reference ladder, cell lysate, column flowthrough, collected fractions.

PYL4 showed good expression levels of soluble protein when expressed at 16 °C, which was successfully purified using a HisTrap column. In this case, the three fractions containing the highest concentration of PYL4 were retained.

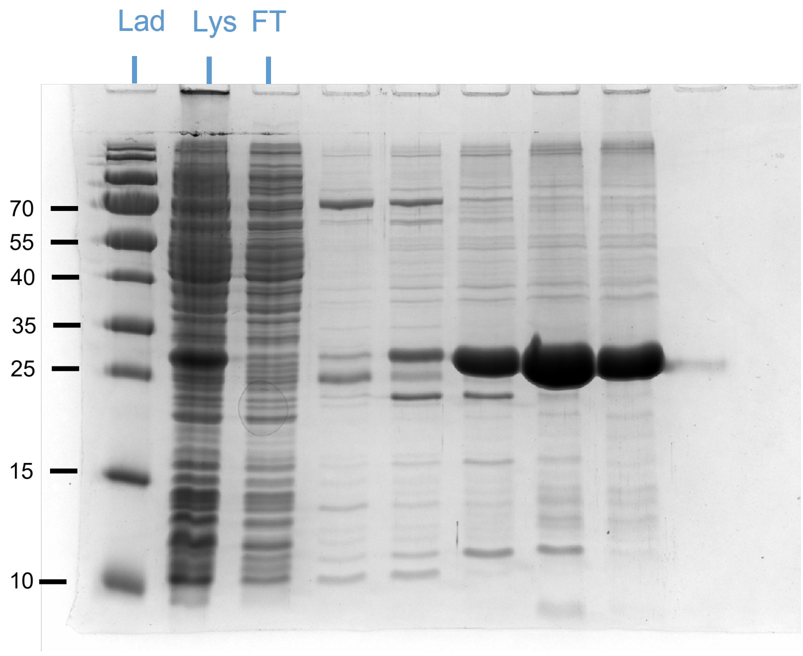

FIGURE 3.15: SDS-PAGE gel of PYL4 (expected relative mass 23.34 kDa). Left to right: reference ladder, cell lysate, column flowthrough, collected fractions.

PYL5 showed good expression levels of soluble protein when expressed at 16 °C, which was successfully purified using a HisTrap column. In this case, the four fractions containing the highest concentration of PYL5 were retained.

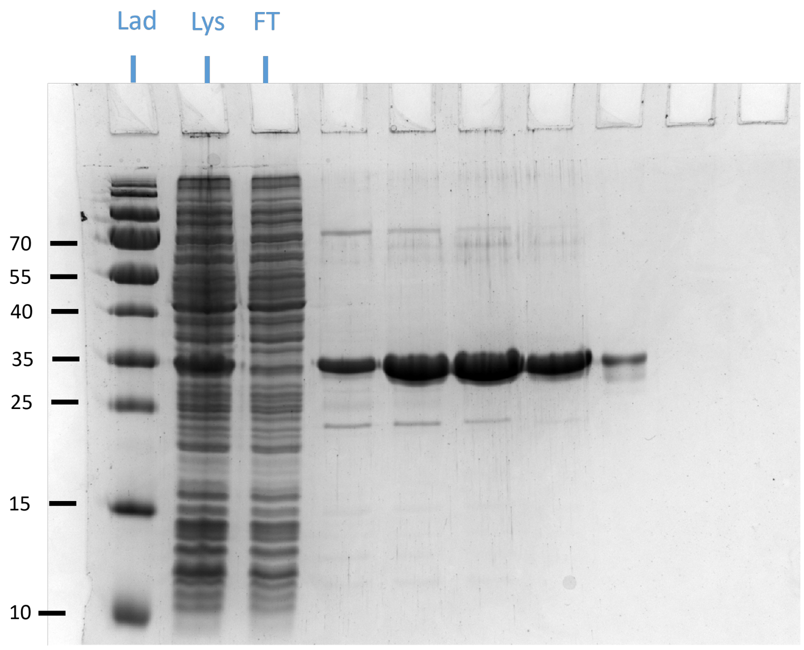

FIGURE 3.16: SDS-PAGE gel of PYL5 (expected relative mass 23.57 kDa). Left to right: reference ladder, cell lysate, column flowthrough, collected fractions.

In summary, PYR1 and several members of the PYL family were expressed and purified in workable quantities following several optimisations to expression and purification.

3.2.2 Microscale Thermophoresis

Purified PYR1 was coupled to NT-647-NHS fluorescent dye and separated from unbound protein through gel filtration. A 500 µl fraction of 0.15 mg ml^{-1} fluorescently-labelled protein was eluted and used for MST experiments. MST responses were calculated as the difference between baseline fluorescence (0.5 s before the thermal gradient is applied) and fluorescence 14.5 s after the thermal gradient is applied. Figure 3.17 shows MST traces for PYR1 in the presence of ABA and illustrates the points at which fluorescence responses were recorded.

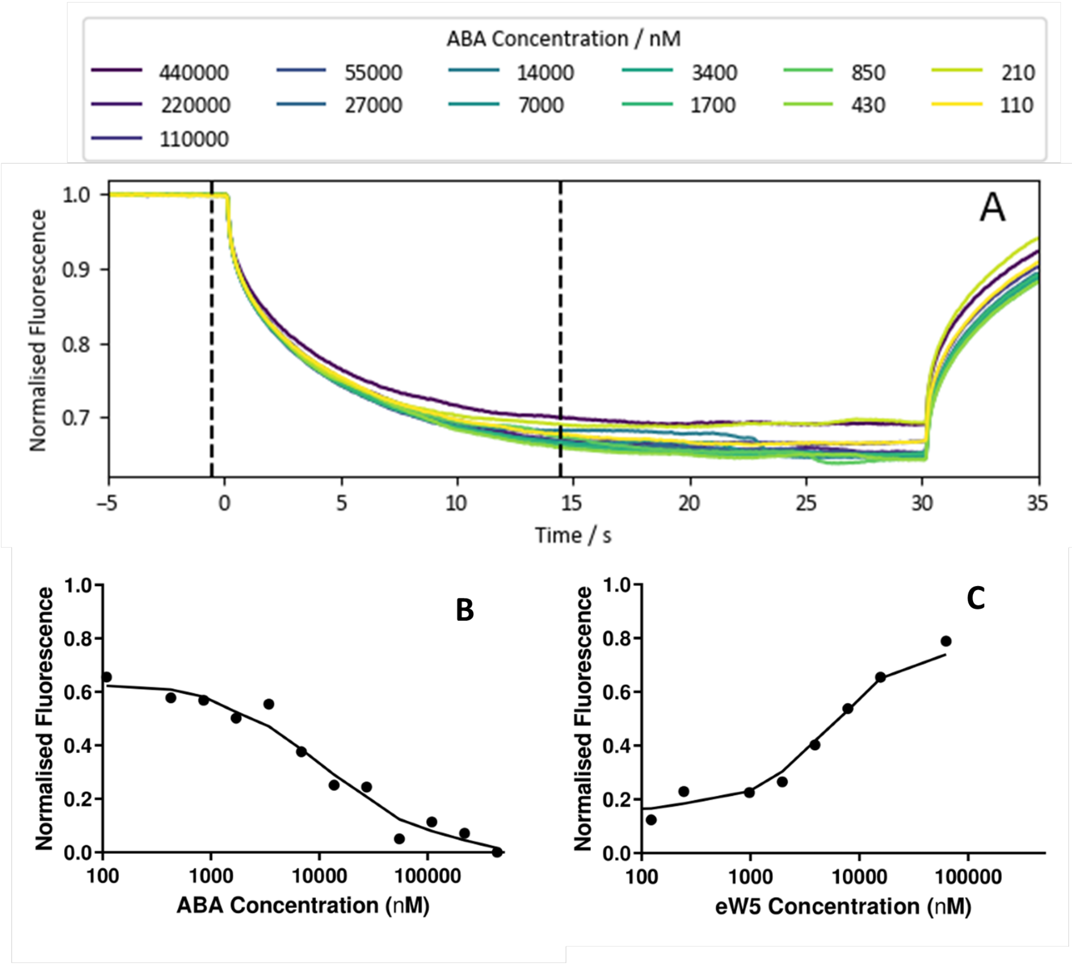

FIGURE 3.17: A) Example normalised MST traces from PYR1 with ABA. Vertical lines are shown where fluorescence readings were measured to calculated MST responses for each condition. B) Variation of MST response of PYR1 with varying concentrations of ABA. C) Variation of MST response of PYR1 with varying concentrations of eW5.

3.2.3 Surface Plasmon Resonance

Affinity Capture - Sensor Chip NTA

Initial SPR experiments involved immobilising PYR1 noncovalently through its His_6 tag to a Sensor Chip NTA. Figure 3.18 shows a representative sensorgram of a typical Sensor Chip NTA cycle with PYR1. While capture levels reached acceptable levels (the change in response was approximately 2500 RU for 50 µg ml^{-1} PYR1) the SPR response decayed significantly after exposure to PYR1 stopped.

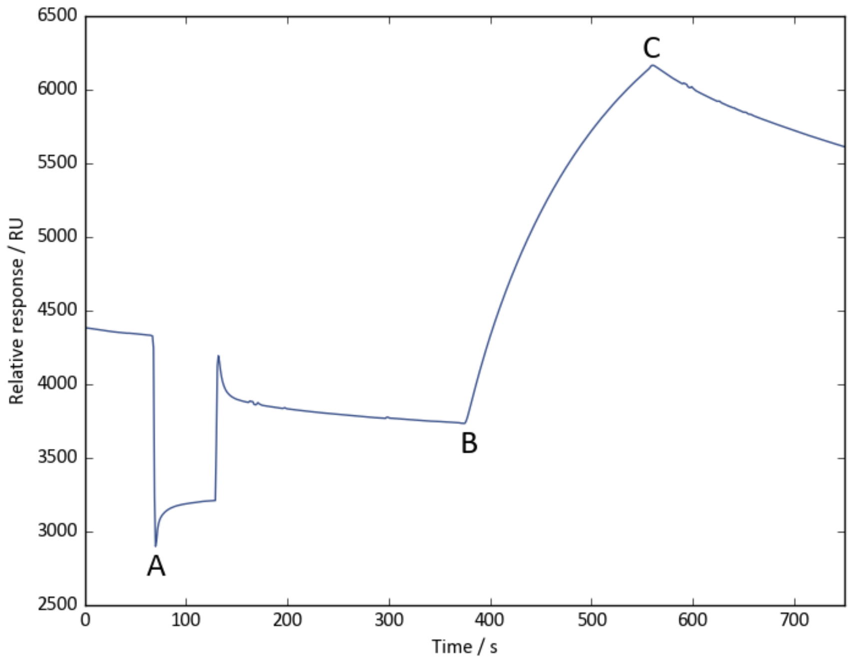

FIGURE 3.18: Sensorgram of PYR1 capture with a Sensor Chip NTA. Point A corresponds to an initial conditioning of the sensor chip surface using 0.5 mM $NiCl_2$, point B corresponds to exposure of the sensor chip surface to PYR1 and point C corresponds to the termination of PYR1 exposure. No small molecule solution is tested in this sensorgram.

Covalent Immobilisation - Sensor Chip CM5

pH Scouting The Sensor Chip CM5 (GE Healthcare) possesses a carboxymethylated dextran surface, which can covalently react with amine groups on a protein after surface activation with NHS/ EDC. To maximise the yield of this step, it is important to generate high local concentrations of protein at the sensor chip surface prior to reaction in a process called preconcentration. By protonating the carboxymethyl moieties of the sensor chip surface, it is possible to electrostatically attract protein and induce workably high concentrations even from dilute sample. For most proteins, preconcentration is at its highest at low pH, where a larger population of carboxymethyl groups are protonated. However, low pH can also induce nonspecific binding of protein to sensor chip surface and reduce the efficiency of the NHS/ EDC coupling step, reducing overall immobilisation yield. Therefore, the general goal of pH optimisation experiments is to find the highest pH value that provides workable preconcentration and does not show nonspecific binding of protein to the sensor chip surface. Since preconcentration is a noncovalent interaction, it is possible to assay several pH values using the same SPR chip.

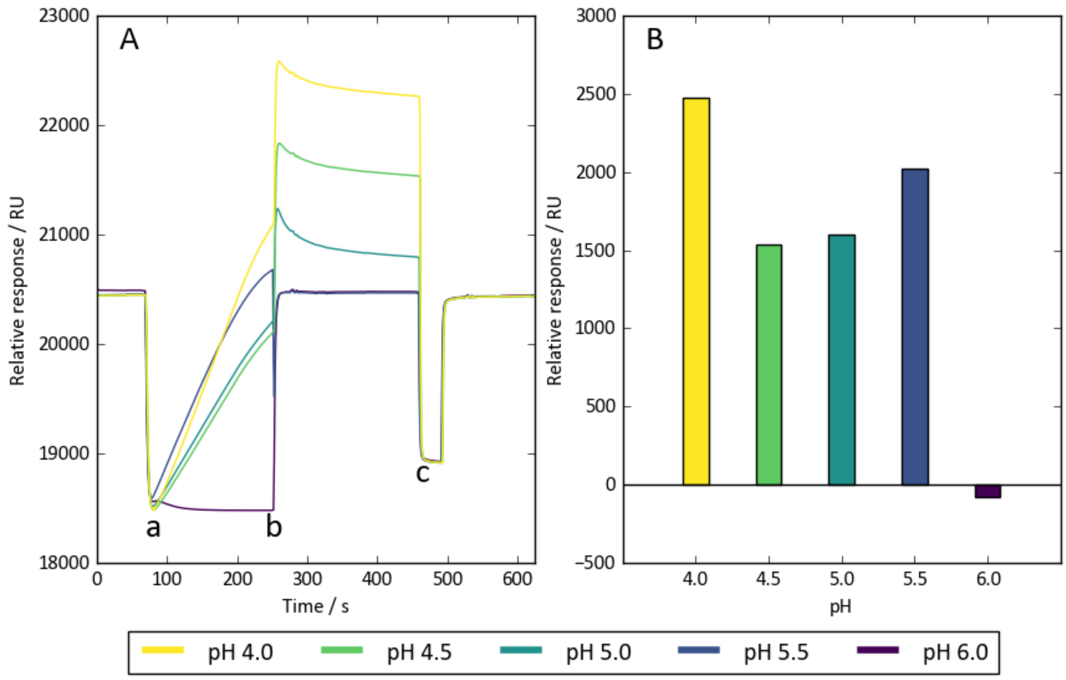

FIGURE 3.19: A) Sensorgrams showing PYR1 preconcentration in 10 mM acetate buffer at pH values from 4.0-6.0. Major events in the experiment are: a) introduction of PYR1 to sensor chip surface; b) stopping of PYR1 exposure, flow of buffer continues; c) regeneration of surface using 50 mM NaOH.B) Plot of SPR response change relative to baseline (point a) obtained just before the chip's exposure to PYR1 is stopped (time = 239 s).

Figure 3.19 shows sensorgram data from PYR1 preconcentration experiments. As Figure 3.19B shows, pH 4.0 provides the highest levels of PYR1 preconcentration. However, Figure 3.19A reveals that pH 4.0 also causes significant nonspecific interaction with the sensor chip surface (indicated by the large residual SPR response after point b). Since pH 5.5 does not show nonspecific interaction and is likely to give a better yield in the amine coupling reaction while still giving an acceptable preconcentration level, it was selected as the buffer pH for the PYR1 coupling step.

Amine Coupling Amine coupling was achieved through surface activation with EDC/ NHS followed by reaction with protein amine groups. The sensorgram in Figure 3.20 shows the SPR response throughout the coupling process. The final response following the immobilisation procedure increased by 477.0 RU, indicating successful reaction.

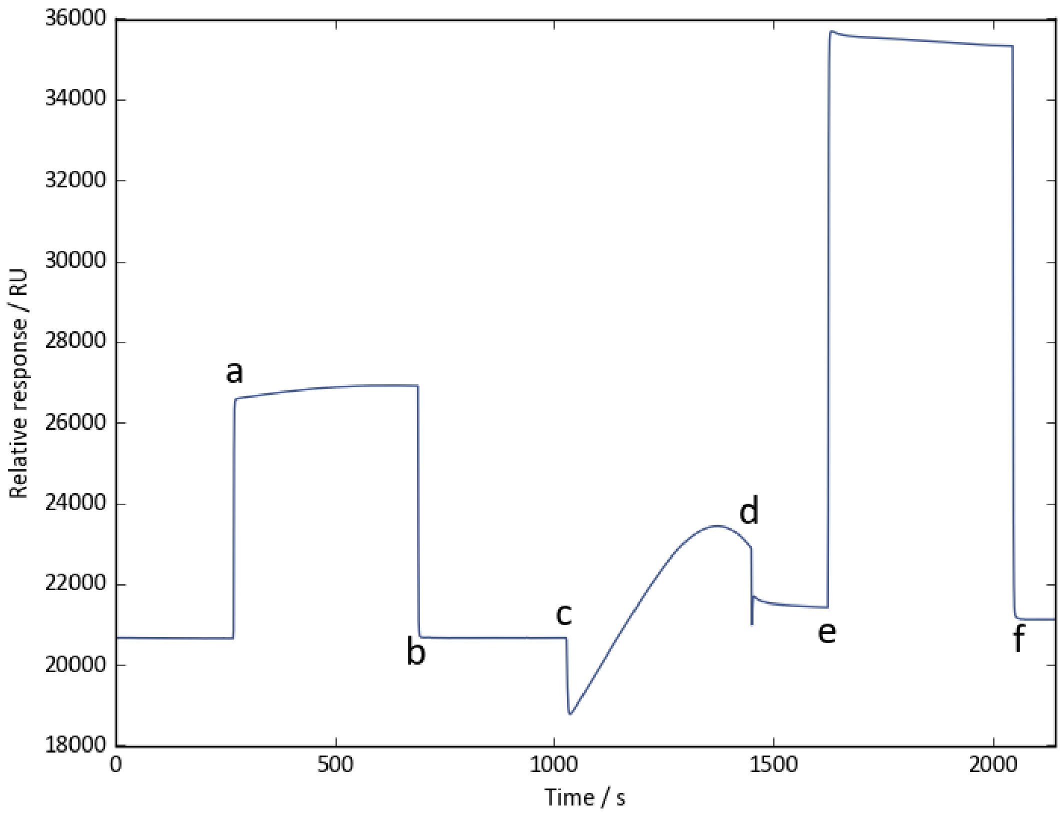

FIGURE 3.20: Sensorgram of coupling reaction. Major events in the experiment are: a) activation of surface with EDC/ NHS b) residual signal after EDC/ NHS activation c) exposure of surface to PYR1 d) stopping of PYR1 flow e) deactivation of unreacted NHS-esters with 1 M ethanolamine f) immobilised protein SPR response. The spike at point d corresponds to an air spike which should not affect the coupling process.

Adjustment	Baseline	Activated Surface (b)	PYR1 + Residual NHS-Ester (e)	Immobilised PYR1 (f)
Raw response	20647.4 ± 0.4	20658.0 ± 0.1	21426.1 ± 0.9	21124.4 ± 0.1
Baseline-adjusted	0 ± 0.4	10.6 ± 0.1	778.7 ± 0.9	477.0 ± 0.1

TABLE 3.1: Summary of response values obtained at key points in the PYR1 immobilisation process.

Ligand Binding Experiments

With PYR1 successfully bound in workable quantities, the SPR response of the chip when exposed to solutions of ABA and eW5 was studied to probe for protein-ligand interactions.

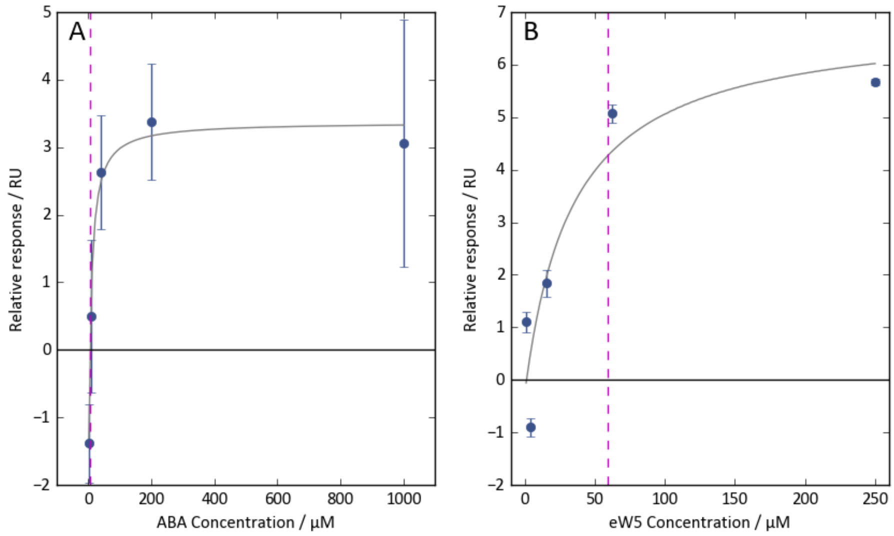

FIGURE 3.21: A) Plot of SPR response as immobilised PYR1 is exposed to varying concentrations of ABA. Error bars represent standard deviation of two experiments. The curve and dashed line represent the fit and associated K_d value as calculated by the BiaCore X100 Evaluation Software. B) Plot of SPR response as immobilised PYR1 is exposed to varying concentrations of eW5. Error bars represent the standard deviation of SPR signal around each data point collected with a 5 s window. The curve and magenta dashed line represent the fit and associated K_d value as calculated by the BiaCore X100 Evaluation Software.

Compound	K_d / µM	X^2 / RU^2
ABA	7.1	1.88
eW5	59.3	0.53

TABLE 3.2: Summary of K_d values calculated from SPR experiments.

The magnitude of SPR response was low, but a concentration-dependent change in SPR response was observed for both ligands.

3.2.4 Thermal Shift Assays

The Thermal Shift Assay (TSA) is an established technique for studying protein-ligand interactions. Protein-ligand complexes typically have a higher degree of stabilising internal interactions compared to free protein. This usually translates to an increase in the thermal denaturation temperature of a protein with increasing ligand concentration.[7–9] The midpoint of the observed thermal denaturation step (T_m) is used throughout TSA experiments in this project as a metric through which sample thermal stability is compared (see Figure 3.22).

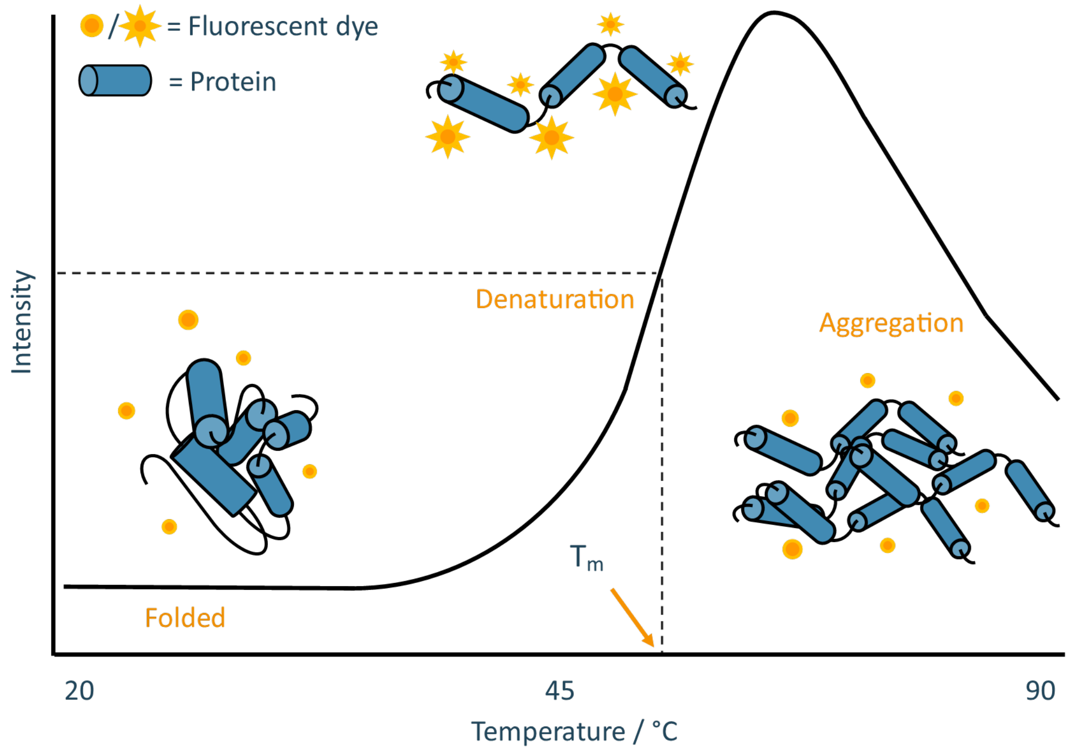

FIGURE 3.22: Annotated example of data collected in a typical TSA experiment with major regions of the TSA curve explained.

eW5 was assayed with PYR1 to check for a binding response. ABA was selected as a positive control and DB1 was used as a negative control. All conditions gave topologically similar TSA curves that could be numerically differentiated by NAMI TSA analysis software[8] to obtain T_m values (see Figures 3.23 and 3.24A for representative results).

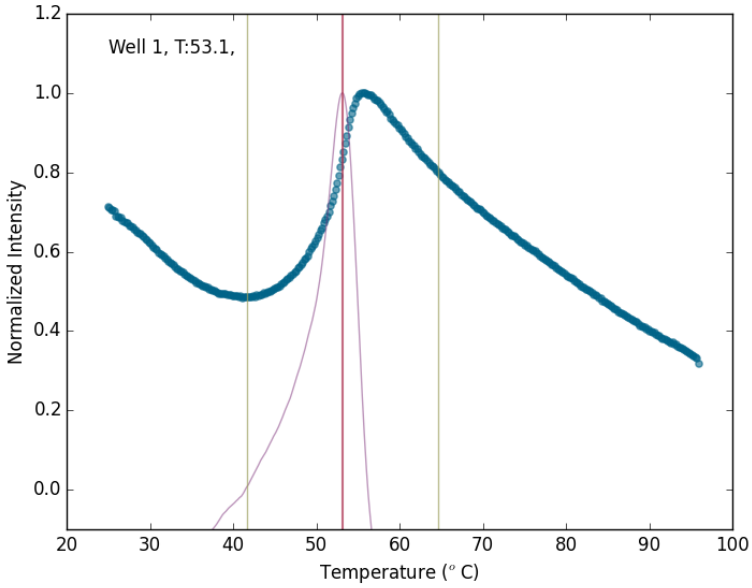

FIGURE 3.23: Example data of the fluorescence observed from a single well (PYR1 with 500 µM ABA) during a TSA experiment. Normalised fluorescence values are depcited with blue dots, the first derivative plot is depicted with a purple line. The T_m value is shown with a vertical red line and the window used to calculate the T_m value is bounded by green vertical lines.

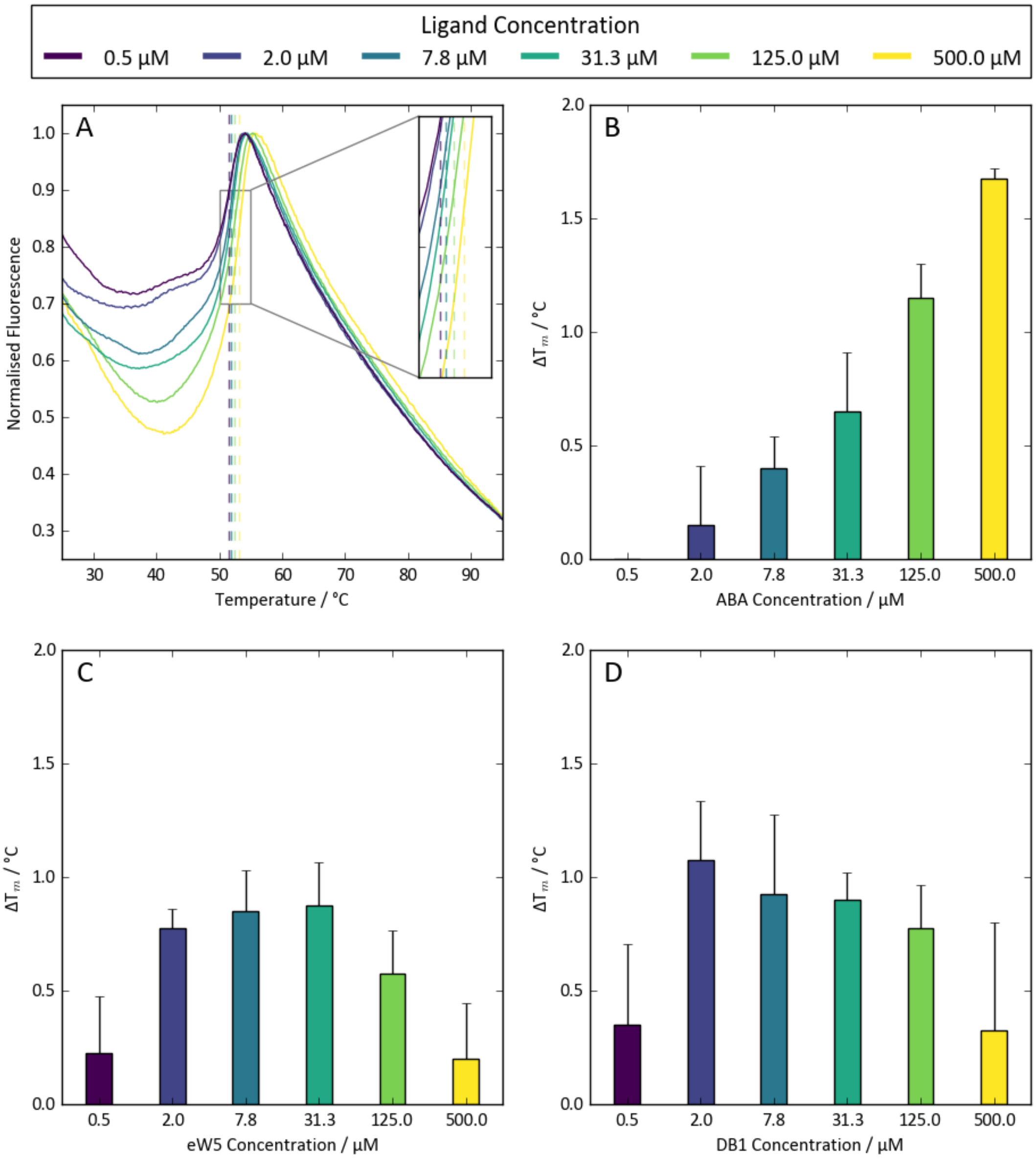

FIGURE 3.24: A) Representative data from DSF experiments with PYR1 and potential ligands. Curves shown represent TSA curves produced with PYR1 in the presence of varying concentrations of ABA, with each curve's associated T_m as calculated by NAMI displayed as vertical lines. B) Summary of T_m value increases obtained with PYR1 in the presence of different concentrations of ABA. C) Summary of T_m value increases obtained with PYR1 in the presence of different concentrations of eW5. D) Summary of T_m value increases obtained with PYR1 in the presence of different concentrations of ABA. Error bars displayed in all figures represent standard deviations of two biological replicates. ΔT_m values are calculated relative to a DMSO control T_m of 51.4 ± 0.2 °C (standard deviation of four technical replicates).

eW5 Derivative TSAs

A set of eW5 derivatives produced for a previous project[1] (Figure 3.25) were obtained and used in a series of TSA studies with PYL family proteins. Like eW5, the derivatives are naphthalenesulfonamides, with the 'AC' series differing from the 'A' series by a chlorinated naphthalene ring at the 5 position. The series was originally produced as derivatives of W5, a plant calmodulin inhibitor. eW5 was originally produced in this series and designated A1. TSA experiments were performed this compound series and the purified PYL family proteins to check for any thermal stabilisation that might indicate complexation. Figures 3.26-3.29 show the ΔT_m values of other PYL family proteins in the presence of differing concentrations of each compound in the series alongside eW5 and ABA.

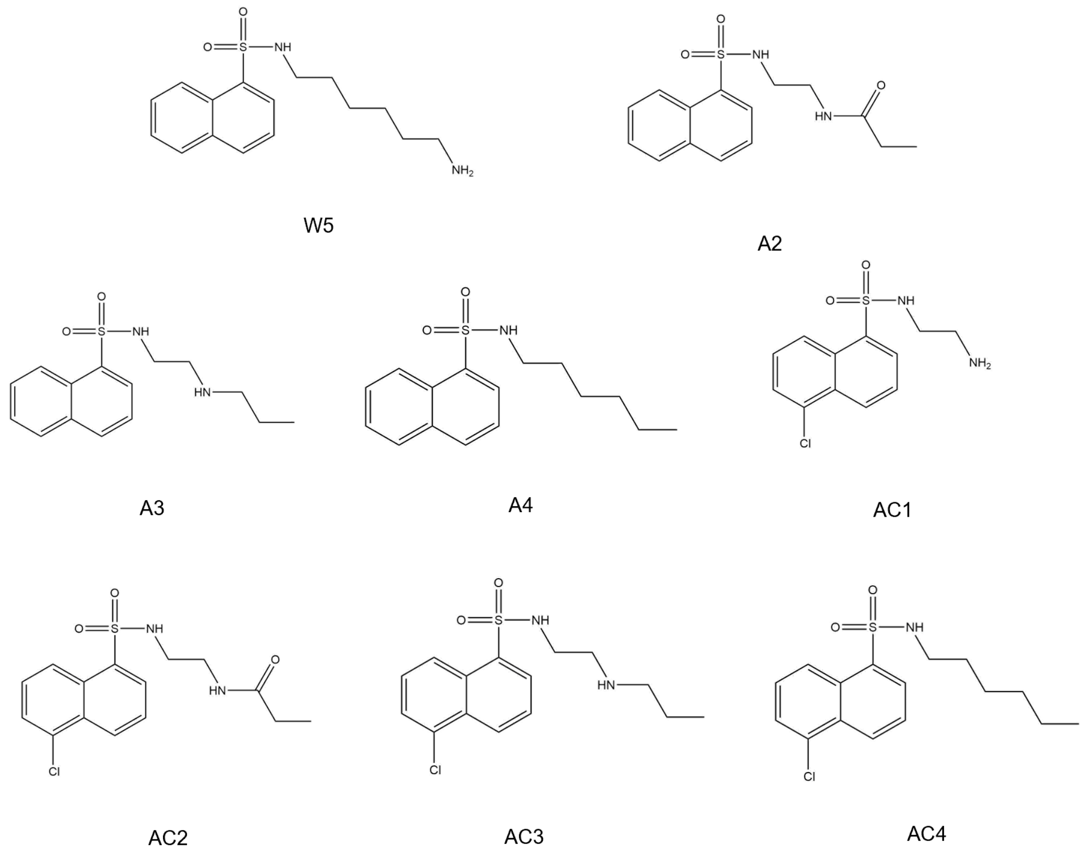

FIGURE 3.25: Series of ew5 derivative compounds. The A series of compounds is linked to the AC series by chlorination of the naphthalene ring at the 5 position.

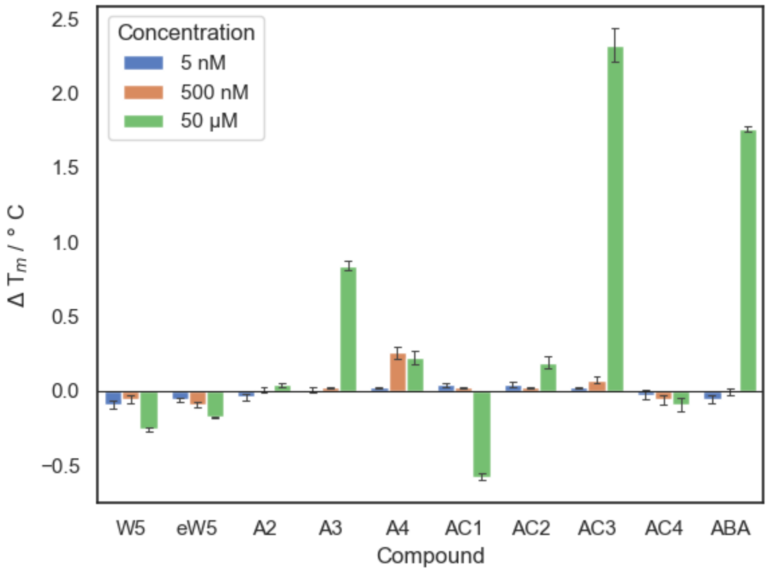

FIGURE 3.26: TSA results of PYL1 with eW5 derivative compounds. Error bars represent the standard error of 2 biological replicates.

PYL1 displays several interesting changes in T$_m$ with the A/AC series of compounds. eW5 shows a weak reduction in T$_m$, but does not display any significant evidence of interaction. A3 and AC3 elicit a large increase in T$_m$ at high concentrations- slightly more even than ABA- which suggests binding in some capacity.

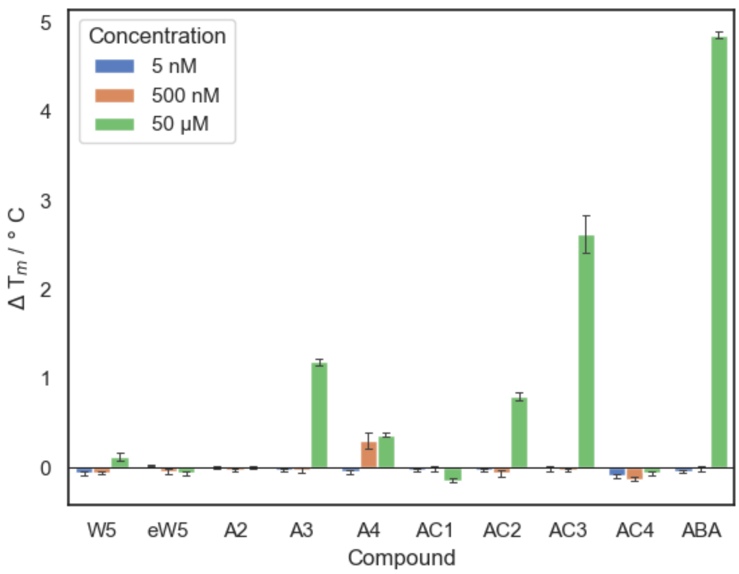

FIGURE 3.27: TSA results of PYL2 with eW5 derivative compounds. Error bars represent the standard error of 2 biological replicates.

PYL2 behaves similarly to PYL1, with little evidence of interaction with eW5 and AC1 but a noticeable stabilisation in the presence of A3 and AC3.

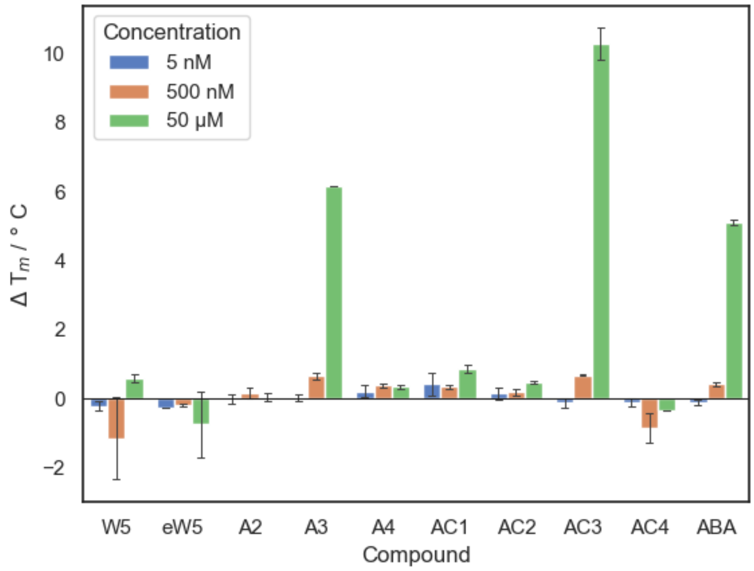

FIGURE 3.28: TSA results of PYL5 with eW5 derivative compounds. Error bars represent the standard error of 2 biological replicates.

PYL5 continues the trend of thermal stabilisation in the presence of A3 and AC3, with an exceptionally large increase in T_m. Of the PYL family proteins tested, apo-PYL5 displayed the lowest T_m but reached comparable T_m values in the presence of A3 and AC3.

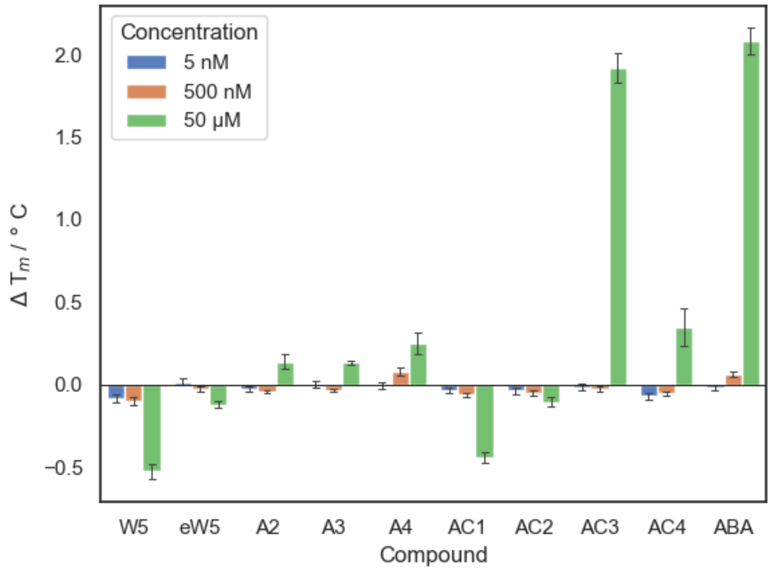

FIGURE 3.29: TSA results of PYR1 with eW5 derivative compounds. Error bars represent the standard error of 6 biological replicates.

In contrast to other members of the PYL family discussed previously, ligand A3 does not show a significant thermal stabilisation in complexation with PYR1, which may suggest that the chloro in AC3 has a more significant contribution to binding affinity for PYR1 than for other PYL family members.

Both PYL3 and PYL4 displayed melt curves that were unsuitable for T_m determination, likely due to a weak interaction with the fluorescent dye or an intrinsic structural instability in the assay conditions. Figure 3.30a shows a representative example of the TSA curve topology encountered with PYL3 and PYL4, which can be contrasted to the clear sigmoidal melt transition found in Figure 3.30b generated by a sample containing PYL5.

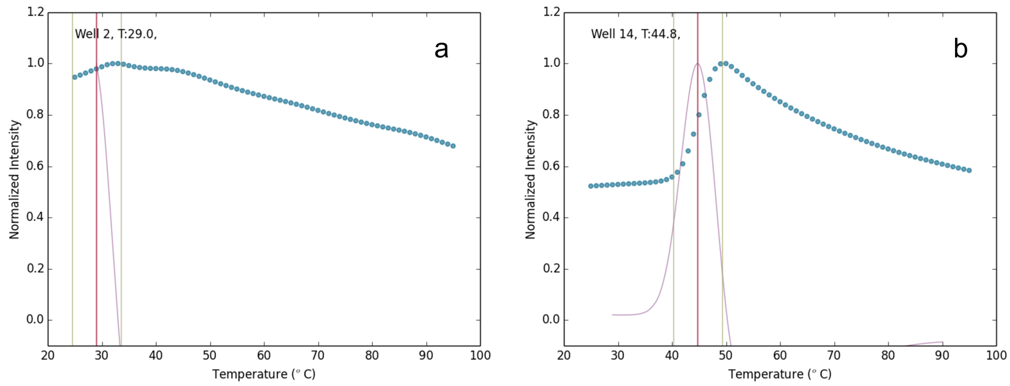

FIGURE 3.30: Representative examples of normalised fluorescence (blue) recorded during a TSA with a) PYL4 and b) PYL5. This data was processed using NAMI. Green vertical lines indicate the window that was automatically selected for analysis by NAMI, with a smoothed first derivative in purple and the proposed T_m value given by the red vertical line.

3.3 Discussion

3.3.1 Protein Production

PYL family proteins were successfully purified by HisTrap IMAC with yields between 0.5 mg and 70 mg per litre of cultured *E. coli*. This range of obtainable protein within the family correlates with protein thermal stability encountered in TSA experiments; the lesser-yielding PYL proteins (PYL3 and PYL4) displayed significantly lower T*m* values and were observed to precipitate significantly faster in storage than higher-yielding PYLs.

3.3.2 Ligand Affinity Studies

Microscale Thermophoresis

Microscale Thermophoresis (MST) provides insights regarding a molecule's size, charge and hydration shell. It relies on thermophoretic movement, the directed motion of molecules along a thermal gradient. A molecules thermophoretic affinity is affected by its size as well as its affinity to its surrounding medium, meaning that protein-protein interactions or ligand binding events can be analysed through this technique.

Neither ligand was present in concentrations high enough to saturate the MST response due to solubility limitations, meaning that curves could not be fitted to the ligand series and no quantitative measures of binding affinity were obtained. This suggests that, if binding occurs, dissociation constants would be, at best, in the micromolar regime. Nevertheless, Figure 3.17B and Figure 3.17C show correlations between MST response and the concentrations of ABA and eW5, which is indicative of a binding interaction.

It is interesting that ABA has a negative response on thermophoretic mobility while eW5 has a positive one. While the reasons for this is unclear, potential explanations might include that eW5 binds to a different site than ABA in PYR1, or that one ligand has a more pronounced effect on the PYR1 dimerisation equilibrium.

Similar examples of differing thermophoretic mobility changes have been discussed in the literature. For example, in the case of β-lactamase TEM1 binding to the β-lactamase inhibitory protein BLIP, the thermophoretic mobility of TEM1 can change positively or negatively with varied pH, ionic strength or through single amino acid substitutions in BLIP.[75]

Surface Plasmon Resonance

Affinity Capture During protein capture, the rapid decrease in SPR response upon the cessation of PYR1 exposure could be attributed to monomerisation of captured PYR1. It is possible that the strength of interaction between the His_6 tag and the Ni-NTA sensor chip surface is stronger than that of the PYR1 homodimer. Therefore, as the system continues to flow after PYR1 exposure stops, PYR1 could monomerise if only one half of the dimer is associated to the sensor chip through its tag. Regardless of the underlying mechanism, the rapid decrease in response could produce significant errors in binding and kinetic assays since the response of small molecule binding is generally very small compared to macromolecular processes such as monomerisation. Because of this, affinity capture was not explored further as an immobilisation technique for PYR1.

Ligand Binding Although a literature K_d value of the PYR1-ABA complex could not be obtained, the experimental value of 7.1 µM is close to the value of 7.7 µM reported for the PYL3-ABA complex by Zhang et al. in 2012.[76] Lower K_d values (corresponding to a higher affinity) have been reported for PYL5, PYL8 and PYL9: 1.1, 0.7 and 1.0 µM, respectively.[40, 54] However, as PYL5, PYL8 and PYL9 are monomeric while PYL3 is dimeric, it is possible that the quaternary structure of PYLs is a factor in their ABA affinity making the comparison between PYR1-ABA and PYL3-ABA more meaningful.

The SPR response of PYR1 with eW5 was small in magnitude, with a calculated K_d value of 59.3 µM. Solubility issues meant that high concentrations of eW5 were difficult to produce in buffers compatible with PYR1, further limiting the usefulness of SPR experiments.

Thermal Shift Assays

TSAs can be used to study both favourable protein buffer conditions and the presence of protein-ligand interactions. Generally speaking, a change in protein T_m of around 0.5 °C can be considered significant as a qualitative benchmark of potential ligand interaction.

The TSA studies using the A/AC series of compounds provide further evidence that the interactions, if any, between PYL family proteins and eW5 are weak in nature. DB1 was abandoned as a negative control compound in extended PYL family experiments as TSA results suggested that the lack of a terminal amine group did not preclude protein thermal stabilisation. Large thermal stabilisation was observed with A3 and AC3 amongst all surveyed PYLs, with the largest δT_m achieved by AC3 in all cases.

The additional stability conferred with A3/AC3 compared with A4/AC4 is indicative of a potentially significant interaction with the secondary amine on the aliphatic portion of the compounds. The presence of a putative interaction with A3/AC3 when compared with eW5/AC1 could either be attributed to a larger hydrophobic surface to make contact with a binding pocket, or an increase in nucleophilicity of the secondary amine through the inductive effect. The latter hypothesis is further supported by reduced stability in A2/AC2, as conjugation with the amide oxygen would draw electron density away from the amide nitrogen and affect the geometry of the chain, which would lead to a reduction in protein-ligand interaction, as observed.

3.4 Conclusions and Future Work

The work performed as part of this chapter suggests that eW5 potentially displays weak interactions with some members of the PYL family. However, the strength of the interactions observed is not sufficient to suggest that the PYL family are the only, or even the primary, target of eW5. Since PYL family proteins interact with PP2Cs and ABA in a ternary complex when influencing ABA signalling, it is possible that the presence of both proteins from the ternary complex is necessary for a significant binding interaction to be observed. An alternative approach, detailed in Chapter 4 was adopted to ascertain the effects of eW5 on a whole-organism basis.

Despite a lack of promising interactions observed with eW5 itself, derivative compounds such as AC and AC3 show strong stabilisation of PYL family proteins in thermal shift assays which is often indicative of ligand binding with good affinity. These compounds do not display the same *in vivo* growth-enhancing effects as eW5[1], but their interaction with PYL proteins suggests that eW5 derivatisation could lead to novel small molecule effectors of the ABA pathway.

Chapter 4

An *Arabidopsis* Forward Genetic Study with eW5

4.1 Introduction

Biophysical evidence discussed in Chapter 3 did little to support the case of eW5 interacting directly and specifically with PYL proteins in the ABA pathway. To provide a more comprehensive analysis of potential eW5 targets, a forward genetic approach was adopted using mutant *A. thaliana* lines. Results described in Sections 4.2.1 and 4.2.2 were obtained collaboratively with another PhD student, Jonathan Reuven[77].

4.1.1 Growth Control in *Arabidopsis thaliana*

The signalling pathways that control *Arabidopsis* growth are many in number and varied in function. Plant growth is modulated by a complex network of signalling hormones that are capable of influencing each other via crosstalk.

Abscisic Acid

The action of abscisic acid (ABA) with a particular focus on PYL proteins is described in some depth in Section 3.1. ABA affects several physiological responses including stomatal closure, leaf senescence, bud dormancy, germination and growth inhibition. ABA signalling is often linked to abiotic and biotic stress responses in plants and generally results in a decrease in the rate of growth.[78]

Auxins

Auxins play a significant role in coordinating plant growth in response to environmental features. Phototropism, geotropism and hydrotropism are driven by the distribution of auxin at root and shoot tips.[79] Auxin has also been linked to plant wound responses, flower formation and fruit development.

Gibberellins

Giberellins (GAs) are a class of plant hormones that regulate a variety of developmental processes, generally as a promoter of growth. Early experiments on gibberellic acid showed its ability to rescue dwarf mutants of maize and pea, and also induce bolting and flowering in rosette species.[80, 81] Enhancement of growth caused by GA activity can be linked to the degradation of DELLA proteins, which are themselves negative regulators of growth.[82]

Jasmonic Acid

Jasmonic acid (JA), alongside its precursors and derivatives, are important regulators of developmental and physiological processes in plants. Like ABA, JA signalling is often activated in the presence of stress factors. JA signalling is most strongly linked with plant resistance, responding to external damage (for example by herbivores) or by pathogen infection.[83]

4.1.2 Forward Genetics

Forward genetics approaches have been employed in a wide range of organisms and contexts. Forward genetics involves identifying a phenotype of interest caused by a genetic mutation, then discovering the loci responsible for the observed phenotype. Classically, this would be an incredibly laborious process involving many rounds of crossbreeding to map a gene to a specific chromosome by observing inheritance patterns with other previously mapped genes. Thanks to modern sequencing techniques, this mapping of a mutation of interest has been reduced to a (still pretty laborious in the case of this chapter) process of generating populations of mutant lines displaying the phenotype of interest and sequencing their extracted DNA. The

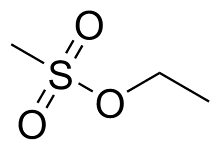

FIGURE 4.1: Structure of EMS.

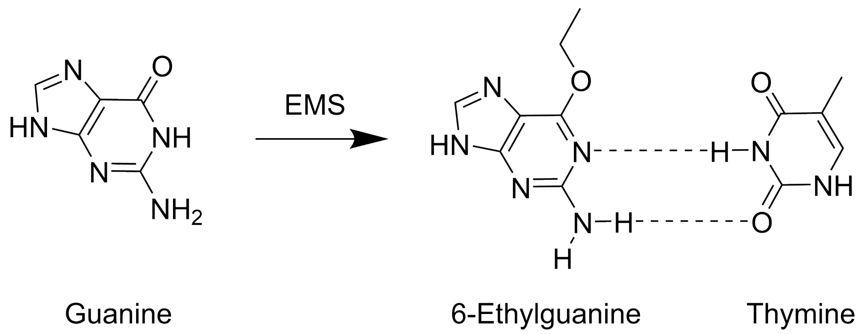

FIGURE 4.2: Schematic of O^6-ethylguanine formation and subsequent pairing with thymine.

scalability of Next Generation Sequencing (NGS) has allowed for a significant increase in throughput for forwards genetics projects.[84, 85]

Ethyl Methanesulphonate Mutagenesis

Ethyl methanesulphonate (EMS) is a mutagen that induces random mutations in DNA through nucleotide substitution. EMS is an ethylating agent, generally causing G:C to A:T point mutations by alkylating guanine to form O^6-ethylguanine (see Figure 4.2.[86] O^6-ethylguanine can pair with thymine but not cytosine, and is eventually replaced with adenine through DNA repair.

The resulting mutants can be used in loss- or gain-of-function studies and, if a missense variant is created in an appropriate position, information can be provided in the role of specific amino acid residues in protein function.

EMS Mutagenesis in *Arabidopsis thaliana*

The random nature of EMS mutagenesis can result in lethal mutations accruing in mutated organisms. A standard method of preparing mutagenised seeds with non-lethal mutations is to breed the selection of seeds that are exposed to EMS (the M_1 generation) and collect the progeny (M_2). In doing so, mutations that result in non-viable or sterile plants are not carried forward to the M_2 generation.

Lines showing a phenotype of interest are then selected from the M_2 generation. Before sequencing, the M_2 plants of interest can be crossed with a wild-type *Arabidopsis* ecotype to produce F_1 hybrids. If the phenotype of interest is still present in the F_1 hybrids, the phenotype can be seen as dominantly inherited. Whole-genome sequencing data can be collected, then analysed for single nucleotide polymorphisms (SNPs) by comparing it to a wild-type background.

Breeding to produce the F_2 generation before sequencing can result in the introduction of some false positive results, therefore it is important to follow up any genes of interest identified in the sequencing stage with further experiments to confirm the gene or genes of interest.

4.2 Results

4.2.1 Hypocotyl Assays

The major limiting factor of the forward genetic screening procedure was the throughput that was possible given limited growth chamber space. To mitigate the impact of the low throughput while still collecting reliable data, an approach was adopted that gradually filtered out seedlings that were sensitive to eW5.

Generation 1

The first stage of the forward genetics experiment was to select the individual seedlings from the entire mutant population that were smallest in hypocotyl size. Whether or not the low growth levels of these seedlings were due to an insensitivity to eW5 was not considered in the first stage. eW5-free control plates were still used to confirm the efficacy of eW5 in each batch of plates with wild-type seedlings only. Around 100 seeds from each line were sown on MS-agar plates containing eW5, then imaged and measured as described in Section 2.2.1. Due to limited growth chamber space, the lines were screened in several waves, with the 1.5 % of the population possessing the shortest hypocotyls taken forward to the next stage. 23,997 seedlings were sown and measured, with the smallest 1.5 % selected.

The seedlings selected for the next stage were transferred to peat plugs, grown to maturity and harvested to form generation 2.

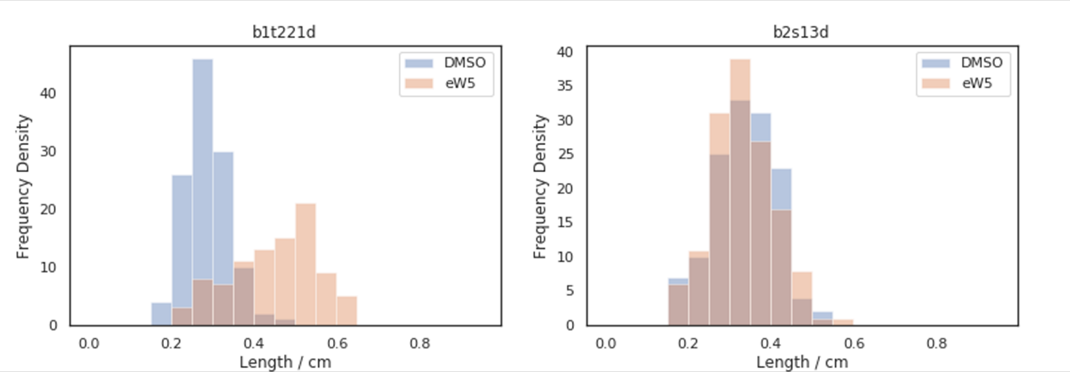

FIGURE 4.3: Histograms of hypocotyl lengths found in two mutant lines. The example on the left shows a typical eW5 growth response, with a positive effect on growth visible in the population exposed to eW5 (t-test $p = 5.82 \times 10^{-34}$). The example on the right shows minimal response to eW5 (t-test $p = 0.69$).

Generation 2

The second stage of screening involved distinguishing between mutants that have shorter hypocotyls because of an insensitivity to eW5 and mutants that are short because of other factors. Generation 2 screening was performed in several rounds to check for insensitive mutants; all plants were tested on an eW5 and DMSO control plate in the first round, with plants taken forward to the next round for retesting only if they were significantly insensitive. Insensitivity to eW5 was measured by performing an independent t-test on the hypocotyl lengths for the population grown on eW5 plates and control plates. Figure 4.3 shows two representative hypocotyl length distributions of lines that were deemed eW5-sensitive or eW5-insensitive, respectively.

An independent t-test was performed between the eW5 and control populations for each mutant line. A t-test p-value significance threshold of 0.005 was established to differentiate between responsive and unresponsive seedlings. Figure 4.4 shows the proportion of seedlings displaying poor sensitivity to eW5 versus those showing growth modulation.

The final wave of testing was performed with 43 lines that had shown minimal responses to eW5 across two tests. Figure 4.5 shows the distribution of t-test p-values and overall differences in the mean hypocotyl lengths of the eW5 and control populations. A comparison is also made between the mean hypocotyl lengths of each mutant control population and the mean hypocotyl lengths of wild-type *A. thaliana* grown in the same conditions.

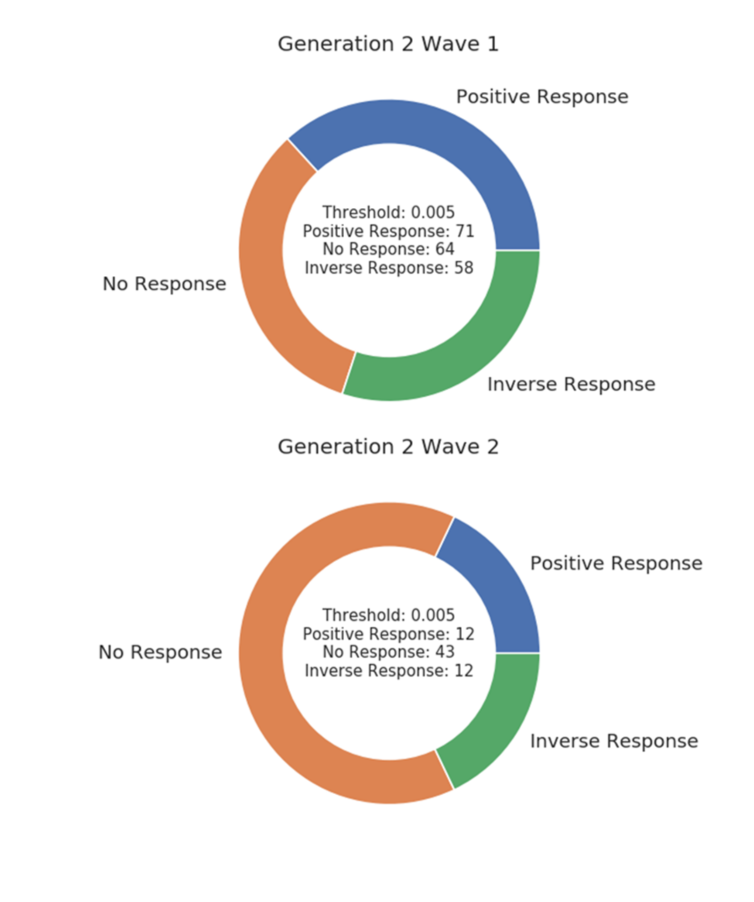

FIGURE 4.4: Doughnut plots showing the hypocotyl elongation response in each mutant line of eW5 versus a DMSO control. 'Inverse Response' describes plants that show an inhibition of growth when exposed to eW5.

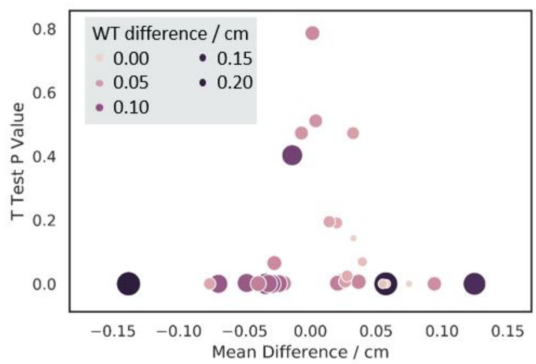

FIGURE 4.5: Plot of t-test p-values of the final wave. The six lines with the highest p values were taken forward to the next round of testing.

Common Lines	Count
1	1121
2	256
3	14
4	1
5	4
6	2

TABLE 4.1: Counts of mutated genes shared between each line

4.2.2 Mutant Crossing

Mutants that were established to be insensitive to eW5 were crossed with wild-type *A. thaliana* to produce an F_1 hybrid generation before individuals were selected for next generation sequencing. If the genes responsible for eW5 activity displayed a classical recessive phenotype, it could be expected that approximately a quarter of the progeny would continue to display the eW5 insensitivity making them appropriate candidates for sequencing. A final wave of hypocotyl elongation assays were performed, with seedlings plated only on MS-eW5-agar. For each line, the smallest 20 % of seedlings were extracted and stored in preparation for sequencing.

4.2.3 Next Generation Sequencing

Next Generation Sequencing (NGS) was performed using gDNA extracted from the final 6 crosses. Initial data processing was performed as described in Section 2.2.3.

Common Gene Counts

Gene variants observed within the 6 lines were combined into one database and examined to see which mutated genes were common between each line. The frequency of mutated genes is displayed in Table 4.1. Applying an arbitrary significance threshold of three common mutation occurrences, 21 regions of interest can be identified using this analysis. These regions of interest comprise only 5 mutations that are in coding regions, with the remaining 16 in intergenic regions. A full list of these commonly mutated regions is available in Appendix A.1.

Gene Name	Gene Description
AT5G60660	plasma membrane intrinsic protein 2;4
AT5G60640	PDI-like 1-4
AT5G60650	proline-rich receptor-like kinase
AT4G06688	myosin heavy chain-like protein
AT2G08986	hypothetical protein
AT4G06688	pseudo

TABLE 4.2: Entrez descriptions of commonly-mutated genes shared between at least 3 mutant lines.

An Entrez[87] search was performed to identify the name and function, if known, of each commonly-mutated gene. A summary of this information can be found in Table 4.2.

Two of the commonly-mutated genes encode proteins that are functionally-characterised: AT5G60660, which encodes Plasma membrane Intrinsic Protein 2;4 (PIP2;4), and AT5G60640, which encodes a protein disulphide isomerase-like protein (PDIL). The remaining proteins' functions are predicted based on homology. The intergenic regions that were commonly mutated amongst the mutant lines are listed in Table 4.3.

Allelic Frequency Analysis

As well as looking at common mutations between each line, a second method of identifying significant mutations is to filter the results by the effect the mutation has on the translated protein; whether the mutation introduces a truncation to the protein by the introduction of a stop codon or an amino acid is mutated by the introduction of a missense variant. After filtering for significant mutation outcomes, the allele frequency of each mutation can be checked to see how many times the mutation appears in the sample population versus the reference population during sequencing. Finally, read depth can be used to see how many reads of the sample population possess the mutation versus the reference population.

By filtering the data to only contain missense variants and mutations that cause a gain in stop codons, 97 mutations remained. Figure 4.6 shows the distribution of allele frequencies found in the data.

A significance threshold of 0.8 was determined from the allele frequency distribution. 10 genes of interest remained after further filtering, listed in Table 4.4

Gene Name	Gene Description
AT3G33520	actin-related protein 6
AT4G06634	zinc finger (C2H2 type) family protein
AT3G42050	vacuolar ATP synthase subunit H family protein
AT3G32980	Peroxidase superfamily protein
AT1G38131	O-fucosyltransferase family protein
AT5G32440	Ubiquitin system component Cue protein
AT4G06744	Leucine-rich repeat (LRR) family protein
AT5G36001	RING/U-box superfamily protein
AT4G06598	basic-leucine zipper (bZIP) transcription factor
AT2G10260	Ulp1 protease family protein
AT2G07771	Cytochrome C assembly protein
AT1G40104	hypothetical protein
AT4G06655	transmembrane protein
AT4G03740	hypothetical protein
AT2G07640	NAD(P)-binding Rossmann-fold superfamily protein
AT3G30705	transmembrane protein
AT1G40129	hypothetical protein
AT5G36080	Myb/SANT-like DNA-binding domain protein
At4g06634	zinc finger protein 410-like
AT5G05005	ncRNA

TABLE 4.3: Table of proteins that bookend commonly-mutated intergenic regions shared between EMS-mutagenised *A. thaliana* lines. Gene Description denotes the Entrez description of each gene. Genes without an Entrez description are omitted from this table.

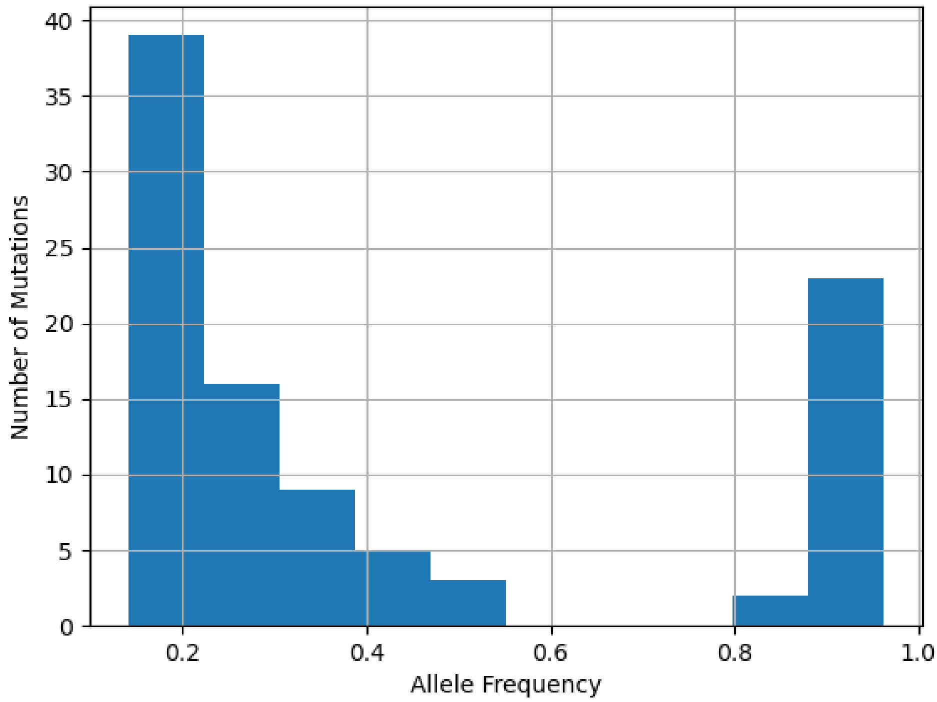

FIGURE 4.6: Histogram of allele frequencies observed in the NGS data.

alongside their Entrez gene desriptions.

TraVA Analysis

To provide contextual information on the genes of interest, a search was performed on TraVA, [88, 89] the transcriptome variation analysis database. TraVA offers expression profiles of the *Arabidopsis* genome at various developmental stages and allows differentiation of RNA-seq data based on expression localisation within the organism. Each gene of interest was profiled in TraVA to discover its relative expression level during germination stages and how much of its expression is localised in the hypocotyl (Figure 4.7).

Gene Name	Gene Description
AT1G67230	little nuclei1
AT4G35580	NAC transcription factor-like 9
AT4G15110	cytochrome P450, family 97, subfamily B, polypeptide 3
AT3G54280	ROOT GROWTH DEFECTIVE 3
AT5G19420	Regulator of chromosome condensation (RCC1) family with FYVE zinc finger domain-containing protein [Arabidopsis thaliana (thale cress)]
AT2G18630	transmembrane protein, putative (DUF677)
AT4G31880	transcriptional regulator
AT3G53960	Major facilitator superfamily protein
AT4G01570	Tetratricopeptide repeat (TPR)-like superfamily protein [Arabidopsis thaliana (thale cress)]
AT1G31555	hypothetical protein

TABLE 4.4: Entrez descriptions of genes with funcitonally significant mutations with high allelic frequency.

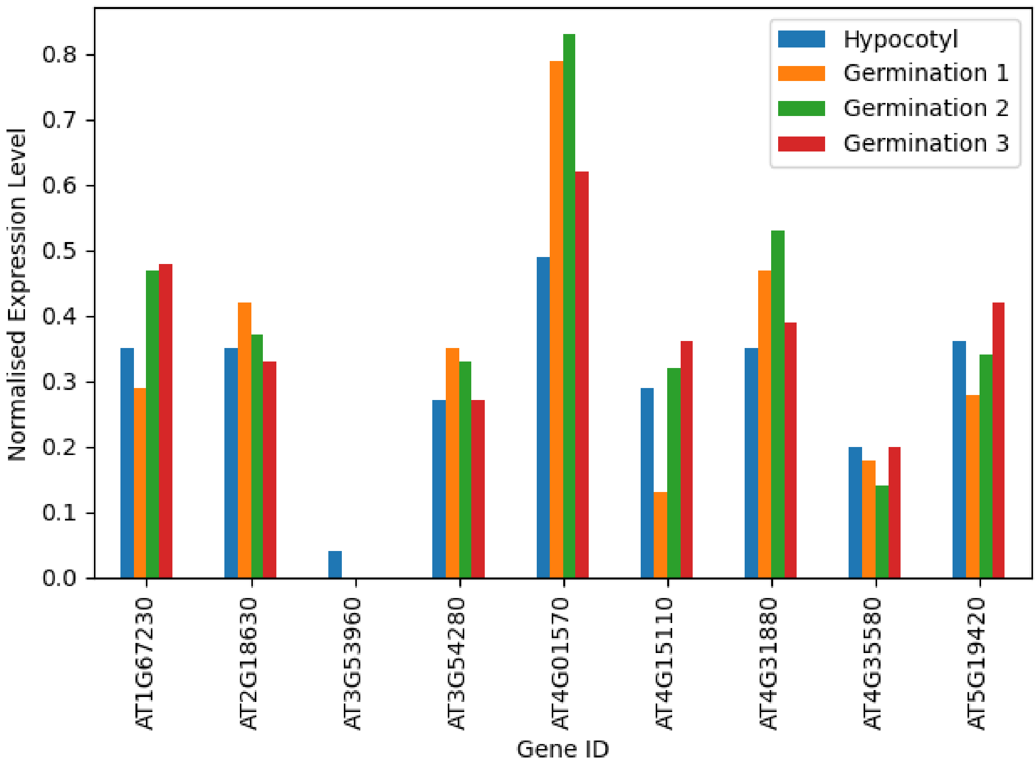

FIGURE 4.7: Relative expression profiles of each gene of interest generated using TraVA.

4.3 Discussion

Screening

EMS mutagenisis provides an efficient method of producing point mutations in *Arabidopsis*. The six candidates taken forward for NGS analysis represent mutants that are consistently insensitive to eW5 in the screening experiment.

Common Gene Counts

A fairly small selection of coding regions are commonly mutated between multiple lines. Given the random nature of EMS mutagenisis, this is perhaps to be expected.

Genes of Interest

AT1G67230 encodes LINC1, a component of SUN-protein-containing multivariate complexes also called LINC (LInker of Nucleoskeleton and Cytoskeleton) complexes, which link the nucleoskeleton and cytoskeleton by providing versatile outer nuclear membrane attachment sites for cytoskeletal filaments.[90] LINC1 also plays a role in plant immune response, facilitating pathogen-associated molecular pattern (PAMP) recognition in *Arabidopsis*. LINC1 is further associated with jasmonic acid signalling in plants, with *linc1* mutants being compromised in their ability to encode factors involved in jasmonic acid signalling and responses. Given its role in nuclear morphology and plant immune response, it is unclear how a potential small molecule interaction with LINC1 would modulate hypocotyl elongation to a significant extent. Nevertheless, jasmonic acid exerts some influence over plant growth in stress conditions, inhibiting growth in response to microbial pathogens, herbivorous insects, wounding, drought, salt stress, and low temperature.[83] Its designation as a gene of interest within this experiment could be due to a dampening of the usual growth inhibition that jasmonic acid can encourage at cold temperatures, which are experienced by every plant during the stratification process.

The function of AT2G18630 is uncertain, but transcriptomic analysis suggests that it encodes a nuclear or peroxisomal protein.[91] Transcriptomic analysis generated by TraVA shows moderate expression in the hypocotyl and during germination periods.

AT3G53960 encodes a Major Facilitator Superfamily (MFS) protein. MFS proteins are involved in the transport of a diverse set of molecules across cell membranes, and the expression of many members of the superfamily is modulated under stress conditions.[92] This gene showed poor expression in the hypocotyl and also had no detectable expression during germination stages, making it an unlikely candidate for eW5 interaction.

AT3G54280, also known as RGD3, encodes TATA-binding protein-associated factor BTAF1. This gene acts as positive regulator of the Cup-shaped Cotyledon (CUC)-Shoot Meristemless (STM) pathway in shoot apical meristem (SAM) neo-formation.[93] STM, a class I KNOX homeodomain transcription factor, influences hormone homeostasis and signalling by creating an environment of high cytokinin and low gibberellin in meristem.[94–98] Given its strong links with growth regulation in the shoot area, this could be a good candidate to take forward for in vitro experiments.

AT4G01570 displays good localisation in the hypocotyl and strong activity within the first few days of germination in Arabidopsis. It encodes a Tetratricopeptide repeat (TPR)-like superfamily protein. The TPR domain is found in a diverse range of proteins, but predominantly exists as an interface for protein-protein interactions.[99] It is currently unknown which proteins interact with the protein encoded by AT4G01570; but *Arabidopsis* proteins possessing TPR domains have been shown to play a role in ABA signalling, influencing drought and cold tolerance[100, 101]

AT4G15110 encodes CYP97B3, a cytochrome P450 putatively linked to xanthophyll production[102]. Given the reduction in light exposure that the seedlings were subjected to in this experiment, it is possible that a mutation in genes involved in accessory pigment production could be detrimental to their growth and cause a reduction in hypocotyl elongation that is decoupled with eW5 activity.

The protein encoded by AT4G31880 is AtSAG, a protein containing a midasin homologue 1 (MDN1) domain. This protein has been shown to play a role in seed germination and development. AtSAG is a negative regulator of ABA signalling during seed germination and development that is dependent on ABI3 and ABI5.[103]

AT4G35580 encodes a calmodulin-binding NAC protein (CBNAC). The NAC family is one of the largest families of plant transcription factors, playing a role in both developmental and stress responses.[104]

AT5G19420 has been identified as RCC1-like domain 3 (RLD3). RLD3 is part of a family of proteins that interact with AtLAZY1/LAZY1-LIKE (LZY) proteins to coordinate a gravitropism response by modulating the flow of auxin.[105] RLD proteins coordinate the maintenance of a specific growth angle in roots and shoots known as gravitropic set point angles, which contribute to efficient acquisition of water and nutrients.[106, 107].

4.3.1 Intergenic Regions of Interest

The effects of mutations at intergenic regions are significantly harder to predict than those previously discussed. Nevertheless, the regulatory role that intergenic regions can play in protein expression is significant, and genes adjacent to commmonly-mutated intergenic regions are considered in this section. Table 4.3 denotes genes that were present at either side of commonly-mutated intergenic regions.

Several of these genes are uncharacterised or hypothetical proteins. A number of DNA-binding motifs, including zinc finger and bZIP, were identified among genes of interest, as well as motifs commonly encountered in protein-protein interactions such as leucine-rich repeats. Of all the genes that bookend the intergenic regions observed in this study, the most interesting from a functional perspective is perhaps AT5G36080, which encodes a MYB/SANT-like DNA-binding domain protein. MYB family proteins have been associated with the promotion of lateral root growth in *Arabidopsis* through an interaction with PYL8.[108] Given their role within growth regulation, albeit at the root instead of the hypocotyl, the identification of this gene as one of significance within this study could suggest that the activity of eW5 plays a small role in a hugely complex network of phytohormonal growth regulation.

4.4 Conclusions and Future Work

The work performed in this chapter has identified several promising candidates for eW5 target proteins, notably RGD3. Nevertheless, it is important to remove any false positives from the pool of identified genes. Further genetics experiments to confirm the identity of the genes of interest could involve knockout studies to confirm a loss of eW5 sensitivity. Affinity studies to probe the interaction between eW5 and isolated proteins of interest can also be performed if folded recombinant protein is obtainable.

Chapter 5

Applying Machine Learning to Hypocotyl Elongation Analysis

Within its binomial, *Arabidopsis thaliana* contains the name of Thalia, the ancient Greek muse of comedy. Any researchers unfortunate enough to regularly perform *A. thaliana* hypocotyl elongation assays will understand that the processing of data produced in this assay is no laughing matter. Hypocotyl measurement is typically performed manually using scanned images and, despite the existence of helpful programs such as ImageJ[21] to aid in measurement, often takes a significant amount of time and effort to achieve.

Studies of EMS-mutagenisis in *Arabidopsis thaliana* have calculated each EMS-mutagenised M_1 line possesses at least 700 mutations, meaning fewer than 50,000 M_1 lines are needed to have a 95 % chance of finding a mutation in any given G:C base pair in the genome.[109] With robust studies therefore often requiring tens of thousands of seedlings measurements, automating the processing of hypocotyl measurements would save a significant amount of labour for plant researchers. In an attempt to automate the measurement and downstream processing of hypocotyl elongation assay data, a program was developed utilising a combination of image processing algorithms and a region-based convolutional neural network (RCNN).

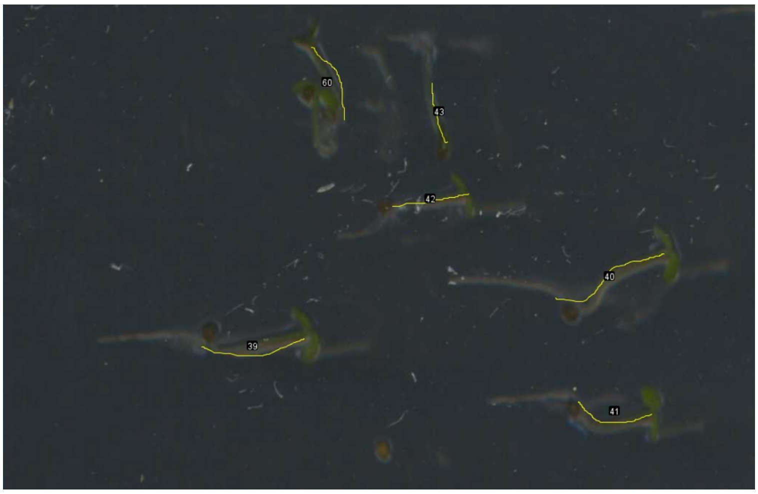

FIGURE 5.1: Example image of seedlings in the process of manual hypocotyl annotation.

5.1 Introduction

The release of AlphaFold[13] and RoseTTAFold[14] to open source in 2021 sent huge shockwaves through the structural biology community. For many users, the programs enabled the first taste of AI-driven experimentation and allowed physically-realistic prediction of protein structures that were previously impossible to characterise using experimental techniques such as crystallography, cryo-EM, SAXS and NMR. In general, the number of applications powered by machine learning algorithms has exploded in recent years, driving huge advances in fields as diverse as genomics, drug design and biological image analysis (computer vision).

Computer vision algorithms have become increasingly popular amongst biologists routinely working with images. The automated subject identification that computer vision algorithms allow can be easily implemented as an initial processing step of automated image analysis pipelines, driving advances in the speed and ease of data processing. Additionally, by implementing the algorithms in programs with simple graphical user interfaces, these powerful techniques can be accessed by users who do not have a strong computational background.

Accompanying the rising ubiquity of machine learning programs is an increasing ease in which customised machine learning models can be implemented and trained from a programmer's perspective. Several machine learning libraries are freely available within the Python ecosystem, including TensorFlow [110], Keras

[111] and PyTorch [36], which provide varying balances of training performance, ease of use and low-level customisation options.

Current computer vision methods to aid in the analysis of plant images offer good segmentation performance when compared with human measurements[112] and can provide analysis of hypocotyl morphology during photomorphogenesis[**Wang2009hypotrace**]. However, the training datasets that these models are based on rely on the ordered placement of seedlings with a high degree of directional uniformity. The applicability of existing models to 'messy' data that is commonly encountered- overlapping plants that are often curled instead of straight- is reduced and the rate-limiting step of collecting measurements shifts to the ordering of seedlings before imaging.

5.1.1 Computer Vision Tasks

Cats often get a raw deal when they become involved in science. Physicists are taught to mourn (or not mourn) the death (or not-death) of Schrödinger's famous hypothetical cat, and many more unfortunate felines have been sacrificed in a more concrete sense on the altar of progress over the years. Much of the early understanding of how humans and animals process visual information stems from neurophysiological research conducted on cats in the 1950s and 1960s[113]. The discovery of a hierarchical processing of visual stimuli, first detecting simple features like edges and feeding that information into more complex features like shapes, was a landmark finding in neurophysiology. Interestingly, this model of visual processing presents a useful basis for understanding computer vision and has inspired many advances in computer vision algorithms.

The field of computer vision can trace its roots back to the early 1960s, with early projects seen as a stepping stone towards artificial intelligence. Early applications involved trying to extract 3D geometrical information from a 2D perspective of simple polyhedra[**Roberts1963**]. By the 2000s, computer vision tasks for static images had advanced from deciphering simple shapes to dealing with real-world images. Some of the most common modern computer vision tasks can be listed (and explained with the enduring help of cats) as:

- **Object Classification:** Is this an image of a cat or a dog?

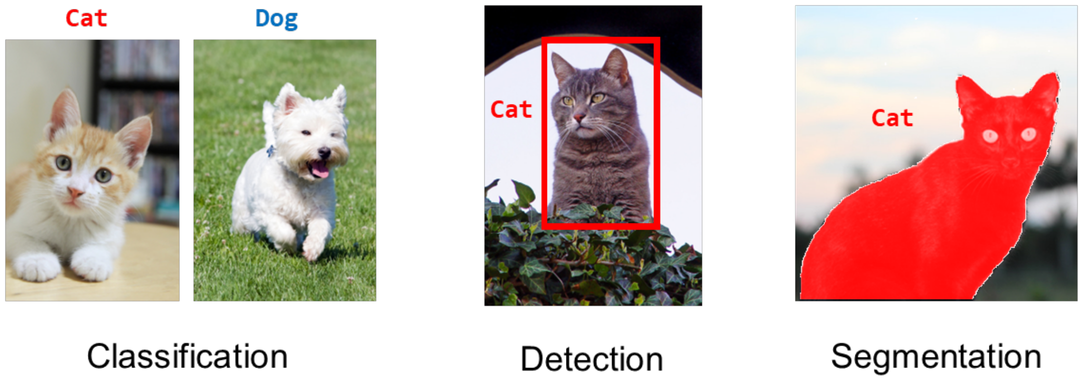

FIGURE 5.2: Examples of common computer vision applications.

- **Object Detection:** Where are the cats in this image?
- **Object Segmentation:** Where are the pixels belonging to cats in this image?

The rapid advances in computer vision applications have been driven by both the success of modern algorithms (including neural networks and deep learning architectures) and the increasing availability of quality labelled datasets to test and train novel computer vision workflows. The advent of the ImageNet dataset in 2009 revealed a huge database of hand-annotated images, today providing over 14 million annotated images, with over 1 million of those images also containing bounding box information.[114]

5.1.2 Processing of Training Images

Image Standardisation

It is important to standardise any dataset used to train and operate machine learning algorithms; without a regular format for data entry and analysis, the predictive power of the algorithms are severely hampered. Each machine learning model requires a specific format of data that is appropriate for its application, with the basic requirements for object detection being training images and a set of bounding box coordinates, each denoting the location of objects of interest within each training image, as seen in Figure 5.8. For segmentation applications, training data will instead comprise images and an image mask denoting which pixel belongs to which class of object. A coloured image of $m \times n$ pixels is provided as a $3 \times m \times n$ array displaying intensity values for red, green and blue channels.

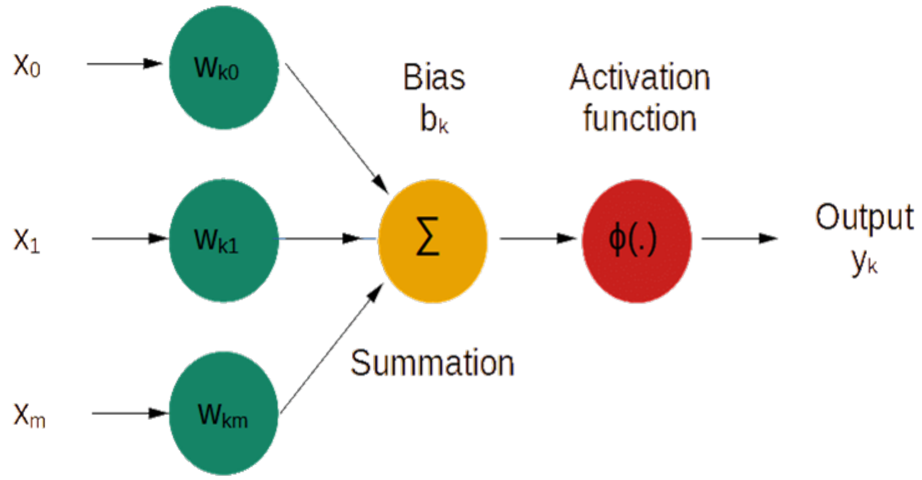

FIGURE 5.3: Diagram of an artificial neuron.

Data Augmentation

The curation and labelling of training datasets is often the most labour-intensive and time-consuming aspect of a machine learning project. When creating a new dataset, images must be manually labelled as accurately as possible to avoid introducing errors into the training dataset. Nevertheless, this is a critical step in machine learning that cannot be avoided if a robust model is to be trained. Several techniques exist to improve the value of labelled data in the training process and are known as data augmentations. These involve transforming the training images in various ways before using them to train a model. Examples of transformations used to augment image training data include padding, rotating, re-scaling, flipping, cropping, zooming and adding noise. Data augmentation contributes to the training of robust machine learning models that are less prone to overfitting.[115]

5.1.3 Computer Vision Neural Networks

Neural networks are a subtype of machine learning algorithm that are named because of their organisational parallels with the human brain. The basic compositional unit of a neural network is an artificial neuron, which consists of several features outlined in Figure 5.3.

As a general definition, an artificial neuron k takes $m+1$ inputs, x_0 through x_m, which are given weights w_{k0} through w_{km}. The weighted inputs are summed and passed through an activation function ϕ. Thus, the output of the kth neuron is:

$$y_k = \phi(\sum_{j=0}^{m} w_{kj}x_j + b_k)$$

The weighting for each input is randomly initialised in an untrained model, then refined during model training. In practice, training a neural network from scratch is a computationally expensive and time-consuming process. For many applications, a pretrained model is used as a starting point and refined to suit the target application.

The perfect case of an abundance of accurately labelled training data is often not a reality for many machine learning projects. However, pretrained models can be refined to suit a surprising range of targets models in a process known as transfer learning [116]. By starting with weights that are already relevant to image data, the number of training iterations required to produce a suitable model can be reduced. One biologically-relevant example of transfer learning is the training of a model that classifies electroencephalographic waves that was initially trained to classify electromyographic waves.[117]

Artificial neurons are typically aggregated into, and connected by, layers. Each layer performs a transformation to its input, with multiple patterns of connection possible between each layer. In fully connected layers, every neuron of the previous layer connects to every neuron of the next layer. Pooling operations can also occur between layers, where a group of neurons in one layer feed into a single neuron in the next.

5.1.4 Operations In Neural Networks

The types of neural networks used in this chapter are specialised networks known as convolutional neural networks (CNNs). CNNs possess a range of possible functions and transformations in each of their layers, the most common of which are outlined below.

Connections between layers

Layers in neural networks can be connected in a variety of different ways, each having different effects on both the state of the transformed input and the overall complexity of the neural network. Two of the most common layer types, distinguished

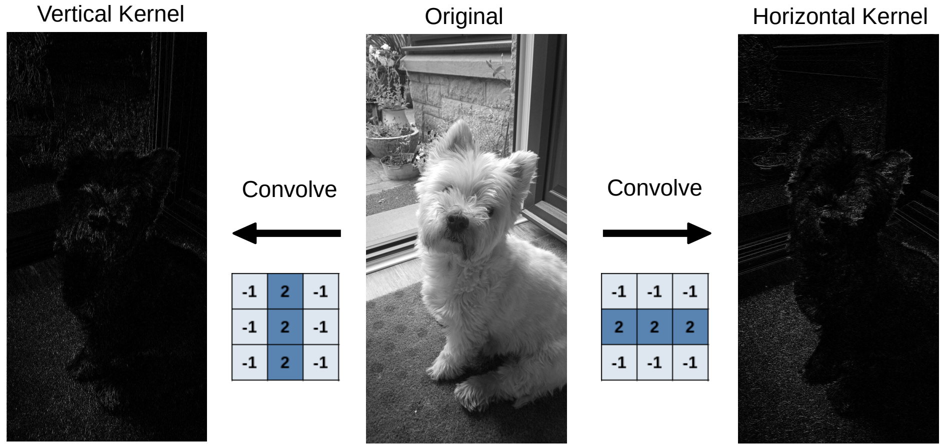

FIGURE 5.4: Examples of convolution transformations on a grayscale image to extract different directions of lines.

by their connectivity, are fully connected layers and convolutional layers. All of the neurons in a fully connected layer are connected to every neuron of the previous layer. Convolutional neural networks are particularly popular in computer vision and language processing machine learning applications. Convolutional layers effectively perform a sliding dot product between an input array and a small transformation array known as a kernel or filter. Depending on the values of the kernel used in convolution, the output image, also known as a feature map, will have certain features filtered out and highlighted. Figure 5.4 shows the feature maps that result from some simple convolutions with a 3×3 pixel kernel.

A lot of flexibility can be applied to convolutions, such as varying the dimensions of the kernel, how small the steps within each convolution are (stride) and the dimensions of the output (one such way to achieve this is adding padding, rows and columns of zeros to the outside of the input array, before convolution to make a larger output array). By making these variations, a large set of features can be extracted and highlighted in training images.[118, 119] The importance of each feature in performing the desired task for the CNN can then relate to the weighting value that is determined during training.

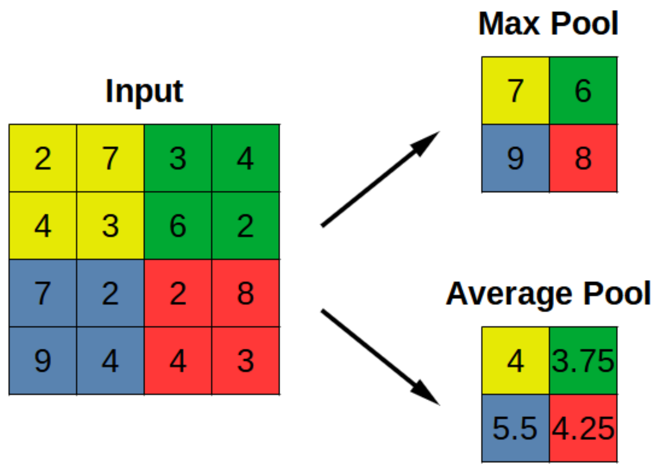

FIGURE 5.5: Examples of pooling transformations with a 2×2 pixel kernel.

Pooling

Pooling is a method used to downsample the number of features that are passed onto subsequent layers in a neural network, in effect summarising higher level features. Pooling operations involve taking an input array and separating it into smaller regions of a predetermined size, often 2×2 pixels. Depending on the type of pooling operation desired, the regions are processed in different ways to produce a new feature map. Figure 5.5 shows some common pooling examples using a 2×2 pixel pooling kernel, max pool and average pool. Max pool operations take the largest value in each region and carry that forward to the corresponding value of the pooled feature map, while average pool operations take the mean.

Activation Functions

The first operation for each artificial neuron in a neural network is to apply a transformation to each its inputs by applying its associated weight. Artificial neurons then need to be able to distinguish between the values in the resulting feature map that should propagate to the next layer, and those that are insignificant. Activation functions are a second transformation that is performed after summing all weighted inputs that filters the resulting feature map. Activation function filtering can be seen as analogous to threshold potentials in biological neurons, where a certain membrane potential must be reached before a signal is propagated. During the training process of a neural network, the derivative of each artificial neuron's activation function is calculated and used to readjust each of its weights in a process called backpropagation.

Linear activation functions are the most conceptually simple activation functions and are unsuitable for complex tasks.

$$f(x) = mx + c$$

The gradient of a linear activation function is constant and independent of its input, resulting in no variation in the influence of an artificial neuron during training. This makes them unsuitable for hidden layers in a neural network, but they can still be used as the output layer in linear regression problems.

A simple but popular activation function is the Rectified Linear Unit (ReLU):

$$f(x) = \max(x, 0)$$

This function transforms any value below zero to zero, but otherwise does not change its input. ReLU is computationally easy to calculate, but can cause issues at negative and large positive values of x, due to it having no upper bound. Modifications of ReLU exist to address these issues and are often implemented.[120, 121]

In binary classification problems, a sigmoid activation function is often employed.

$$f(x) = \frac{1}{1 + e^{-x}}$$

Despite its higher computational requirements, this has several advantages over the previously mentioned activation functions. The sigmoid activation function's output is bounded between 0 and 1, giving a clear idea of significance. Additionally, its derivative varies with its input, meaning that changes to the weights of its associated artificial neuron can provide meaningful and measurable model refinements during training.

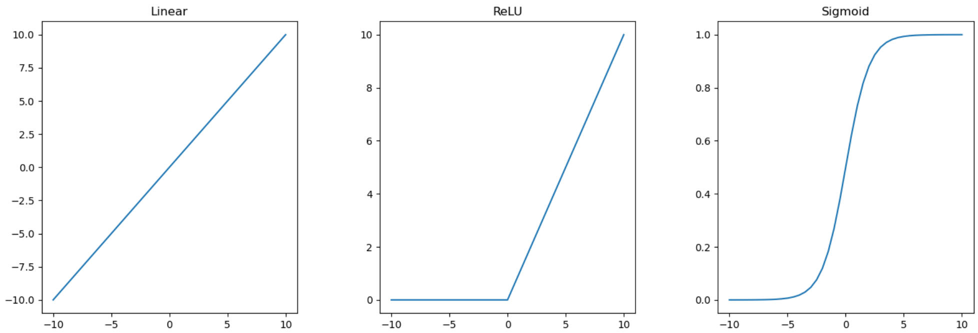

FIGURE 5.6: Examples activation functions. Note that the softmax function is a multivariable function, therefore it is not included in this figure.

A major disadvantage of sigmoid activation functions is that their gradients approach 0 at extreme high or low input values. This can result in the activation function gradient becoming insensitive to large changes in input data if the resulting output tends to 0 or 1, making learning very slow. Sigmoid activation functions are often used in final layers of a neural network for this reason.

The softmax function is a modification of the sigmoid function that allows for multiclass classification. For each value x in a vector of K real values, the softmax function is defined as:

$$f(\vec{x})_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}$$

The softmax function takes all possible class outputs and normalises them into a probability distribution; every output is given a value between 0 and 1 with all values summing to 1.

Each of the activation functions mentioned in this section have variations that are suited to different scenarios and neural networks.

5.2 Results

5.2.1 Image Preprocessing

Images of plates containing *A thaliana* seedlings were recorded using a flatbed scanner. Up to 6 plates were recorded per image in a 2×3 arrangement. An important feature for downstream processing of recorded seedlings is to determine which plate each seedling image originates in. Given the regular shape of the Petri dishes, a preprocessing step using a Hough circle transform was implemented to locate the position of the plates on each image.

Briefly, locating circles on an image with a Hough circle transform first involves converting the coloured image to a binary image using a pixel intensity threshold and Gaussian filter. The edges of the binary image are then located using a Canny edge detector[122]. The positions of each circle with a known radius, r, can be found on the resulting feature map by describing a circle as:

$$(x-a)^2 + (y-b)^2 = r^2$$

where (a, b) are the coordinates of the center of the circle. For each point on the feature map, a circle centred at that point with radius r can be drawn, with the maximal intersection points of all the drawn circles being the center of the circle in the original image.

The parameters of the OpenCV[35] implementation of the Hough Circle Transform were varied until the algorithm performed satisfactorily. With appropriate constraints, all Petri dishes were located on the full set of labelled images with 100 % accuracy. The optimised constraints for the Hough Transformation were an accumulator resolution ratio of 1:1, minimum distance between circles of 1200 px, Canny edge threshold of 200 and circle centre accumulator threshold of 50. Circles detected by the Hough circle transform were also constrained to a radius size of 1750-1950 px, which was appropriate for the size and resolution of images collected. Figure 5.7 shows the results

5.2.2 Bounding Box Annotation

All hypocotyls within each image were manually annotated with bounding boxes, which were then saved as a csv file for each image. An example of an annotated image with its bounding boxes overlayed is shown in Figure 5.8.

A total of 41 images of plates were annotated, corresponding to 4323 hypocotyl bounding boxes available for training.

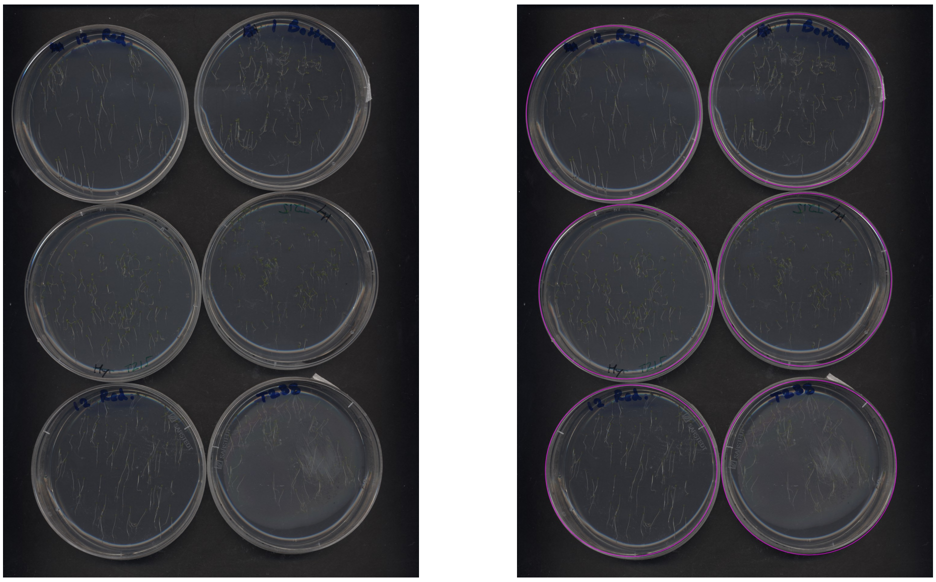

FIGURE 5.7: Example of Petri dishes detected on a training image using a Hough circle transform (shown in pink on the right image).

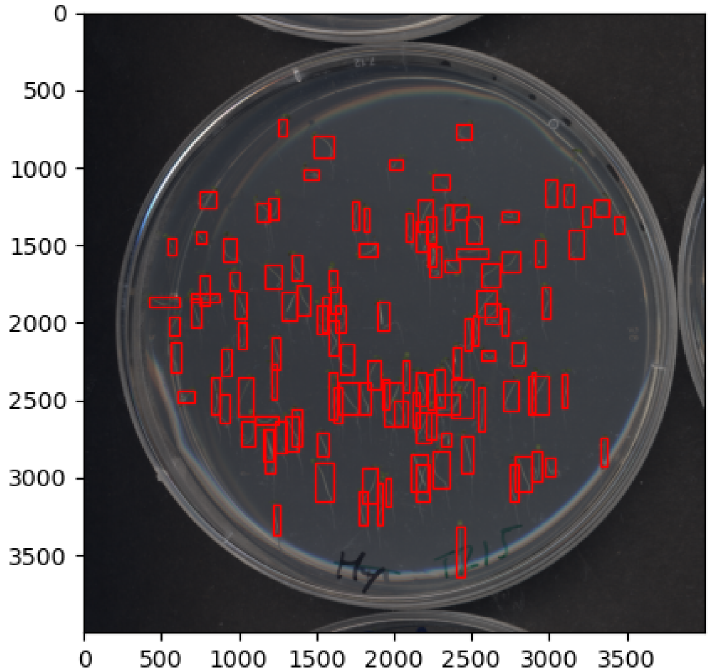

FIGURE 5.8: Examples of a seedling image with overlayed bounding boxes.

5.2.3 Faster RCNN Training

A popular family of machine learning models for computer vision tasks are Region-based Convolutional Neural Networks (RCNNs). These models are usable in both bounding-box and pixel-based computer vision applications. While there are many variants of RCNNs, their primary design philosophy is to first produce regions of interest (ROI) in a target image which are likely to contain important features, then decide which of the regions of interest contain an image of a target class. Important variants of RCNNs include the original RCNN[123], Fast RCNN[124], Faster RCNN[37], Mask RCNN[125] and Mesh RCNN[126].

Anchor Box Generation

RCNNs rely on anchor boxes to subdivide an image into regions which are passed through an RPN to determine whether these are regions of interest. Anchor boxes can be scaled relative to the size of the object that the RCNN is trying to detect.

Figure 5.9 shows a representative example of the training and validation loss over time of the Faster RCNN during training with the following hyperparameters:

The both sets of losses represent a combined figure of classification loss, bounding box loss, and objectness loss for each epoch. Classification and objectness loss are both calculated as a binary cross entropy term:

$$\ell(x, y) = L = \{l_1, ..., l_N\}^\top$$

$$l_n = -w_n [y_n \cdot \log(x_n) + (1 - y_n) \cdot \log(1 - x_n)]$$

where N is the batch size.

Region proposal loss and final bounding box assignments are penalised with a smooth L1 loss term:

$$\ell(x, y) = L_{1smooth} = \{l_1, ..., l_N\}^\top$$

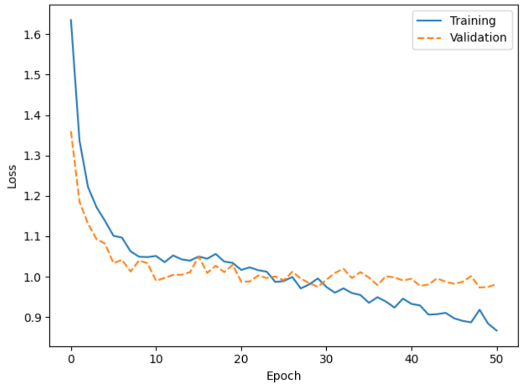

FIGURE 5.9: Training and validation losses for a representative training cycle of the Faster RCNN model.

$$l_n = \begin{cases} 0.5(x_n - y_n)^2, & \text{if } |x_n - y_n| < 1 \\ |x_n - y_n| - 0.5, & \text{otherwise} \end{cases}$$

where N is the batch size.

The training and validation losses suggest that the majority of the learning that can be achieved is done in the first 10 epochs, where a steep decline in both training and validation losses can be observed. The training losses continue to decrease after around the 25th epoch despite there being minimal change to the validation loss, which is indicative of the model overfitting to training data after this point. This suggests that the learning the model can achieve has saturated for this combination of model architecture, hyperparameters and dataset.

Manual inspection of validation dataset bounding box predictions suggests a reasonable ability of the model to identify regions of the image displaying hypocotyls, but ultimately the precision of the bounding boxes are too poor to extract meaningful measurements within a population. Bounding box predictions, while sometimes in a reasonable location, are predominantly focused on densely-populated patches of agar. The model often fails to identify isolated seedlings, suggesting that the model has learned to prioritise identifying dense patches of seedlings as the most likely place to correctly find a hypocotyl at the expense of isolated plants.

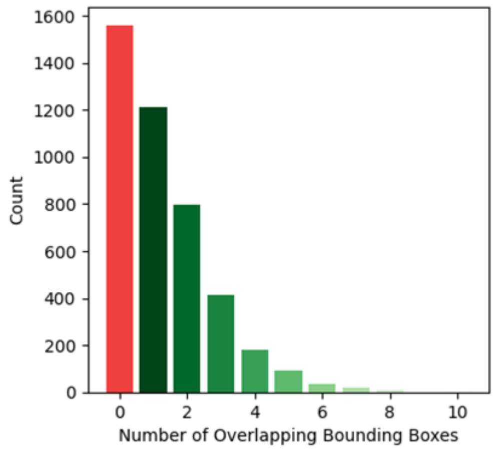

FIGURE 5.10: Bar chart displaying the number of overlapping bounding boxes for each bounding box example in the training data.

Seedling Density

An analysis of bounding box overlap in the training dataset was performed to determine how many hypocotyl examples were provided as isolated images. Figure 5.10 shows the distribution of overlapping bounding boxes across the entire training dataset and illustrates that isolated bounding boxes are actually in the minority.

5.3 Discussion

The loss scores over time during model training show that the model is capable of learning the position of hypocotyls on an image to some extent, but ultimately the performance of the model in validation samples still has significant room for improvement. It is likely that the crowded images used to train the model introduce quite a bit of confusion in its assignments. Unfortunately, this type of image is likely to be the most common type encountered by a program built on this kind of technique. Another potential issue with a computer vision approach reliant purely on bounding boxes is that a significant portion of bounding box area tends to be occupied by either background agar or other seedlings that are not the plant of interest. During training, the objectness loss works to penalise false hypocotyl assignments, but the presence of multiple hypocotyls in one bounding box seems to have caused significant difficulties in training.

It is possible that other network architectures could be used to solve the issue of crowded agar plates, as several developments in recent years have been made in crowded image object detection. Most of these specialise in human detection, with a view to contribute to technologies such as self-driving cars.[127–129] The simplest solution to the difficulties in processing crowded images could be for the experimenter to ensure that sufficient spacing is present between seeds when sowing. In practice, this can be difficult to achieve with consistency and also means that less seedlings can be processed per plate, causing a drop in throughput and also requiring a greater input of consumables to perform each trial.

Training a model with an object segmentation mask approach, with every pixel of hypocotyl images provided, would possibly give a more accurate identification of hypocotyl location but will likely run into similar difficulties with clustered patches of seedlings. Misidentification issues could also occur in overlapped samples, for example considering parts from two overlapping hypocotyls as one continuous hypocotyl, making resulting automated measurement values dubious.

As with any machine learning task, a larger training dataset will almost always contribute to a more robust and powerful final model. The training dataset used in this chapter contains more individual seedlings than are present in existing datasets, but is annotated with bounding boxes instead of pixel-wise segmentation, as is present in Reference [112] The images used in the training dataset of this chapter represent a small subset of the total images processed in Chapter 4. However, given the labour-intensive process of manually annotating images and the purpose of this program as a time and labour-saving tool, no further annotations were performed.

The final processing step of the program following bounding box assignment is to calculate the diagonal of every bounding box identified to approximate the length of the bounded hypocotyl. The resulting csv output for each image contains box coordinates and diagonal length. In the case of isolated hypocotyl assignments, a more accurate measurement of hypocotyl length could be implemented using pixel brightness thresholding, extracting a binarised image, then measuring the resulting length using a function such as OpenCV's arcLength.

5.4 Conclusions and Future Work

The work presented in this chapter provides a foundation for a program that can help to automate laborious data processing when identifying and measuring the lengths of seedling hypocotyl datasets. The image handling and preprocessing steps of the pipeline perform exceptionally well, but ultimately the object detection model that has been produced is not yet suitable for use and requires further optimisations. Further experimentation with different neural network architectures and data augmentation could provide workable improvements in model accuracy, but the likelihood is that the crowded nature of the images used to train the model are the cause of significant difficulties in model training. If Petri dishes are carefully prepared to ensure no overlap between seedlings, the strategy of object detection based purely on bounding box information could see greater success, but well-spaced seedlings can prove difficult to consistently produce.

Automated image processing powered by deep learning remains a powerful tool that can provide unprecedented speed in tasks that would otherwise require a significant investment of speed and manpower.

Chapter 6

Improving New Tools for Small Angle X-Ray Scattering Analysis

6.1 Introduction

Some of the work outlined in this chapter contributed to a 2020 paper [130] published in the Journal of Chemical Theory and Computation. Within the paper, a novel algorithm named the Constrained Backbone (CB) algorithm is described that can be used in the ab initio interpretation of small angle X-ray scattering (SAXS) data. The work towards creating the SAXShund web server was performed collaboratively with another PhD student, Arron Bale.

6.1.1 Introduction to SAXS

Small angle X-ray scattering (SAXS, often referred to as bio-SAXS in a biological context) is a technique that can provide structural information of biomolecules in solution. Observing proteins in a quasi-native state offers important structural information that might not be obtainable in their crystallised form, but the inherent positional and orientational disorder that aqueous proteins possess leads to significant resolution loss compared to protein X-ray crystallography.

The accessibility of SAXS to bioscientists has increased significantly in recent decades, with many synchrotrons now operating dedicated bio-SAXS beamlines[131]. One of the major advantages of bio-SAXS is the ease at which samples can be prepared. For many samples, the only polishing step that follows protein purification is the removal of aggregated particles that have developed during storage. In practice, size exclusion chromatography is typically performed on the sample at the synchrotron shortly before irradiation (SEC-SAXS) to improve sample homogeneity and remove aggregates, which are capable of generating significant amounts of signal noise[132]. The simplicity of sample preparation is a stark contrast to protein crystal production for x-ray diffraction studies, which often require a significant time investment in crystal optimisation and can also require the expression of truncated or optimised protein to produce satisfactory crystals.

In comparison to crystallography, bio-SAXS is also significantly more tolerant of flexible regions of a protein, as found in the linker sections of multidomain proteins or large sections of intrinsically disordered proteins [133]. It can also be used in time-resolved experiments to study large-scale conformational changes of proteins in dynamic settings.[134]

Despite these advantages, the main drawback of bio-SAXS is the limited resolution obtainable from the technique. Protein molecules in solution tumble in a random orientation with respect to an incident X-ray beam and are also afforded more conformational freedom than protein within a crystal lattice. Ab initio models generated with bio-SAXS data are typically limited to a low resolution, making them unable to distinguish important small-scale protein features, such as side chain orientation. Nevertheless, bio-SAXS remains a versatile tool in determining high-level structural information of proteins and their complexes. Bio-SAXS models can be improved by including high-resolution information obtained from complementary techniques, for example with bio-SAXS providing the overall structure of a multi-subunit complex, with low-level information for each subunit provided by high-resolution techniques such as crystallography[135].

Data Collection

A typical bio-SAXS experiment involves exposing a protein sample in solution to X-rays from a highly collimated beam source. Incident X-rays are scattered by electrons in the sample and detected as a function of momentum transfer, q. In dilute solutions, the scattering profile is continuous and isotropic due to molecular tumbling. With an isotropic scattering profile, the intensity distribution observed by the

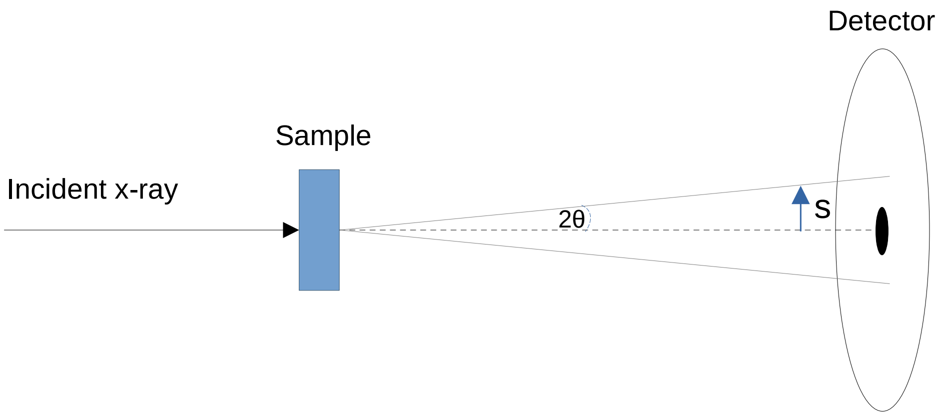

FIGURE 6.1: Schematic of a BioSAXS experimental setup

detector can be radially averaged, resulting in a scattering curve usually plotted as $I(q)$ vs q, where $q = 4\pi\sin(\theta/\lambda)$, 2θ is the scattering angle and λ is the incident wavelength. The background scattering contribution from the sample buffer must also be considered and, in practice, is recorded as a 'blank' scattering curve and subtracted from the scattering curve of the sample before data analysis.

In general, the scattering intensity from a monodisperse macromolecular solution after the background contribution has been removed can be described as $I(q) \approx F(q) * S(q)$, where $F(q)$ is the scattering from the particle, also known as the form factor, and $S(q)$ is the modification to the scattering of each particle by attractive or repulsive interactions with other particles in the solution, also known as the structure factor. Several scattering profiles of different sample concentrations are usually recorded to account for any structure factor effects.

Data Analysis

The resulting scattering curve is plotted in such a way that intensities at smaller values of q (smaller scattering angles) relate to larger, long-range structural information and finer details can be found at larger values of q. Several important sample parameters can be determined with a relatively simple analysis of the scattering curve, including the radius of gyration R_G and maximum particle size D_{max}.

The production of three-dimensional models from scattering data is typically performed either by taking an existing protein model and adjusting it to fit the scattering data, or by producing an ab initio model that is initially simplified to a bead model without explicit secondary structure characterisation. There are many examples of Bio-SAXS data analysis software, with many beamlines possessing their own in-house examples that primarily focus on data cleaning and scattering curve analysis without a 3D-modelling component. The most prominent 3D-modelling softwares for Bio-SAXS are found in the ATSAS Software Suite[136]. DAMMIN[137] and DAMMIF[138] create ab initio models using a 'dummy atom' model with volume representations of atoms belonging to either the modelled protein or the solvent in the immediate vicinity of the protein. GASBOR[139] attempts to refine the bead model further by proposing a chain-like ensemble of 'dummy residues' with each sphere roughly proposing the location of the Cα of each residue.

6.1.2 The Constrained Backbone Algorithm

The unavoidable limitation of resolution encountered in Bio-SAXS makes it difficult to produce 3D models from SAXS scattering data without either supplementing the data with additional information from complementary techniques or by making significant concessions within the model, such as the anonymisation of the 'dummy' atoms and residues by programs within the ATSAS suite. An alternative source of information that can be used to direct model building from Bio-SAXS data are the predictable physical constraints that protein backbones are subjected to. While protein molecules are constrained by the usual attractive and repulsive intramolecular forces encountered in simple molecules, the tendency of secondary structure elements to form in proteins allows for additional granularity in backbone geometry prediction. Ramachandran plots are classically used to visualise energetically-reasonable backbone dihedral angles of amino acid residues for each secondary structure archetype in proteins and are still commonly used in model validation.[140] Despite the usefulness of Ramachandran plots in model design, the permissible angles described in Ramachandran plots require the modelling of N and Cβ atoms in the protein backbone in addition to Cα atoms. The constrained backbone (CB) algorithm uses two empirically-determined constraints, curvature and torsion, to constrain model geometry that require only Cα atom coordinates, allowing for a more parsimonious model while still describing important structural features.

Curvature and Torsion

In general, the curvature (κ) of three points in 3D space, c_1, c_2, c_3, is given by:

$$\kappa(c_1, c_2, c_3) = \frac{2|sin(\theta_{123})|}{||c_1 - c_2||}$$

Where $\theta_{1,2,3}$ is the angle between vectors $c_1 - c_2$ and $c_2 - c_3$. This can be extended to sections of a discrete curve containing four points by using midpoints. For a curve subsection, m_i, comprising four points:

$$m_i = \frac{c_{i+1} + c_i}{2}$$

Thus for a section (c_1, c_2, c_3, c_4) we can define:

$$\kappa(c_1, c_2, c_3, c_4) \equiv \kappa(m_1, m_2, m_3)$$

Torsion (τ) is effectively a measure of curve non-planarity. For a curve subsection of four points (c_1, c_2, c_3, c_4), two planes can be defined for (c_1, c_2, c_3) and (c_2, c_3, c_4) with plane normals n_1 and n_2, respectively. Plane normals are unit vectors given by:

$$N = (c_{i+1} - c_i) \times (c_{i+2} - c_{i+1}),$$

$$n = \frac{N}{||N||}$$

If the curve is planar, n_1 and n_2 will be equal. The angle between n_1 and n_2 can be used as a measure of curve subsection non-planarity. From this, the curve torsion can be calculated using:

$$\tau(c_1, c_2, c_3, c_4) = \frac{2}{l} sin(\frac{\theta_n}{2}),$$

$$l = \frac{||c_2 - c_1|| + ||c_3 - c_2|| + ||c_4 - c_3||}{3}$$

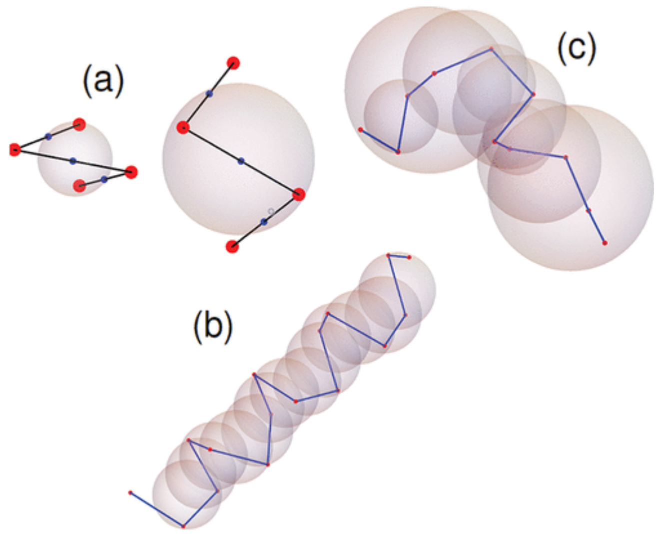

FIGURE 6.2: Figures depicting examples of discretised curves with each subsection, comprising 4 points, inscribing a sphere using the midpoints of each consecutive pair of points. a) Curve subsections (c_1, c_2, c_3, c_4) are in red and their midpoints (m_1, m_2, m_3) are in blue. The curvature κ of each subsection is the reciprocal of the radius of its inscribed sphere, hence tighter subsections inscribe smaller spheres and have larger curvatures. b) α-Helical sections of backbone tend to show consistent (κ, τ) values. c) Flexible (linker) sections show a much more variable distribution of (κ, τ) values. Figures from Reference [130].

Where θ_n is the angle between n_1 and n_2 This definition takes into account the length of the curve as well. The chirality of a curve section is also reflected in its τ sign, which is positive for right-handed coiling and negative for left-handed coiling.

Figure 6.2 depicts several curves that illustrate the variety of (κ, τ) values that can be encountered in protein backbones and the impact of (κ, τ) values on curve subsection morphology. By employing secondary structure information, which can either be predicted from a protein's primary sequence or determined from a previously-obtained model, (κ, τ) values can be constrained for each subsection during model refinement with the CB algorithm.

Model Construction with the CB Algorithm

A detailed description of the parameters and calculations involved in the CB algorithm can be found in Reference [130], but a brief outline is provided in this section. During model refinement with the CB algorithm, a protein Cα backbone of length m is split into $m - 3$ subsections of length 4, each with their own (κ, τ) values. The separation between each Cα, R, is fixed to 3.8 Å during model initialisation.

If the initial model structure is randomised, the coordinates of the first three points in the backbone can be randomised as long as the distance between them is constrained such that all points are separated by R and the distance between c_1 and c_3 is greater than R. The coordinates for c_4 are then generated using the (κ, τ) values associated with the first subsection, and the rest of the backbone is created iteratively using the final three points of the completed subsection as the first three points of the next. The CB algorithm can also be initialised using a preexisting Cα coordinate file, in which case (κ, τ) values are calculated from the initial coordinate set.

With the points of the Cα backbone generated, an explicit hydration layer is modelled and a scattering curve is calculated using the Debye formula[141]

$$I(q) = \sum_{i=1}^{N} \sum_{j=q}^{N} f_i(q) f_j(q) \frac{sin(qr_{ij})}{qr_{ij}}$$

where $I(q)$ is scattering intensity as a function of momentum transfer, $q = \pi \sin(\theta)/\lambda$ (as above). Here N is the total number of Cα atoms and solvent molecules and $f_i(q)$ is the form factor for residue i.

During model refinement, scattering curves of proposed models are calculated and compared with smoothed experimental scattering data using a chi-square statistic considering data of the domain $q \in [0, 0.4]$. Additional constraints during model refinement include penalties for nonadjacent Cα atoms that are placed closer than 3.8 Å, and a penalty for the backbone geometry failing to reflect predicted residue contacts, if any are provided. Structures are altered from initial configurations using Monte Carlo optimisation until a reliable fit with a low chi-squared value is obtained.

The distribution of (κ, τ) values permissible for each secondary structure element in the context of Reference [130] were taken from a small subset of proteins from the Protein Data Bank.[**Armstrong2020**, 18] Work performed in this chapter aimed to curate a more robust dataset to draw (κ, τ) values from. A web server prototype was also designed to act as a user interface for data submission.

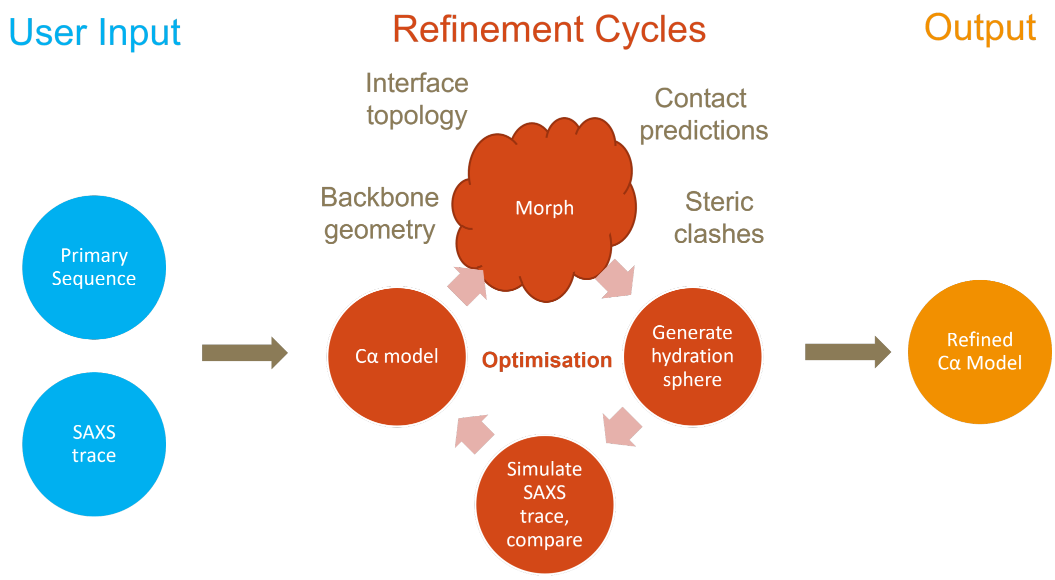

FIGURE 6.3: Simplified schematic of the CB algorithm.

6.2 Results

6.2.1 Data Curation

Figure 6.4 shows the distribution of chain lengths observed in the original 2343 structures downloaded from the RCSB PDB as described in Section 2.3.1. Since there are examples of chains that are longer or shorter than the maximum and minimum length cutoffs, respectively, it was apparent that the search filter permitted heteromultimers possessing chains that violated the criteria as long as at least one of the monomers was a positive match. Data cleaning was performed by taking each protein chain in the PDB files downloaded and thresholding out chains that are above or below the length constraints outlined in section 2.3.2.

Additionally, redundant chains were still present in the dataset, usually as components of homomultimers. Filtering out each redundant chain based on sequence identity with other chains within the dataset produced a final dataset of 3048 unique chains of suitable size.

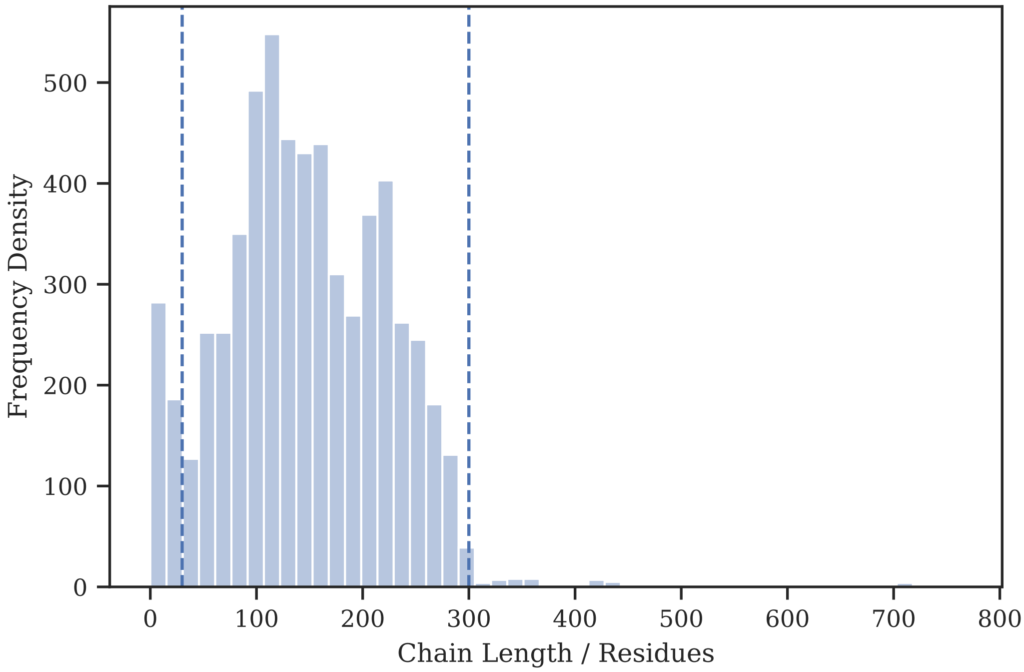

FIGURE 6.4: Histogram of the chain length distribution of the original dataset produced by the PDB query. Vertical lines represent desired chain length cutoff points.

6.2.2 Distance Mapping

After filtering out every model that disobeyed the sampling constraints, 903510 records of residue pairs within 8.0 Å of each other were obtained. For these calculations, protein chains were simplified to only contain Cα atomic coordinates. Contacting residue pairs are shown as an aggregated population and also separated into short-, medium- and long-range contacts based on the distance between residues in sequence space. These definitions were based on the CASP definition of a contacting residue pair[142], albeit considering Cα pairs instead of Cβ pairs since Cβ atoms are not included in CB algorithm models. Short-, medium- and long-range contacts are defined as residues separated by 6-11, 12-23 and greater than 24 residues, respectively. Contacting residues with a separation of less than 6 residues are not considered as they generally correspond to contacts within secondary structure elements. A total of 192,035 contacting residue pairs were surveyed.

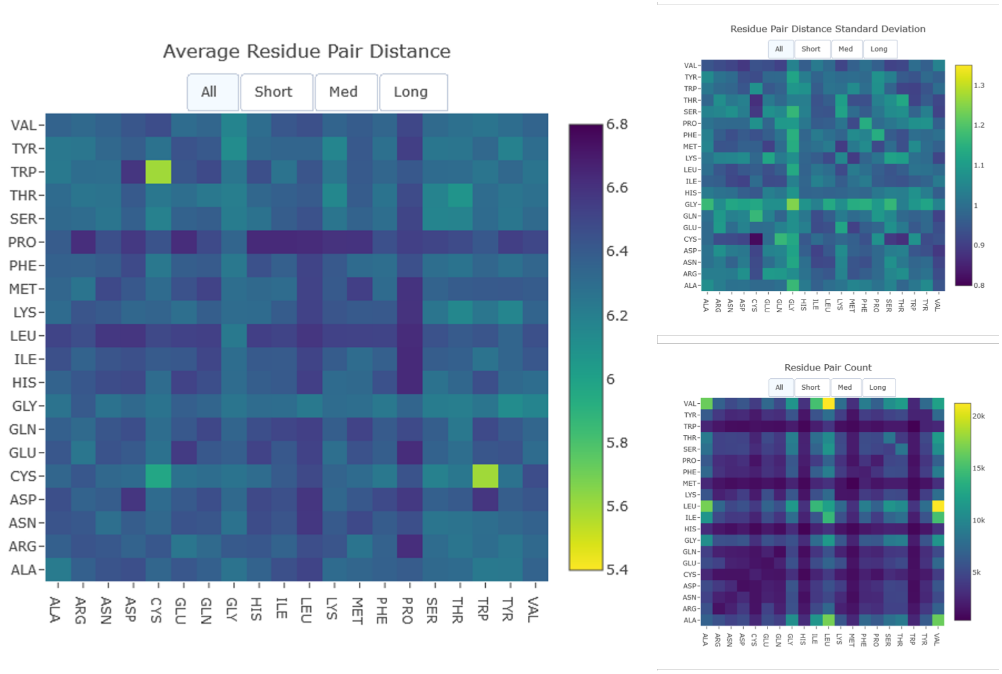

FIGURE 6.5: Heatmap of the average distances between all contacting residues.

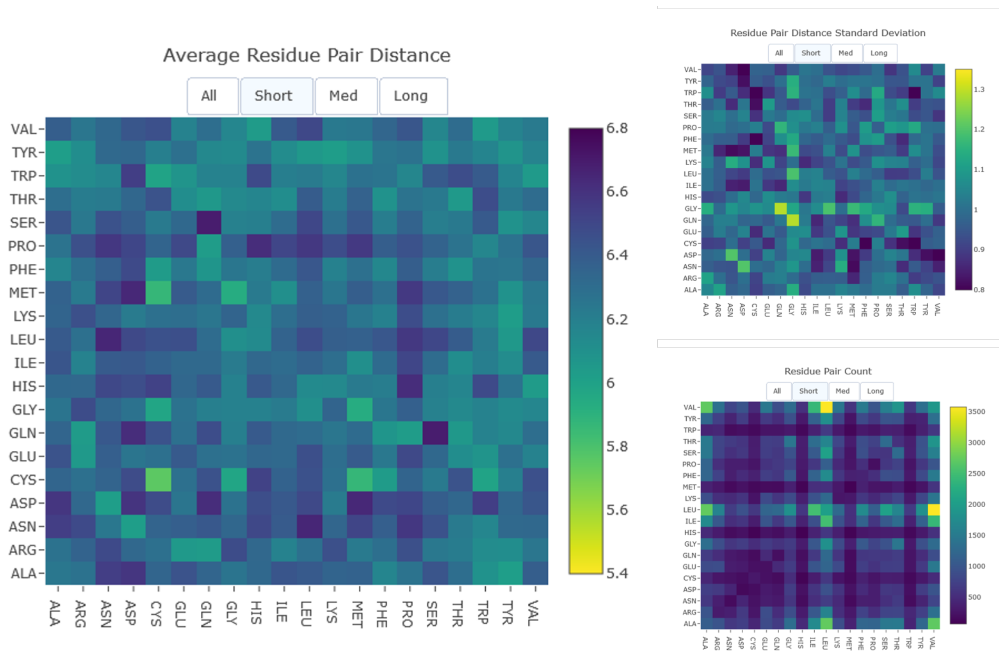

FIGURE 6.6: Heatmap of the average distances between short-range contacting residues.

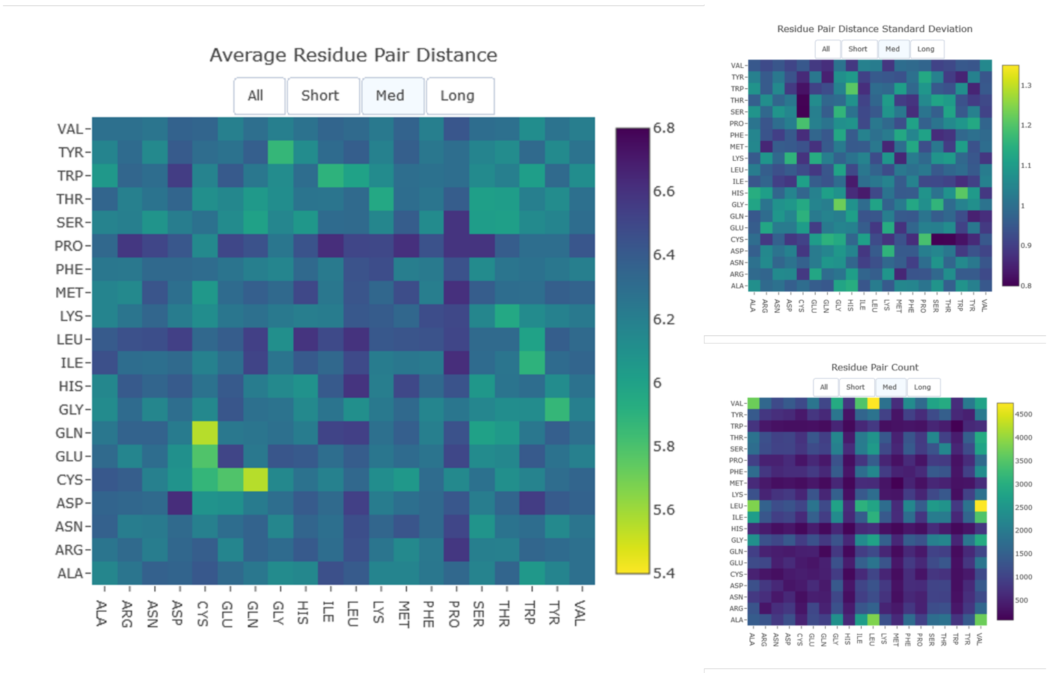

FIGURE 6.7: Heatmap of the average distances between medium-range contacting residues.

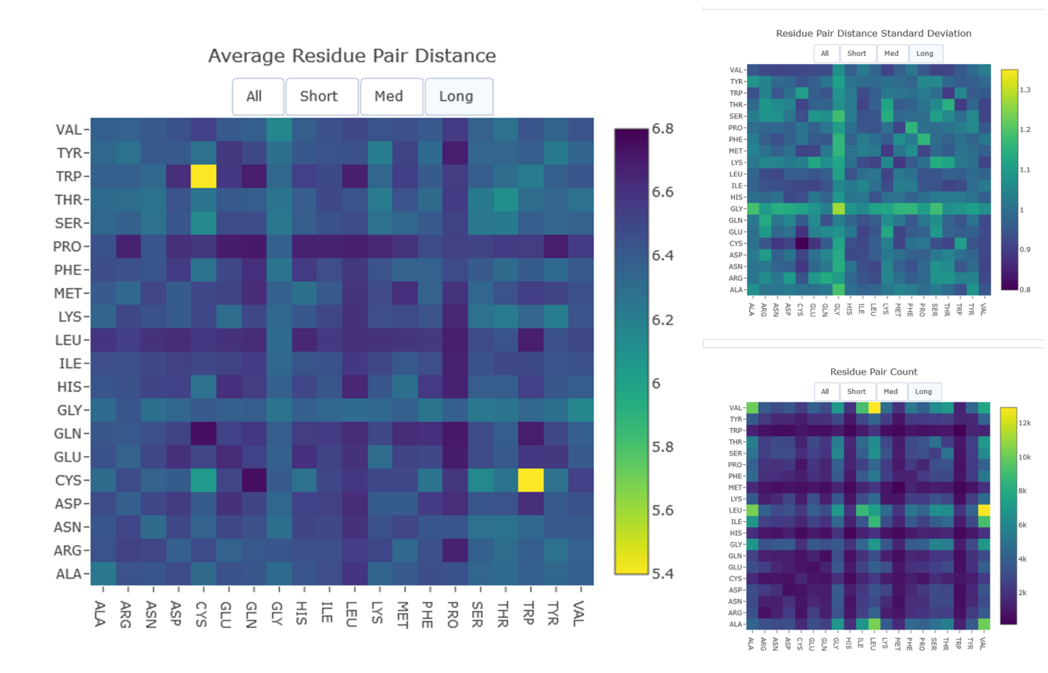

FIGURE 6.8: Heatmap of the average distances between long-range contacting residues.

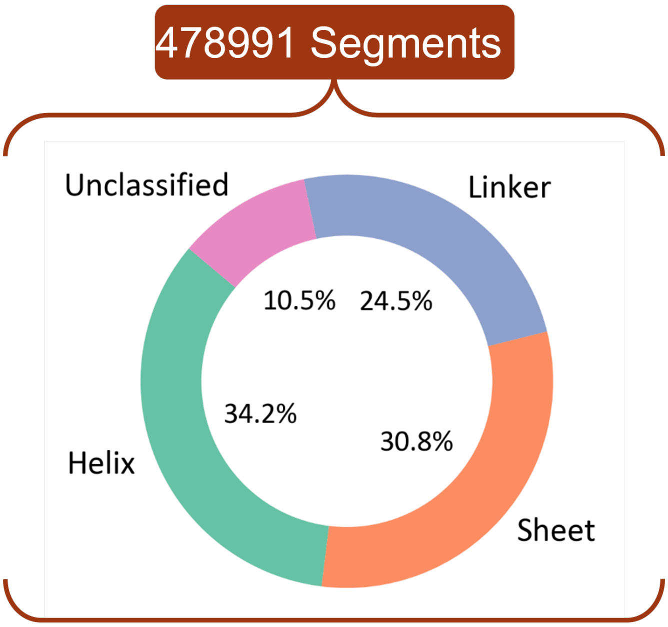

FIGURE 6.9: Doughnut plot summarising subsection secondary structure assignments within the dataset.

6.2.3 Secondary Structure Determination

Every protein chain in the cleaned dataset was divided into subsections comprising 4 Cα atoms as described in Section 6.1.2. DSSP[33] was used to provide the secondary structure element that each residue belongs to (each residue is given an individual assignment) and the DSSP assignments were simplified into helix, sheet or linker categories as described in Table 2.4. Subsections were assigned a classification of helix, sheet or linker based on the modal secondary structure element of its component residues. In the event of a tie between modal secondary structures, the segment classification were given an 'Unclassified' assignment (also given if the majority of residues in the subsection were not assigned a secondary structure with DSSP) to avoid confusion. Figure 6.9 summarises the secondary structure assignments of all 478,991 subsections.

6.2.4 Curvature-Torsion Analysis

Figure 6.10 is a summary of curvature and torsion values obtained for the entire dataset. While the curvature (κ) shows values broadly between $0.0 < \kappa < 0.8$, it has two identifiable regions of higher sample density: a broad, minor peak around $\kappa = 0.2$ and a sharp major peak around $\kappa = 0.7$. Torsion (τ) values are generally localised

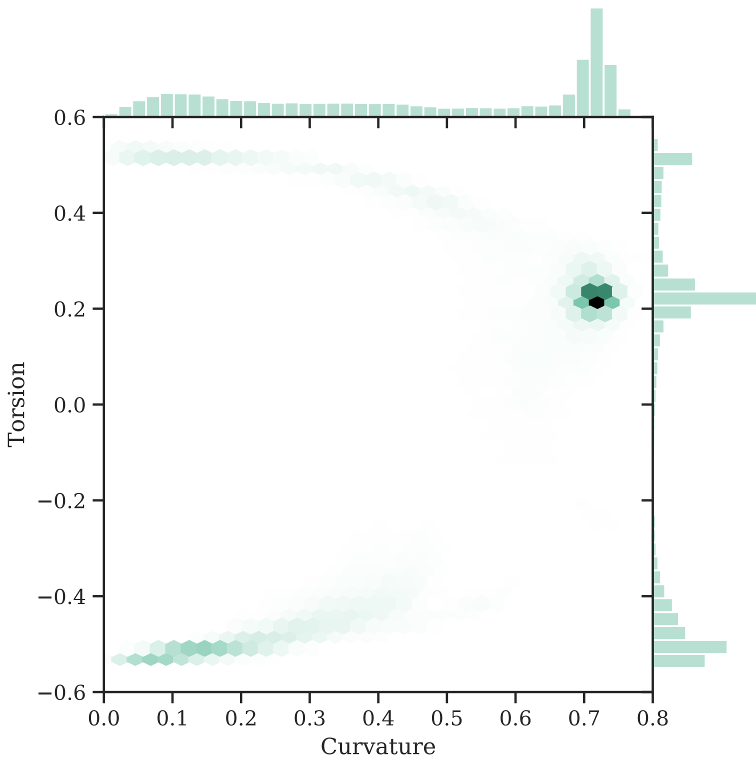

FIGURE 6.10: Hexbin plot showing the distribution of curvature and torsion values for all chain subsections.

to two regions: 0.0 < τ < 0.6 and -0.3 > τ > -0.6, with major peaks around τ = 0.2 and τ = -0.5, and a minor peak around τ = 0.5.

Figure 6.11 shows curvature and torsion distributions for each segment when classified into their modal secondary structure elements. These plots show that segments of mainly helical character are strongly localised around κ = 0.8 and τ = 0.2. This is consistent with the fairly narrow geometric constraints of α-helices in proteins. Strands are mainly found in two regions both found between 0.0 < κ = 0.2, with one at τ = 0.5 and the other at τ= -0.5. These regions correspond to right- and left-handed strands, respectively, with the left-handed population being the larger of the two. Unclassified regions seem to be predominantly strand-like in character, while linker sections are found at a much broader range of curvature and torsion values, with a large population found around the 'helix region' of curvature-torsion space and strong densities around the 'strand regions'. Linker regions are also afforded more flexibility in the curvature and torsion values, leading to less well-defined clusters.

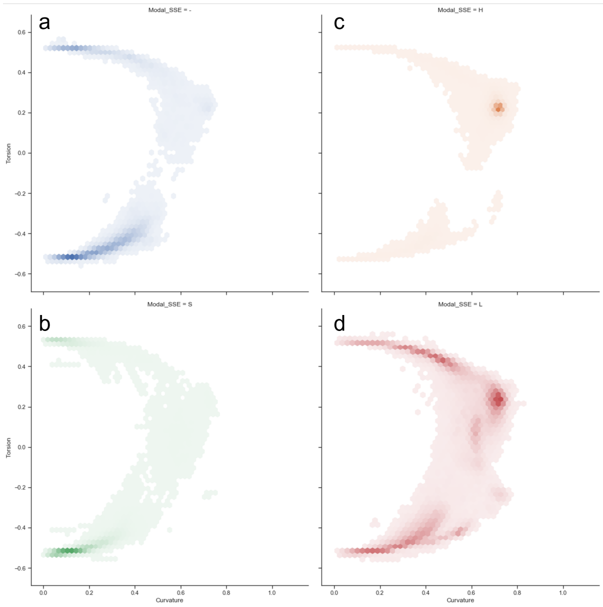

FIGURE 6.11: Hexbin plots showing the distributions of curvature and torsion values for chain subsections classified using their modal secondary structure element: a) Unclassified, b) Sheet, c) Helix, d) Linker. The minimum bin size to be displayed is 10.

6.2.5 Web Server

A Python Django web server prototype named SAXShund was developed to accompany the CB algorithm code comprising a customisable user data submission section, automatic data preprocessing, user account functionality alongside some decorative elements such as a blog section to provide updates on SAXShund development. Each submission is treated as a separate project and saved to a projects database alongside appropriate metadata. The user must provide, at a minimum, a FASTA format protein sequence and a SAXS scattering curve data file. However, a significant amount of additional data can be provided to fine-tune the output of the CB algorithm. These include:

- **Secondary structure information** provided as a raw text file. Secondary structure predictions could be obtained from either a secondary structure prediction server or from a user-generated homology model. By providing secondary structure element assignments in advance, sections can be constrained within reasonable curvature-torsion values during model refinement.
- **Initial atom coordinates** provided as a pdb file. The CB algorithm can generate randomised initial Cα positions in the absence of a template to begin with, but

the CB algorithm generally outputs a better model in a shorter period of time if it is given a good template to start from. For most users, this template is expected to be from either experimentally-obtained coordinates, a model from the Protein Data Bank, a homology model or another structure prediction.

- **Fixed sections** can be specified to keep the relative positions of certain atoms within the template fixed during the refinement process. This can be useful if the user has a specific helix or fold that they want to preserve during model refinement.

- **Contact predictions** could be obtained from a contact prediction server, or could be used to restrict atoms at a predicted interface during the model refinement process. Residues specified as contacting are given stricter distance penalties during model refinement.

- **Biological assembly** allows the user to specify the quaternary structure of their model. For monomers, no further preprocessing of atomic coordinates is performed at this stage. Users uploading initial atom coordinates in pdb format with symmetry information records can select a 'Parse PDB' option, which uses a preprocessing function to parse the pdb file for REMARK350 rows, which specify symmetry operations as described below. A final option allows the user to specify their own symmetry operations to generate initial atomic coordinates.

- **Symmetry** Many protein atomic coordinates are provided as an asymmetric unit instead of a biological assembly. The standard pdb file REMARK350 specifies n 4×3 transformation matrices (a 3×3 rotation matrix and a 1×3 translation vector) and which chains to apply each transformation on. Users can upload a symmetry operation to generate customised biological assemblies from their own asymmetric units for initial atomic coordinates.

- K_{max} settings allow the user to specify the maximum momentum transfer in reciprocal space (also referred to as s or q) value considered in the SAXS scattering curve. This value inversely scales with real-space distances between atoms

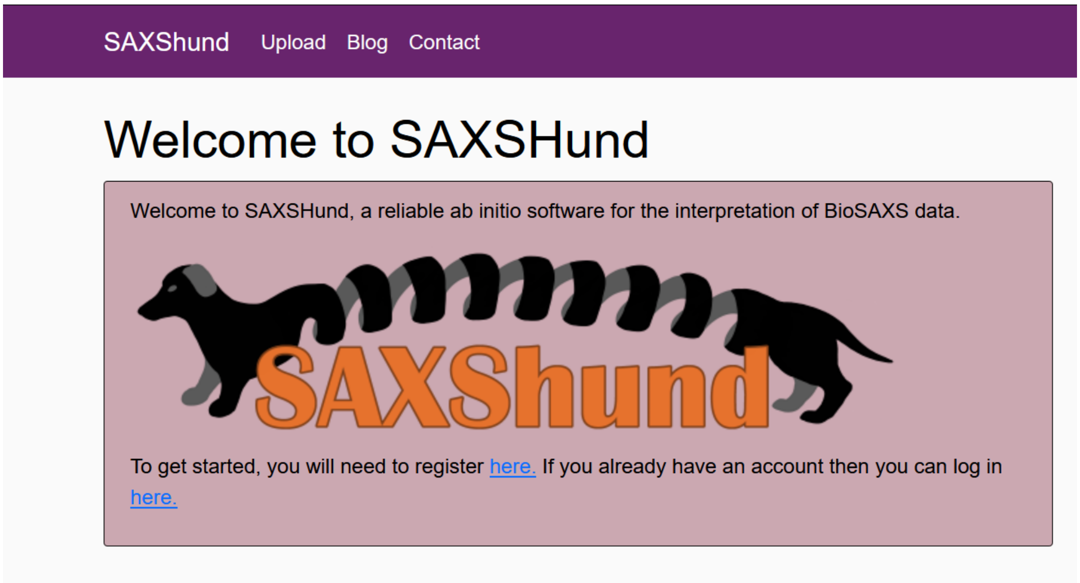

FIGURE 6.12: Screenshot of the SAXShund landing webpage. Data submission and processing can be performed after a user account is created.

in the scattering molecule and is often truncated to smaller values during data processing as noise tends to dominate at larger values.

- K_{min} settings allow the user to specify the minimum momentum transfer in reciprocal space value considered in the SAXS scattering curve.
- **Refinement steps** sets the number of refinement cycles within the CB algorithm.

The filepaths of each of the submitted files for each project are automatically saved to a project database alongside username, project name and date/time metadata for easy project organisation.

6.3 Discussion

6.3.1 Data Curation

The cleaned dataset represents a collection of high-quality structures obtained from the PDB that are of an appropriate size. Despite considerable advances in resolution and data quality obtainable with complementary techniques such as cryo-EM, crystallographic data provides the largest breadth of high-resolution data to draw from, especially for proteins of the size ranges considered in this study. However, there are unavoidable factors that could affect protein geometry within a crystalline lattice versus in solution. The translational and rotational order that is enforced on protein molecules within a crystal is ultimately what allows x-ray diffraction to occur and high-resolution structural information to be obtained. This order could result in protein conformations being favoured that would disfavoured in solution, particularly in regions involved with crystal contacts. With the exception of extended flexible regions, it can, however, be expected that overall protein structure remains workably consistent between crystallographic models and Bio-SAXS models and thus the (κ, τ) values calculated in this study remain applicable to solution-phase proteins.

6.3.2 Distance Mapping

Two significant pieces of information can be obtained from the residue distance mapping results. Firstly, contacting residues have a strong tendency to sit closer than the 8 Å threshold that is currently enforced in the CB algorithm, with the average of some residue pairs sitting between 5 Å and 6 Å. Secondly, there are noticeable differences in mean contact distances between residue pairs when they are segregated based on the distance between them in sequence space. Despite these differences, the variation in contact distances between different sequence separations is generally small and falls well under the expected model resolution achievable with Bio-SAXS data.

A result that stands out within the residue distance mapping is the long-range TRP-CYS pair. With a mean distance of 5.4 Å (SD = 1.1), the TRP-CYS pair sits at a shorter distance than both the other long-range amino acid pairs and TRP-CYS pairs of shorter sequence separation. This can perhaps be explained by thiol-π interactions that have been observed to involve cysteine residues in other studies.[143]

6.3.3 Curvature-Torsion Analysis

The results shown in Figure 6.11 show (κ, τ) distributions of each subsection secondary structure classification. Helical subsections show generally uniform geometry, with a high density peak around $\kappa = 0.7$, $\tau = 0.2$. The observed τ value of 0.2 is consistent with expectations, given the favourable energetics of right-handed helices with levorotary amino acids that are predominantly found in naturally-occurring proteins. The narrow range of curvatures at this torsion range is also predictable due to the conformational strictness required to maintain a hydrogen bonding nextwork between residues in the helix. There are regions of different (κ, τ) values on the distribution that are significantly less populated and appear more 'sheet-like' or 'linker-like' in their geometries. Further analysis could be performed here by segregating the helix subsection population by the purity of the subsection (e.g. subsections with 'HHHH' residues vs 'HHHS').

Sheet subsections are confined to two ranges of torsion values at either extreme of the scale. The curvature values are low, indicating significantly flatter geometries compared to helical sections. The wider spread of densely-populated regions suggests a higher degree of conformational variation when compared with helical subsections, including a bimodal distribution in τ values which indicates left and right handedness in different subsections. Since the stabilising hydrogen bonds in β-sheet structures are found between its constituent strands, there is less need for structural rigidity between consecutive subsections, as is found in helical subsections.

Linker subsections display a wide range of moderately-populated geometries which encompass both topologies encountered in helical and sheet subsection distributions. Given the similarities between the linker subsection (κ, τ) distribution and the overall distribution in Figure 6.9, it can be concluded that linker subsections have the widest conformational variability. Within the linker subsection distribution, there are regions of relatively higher population density at (κ, τ) values characteristic of helix and sheet subsections, suggesting that linker sections may be 'sheet-like' or 'helix-like' in their geometry. Again, segregating the subsections by their secondary structure purity could provide additional insight in the distributions that are possible, but could risk excessively limiting the (κ, τ) values that are available for a model to sample from during CB algorithm model refinement.

It is also important to note that protein molecules confined within a lattice are still afforded varying degrees of conformational freedom which can lead to localised regions of poorly-resolved electron density during model construction. The models produced with crystallographic data have a degree of uncertainty associated with each atomic coordinate, quantified as an atomic B-factor. B-factors were not considered when creating (κ, τ) distributions, but could be potentially useful in identifying less-likely regions in (κ, τ) value probability distributions in further analysis.

6.3.4 Web Server

In keeping with a decades-old tradition of punmanship in the naming of structural biology software, the web server designed to apply the CB algorithm to user datasets has been dubbed SAXShund (sounds like dachshund). The majority of code written to perform the preprocessing of user data before using the CB algorithm is in Python. As a programming language, Python's main strengths are its ease of writing and the robust ecosystem of scientific packages that are designed to deal with scientific data. Biopython[31], a popular Python package, has been used frequently throughout this chapter due to its useful handling of pdb and mmcif files, which are commonly used to store protein coordinate information. The Python ecosystem also possesses several frameworks that are suited to web design, including Django[**django**] and Flask[**flask**]. Django was selected as the best option to develop the SAXShund server because of its strong documentation and its ability to quickly implement user accounts, which were predicted to be useful in organising and delivering results in a secure way.

For bioinformaticians and programmers in the life sciences, an important consideration to make when creating tools and techniques is how much subject knowledge the average end user will possess in the programmer's area of expertise. While most 'wet lab' scientists possess a working knowledge of the techniques they employ as part of their research, for many users techniques are simply a means to an end in providing experimental insight. There is a wealth of data analysis software available to structural biologists and, for a first-time user, speed and simplicity are often the deciding factors in whether or not a piece of software is used over its competitors. Despite this, proteins represent an incredibly complex and varied class of molecule that often requires nuanced analysis to deal with the quirks of each molecule. This presents an interesting challenge for those designing a user interface for a scientific application; balancing simplicity and ease of access with customisability to tackle unique experimental targets and setups.

Web servers are popular options for applications dealing with light amounts of data that provide a very low bar for access by shifting the need for computing power away from the user and delivering software in a browser environment that is familiar to anyone with basic computer literacy. Significant amounts of programmatic abstraction can be used to produce a simple user interface while all of the requisite analysis is performed 'behind the scenes' on the server end of the application. This accessibility makes a web server an attractive choice for implementing the CB algorithm for a wide user base.

Data Submission

A key feature of any automated web server is the standardisation of data entered into it. SAXShund accepts a wide range of possible input files and data types, as listed in Section 6.2.5, and possesses validator functions to ensure submitted data is submitted with the correct file extension. It possesses a range of preprocessing functions that can convert submitted data into a format that is appropriate for running with the CB algorithm (Appendix B).

Using BioPython parser functions, SAXShund can read and organise fasta and pdb files. If a pdb template is provided as an initial coordinate set for the CB algorithm, SAXShund strips the coordinate data to include only Cα atoms and provides the option for the user to include or exclude individual chains of interest. If no secondary structure information is provided by the user, automatic secondary structure assignment can be performed if suitable information for the protein of interest is available on the PDB. SAXShund can make a request to the PDB REST-API (REpresentational State Transfer Application Programming Interface, a way to communicate information between applications) using the sequence of interest. If a hit is found with 100 % sequence identity with the search sequence, the PDB file is downloaded and DSSP is used to generate a secondary structure assignment (simplified to HSL notation as specified in Table 2.4).

If initial coordinates are given as a crystallographic asymmetric unit but the protein of interest exists as a multimer, SAXShund can generate coordinates of the biological assembly before file conversion as long as symmetry information is provided in the form of REMARK350 records in the PDB file.

After data is submitted and preprocessed into a format that is compatible with the implementation of the CB algorithm, the data is given a unique identifier based on the time and date of submission, the project name and the username. Directories are created for the project and the processed files are placed within. A shell script containing hyperparameters of the CB algorithm implementation is generated based on the data submitted and the newly-designated filepaths which can be used to activate the algorithm and generate model predictions.

6.4 Conclusions and Future Work

The curvature and torsion distributions calculated in this project represent a strong improvement in sample population robustness when compared with the original dataset used in Reference [130]. By greatly increasing the number of proteins sampled while ensuring the data used is of high quality, the (κ, τ) distribution can represent rarer, more exotic subsection topologies and show that the most common (κ, τ) combinations are present in a larger sample of proteins. The (κ, τ) distributions also show that helical and sheet subsections show distinct preferences for curvature and torsion, which supports the use of secondary structure in the construction of protein models.

The distance mapping studies show a variation of up to around 1 Å in contacting residue distances based on both residue identity and the distance in sequence space between the contacting residue pairs. The range of distances encountered in the distance mapping indicates that decreasing the optimal distance for contacting residues in the CB algorithm could be beneficial in model refinement. However, the variation of the distances observed when segregated by both residue identity and by sequence space separation also means that imposing harsh penalties for disobeying a contact distance with minimal leeway during model refinement increases the risk of overfitting the model to supplementary information that is ultimately an uncertain prediction in itself.

The prototype SAXShund server provides a simple-to-use interface that guides a user through data submission and automates many of the preprocessing tasks that are necessary to use the CB algorithm. Before deployment, the hardware to facilitate the server must be organised and a process to automate the passing of preprocessed data to the CB algorithm and the subsequent delivery of model predictions to a user must be implemented.

Appendix A

Region	Common Lines
AT1G40104-AT1G40129	6
AT3G32980-AT3G06315	6
AT4G06701-AT4G06744	5
AT3G06355-AT3G42050	5
AT5G05005-AT5G32440	5
AT1G38131-AT1G38790	5
AT5G04965-AT5G04975	4
AT4G06688	3
AT4G06634-AT4G06655	3
AT3G06055-AT3G30705	3
AT4G06583-AT4G06598	3
AT5G60640	3
AT5G60650	3
AT5G60660	3
AT2G07640-AT2G07771	3
AT5G04975-AT5G32017	3
AT3G33520-AT3G41762	3
AT4G03740-AT4G03811	3
AT2G08986	3
AT5G36001-AT5G36080	3
AT2G10260-AT2G05505	3

TABLE A.1: Table of commonly-mutated regions shared between EMS-mutagenised *A. thaliana* lines described in Chapter 4. Regions joined by a hyphen represent intergenic regions.

Appendix B

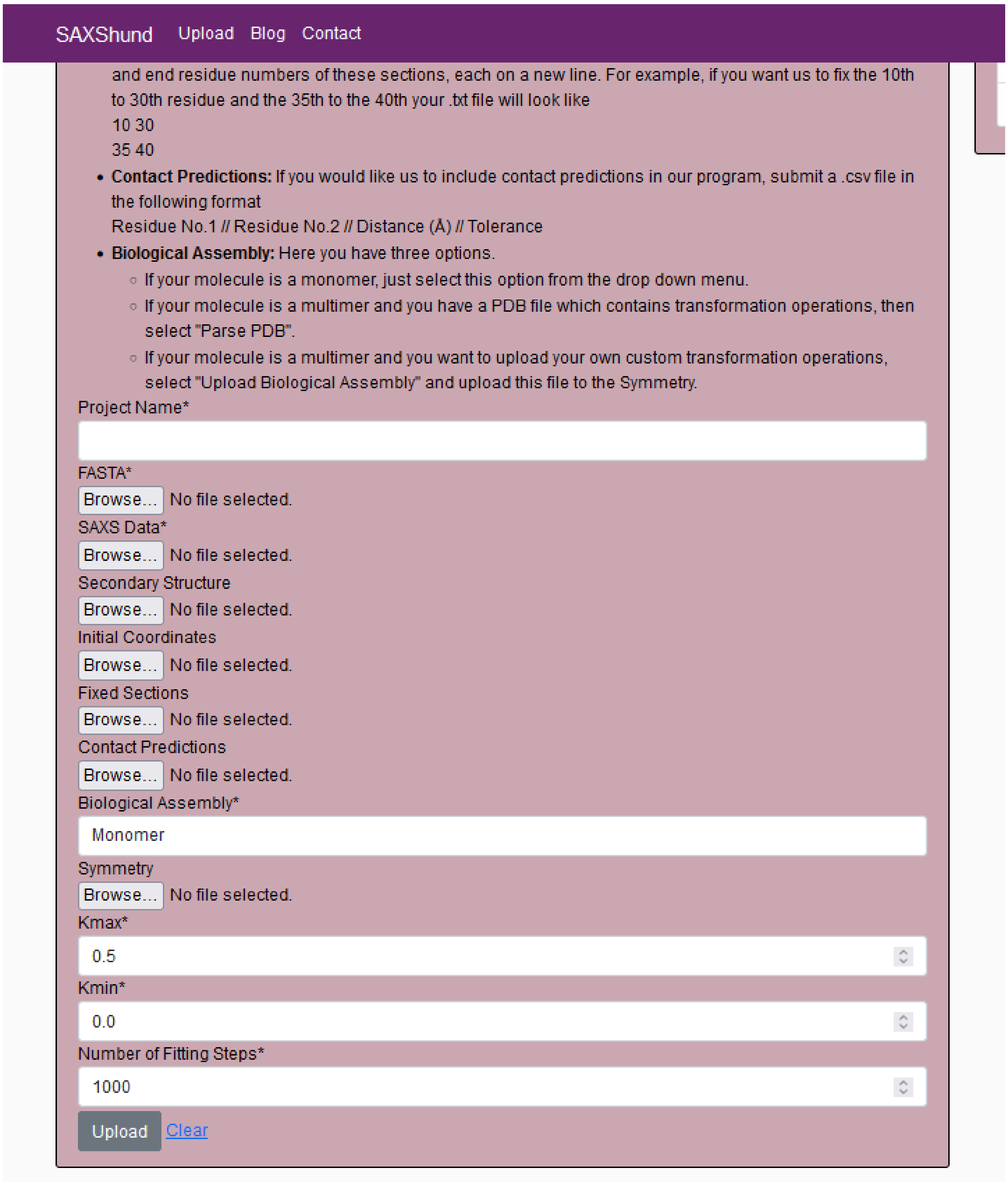

FIGURE B.1: Submission section of the SAXShund website.

The following Python code comprises the set of preprocessing functions that are available to the SAXShund server.

```
import urllib.request
import os
import re
import json
import numpy as np
import pandas as pd
import ast
from Bio import PDB, SeqIO
from Bio.SeqUtils import seq1


def get_unpacked_list(self):
    """
    Returns all atoms from the residue,
    in case of disordered, keep only first alt loc and remove the
↪   alt-loc tag.
    """
    atom_list = self.get_list()
    undisordered_atom_list = []
    for atom in atom_list:
        if atom.is_disordered():
            atom.altloc = " "
            undisordered_atom_list.append(atom)
        else:
            undisordered_atom_list.append(atom)
    return undisordered_atom_list


PDB.Residue.Residue.get_unpacked_list = get_unpacked_list
```

```
def parse_fasta(fname):
    """Make dict of {chain_name: chain_sequence} pairs from fasta
    ↪ format primary sequence file"""
    fasta_dict = dict([(record.id, str(record.seq)) for record in
    ↪ SeqIO.parse(fname, 'fasta')])
    return fasta_dict


def search_pdb(fasta_dict, verbose=True):
    """
    Returns a list of PDBID:Chain pairs from PDB query.

    This uses the updated PDB request protocol active as of 11/11/20.

    :param fasta_dict: a dictionary of (chain_name, chain_sequence)
↪   pairs.
    :return: {Input_chain_name: ('PDBID', 'ChainID')} dict. Returns
↪   dict entry of None for Chain_name if no matches with
    100 % sequence identity of length equal to or greater than the
↪   input sequence are found.
    """

    output_dict = {}

    url = 'https://search.rcsb.org/rcsbsearch/v1/query'

    for key in fasta_dict:
        seq = fasta_dict[key]
        query_text = {
```

```
        "query": {
            "type": "terminal",
            "service": "sequence",
            "parameters": {
                "evalue_cutoff": 1,
                "identity_cutoff": 1.0,
                "target": "pdb_protein_sequence",
                "value": "{}".format(seq)
            }
        },
        "request_options": {
            "scoring_strategy": "sequence"
        },
        "return_type": "polymer_entity"
    }

    # encode query as json
    query_json = json.dumps(query_text)
    query_json = query_json.encode('utf-8')

    # set http request
    req = urllib.request.Request(url, data=query_json)
    # we need to add these headers if we're dealing with a query
    ↪   encoded in this way
    req.add_header('Content-Type', 'application/json;
    ↪   charset=utf-8')
    req.add_header('Content-Length', len(query_json))
    # send the response
    response = urllib.request.urlopen(req)
    result = response.read()
    try:
```

```
            decoded_result_json = json.loads(result)
            idlist = [item["identifier"] for item in
             ↪  decoded_result_json["result_set"]]
            # at the moment, this takes the first item of the list
             ↪  with 100 % identity
            match = tuple(idlist[0].split('_'))
            if verbose:
                print('PDB search returned {} results with 100 %
                 ↪  sequence identity:\n'.format(
                    str(decoded_result_json["total_count"])))
                print(*idlist, '\n')
                print('Chain {} from PDBID {} will be used to assign
                 ↪  secondary '
                      'structure for {}.\n'.format(match[1],
                       ↪  match[0], key))
            output_dict[key] = match

        except json.decoder.JSONDecodeError:
            if verbose:
                print('Failed to retrieve any matches from the PDB
                 ↪  for {}. A call to a secondary structure
                 ↪  prediction '
                      'server will be provided in future versions of
                       ↪  this software, but is currently not '
                      'available.\n'.format(key))
            # output_dict[key] = None
            return None

    return output_dict
```

```
def get_overlap(s1, s2):
    """Returns set of longest common substrings between two input strings. For use with run_dssp. This algorithm will be
    fairly slow with long sequences. Tolerates ambiguous residues denoted with 'X' in either sequence. I'm assuming that
    the case where there are two or more longest common substrings of equal length will be rare enough to discount, so
    in downstream steps I will just take an arbitrary member of the set. To decrease the likelihood of ambiguous resis
    in output, make sure that the user fasta sequence is the s1 parameter."""

    m = len(s1)
    n = len(s2)
    # generates mxn matrix of zeroes
    counter = [[0] * (n + 1) for x in range(m + 1)]
    longest = 0
    lcs_set = set()
    for i in range(m):
        for j in range(n):
            # treats 'X' as wildcard character
            if s1[i] == s2[j] or s1[i] == 'X' or s2[j] == 'X':
                c = counter[i][j] + 1
                counter[i + 1][j + 1] = c
                if c > longest:
                    lcs_set = set()
                    longest = c
                    lcs_set.add(s1[i - c + 1:i + 1])
                elif c == longest:
                    lcs_set.add(s1[i - c + 1:i + 1])
```

```
    assert len(lcs_set) == 1

    return ''.join(lcs_set)


def run_dssp(hit_dict, seq_in):
    """

    :param hit_dict: Dictionary of {input_chain: ('PDBID',
 ↪  'ChainID')} values. Taken from search_pdb method in normal
    protocol.
    :param seq_in: (chain_name, aa_sequence) tuple. Taken from
 ↪  read_fasta method in normal protocol.
    :return: #TODO:
    """

    # important: if function is playing up, check that you're using a
    ↪  release of xssp or w/e they call it now that has
    # mkdssp after compiling. I think the newest release that still
    ↪  has it is 3.0.11 from Nov 2019 that should still
    # be on GitHub. I ran it with 3.0.10. Update the dssp_location
    ↪  variable to match your location too!

    # DSSP assigns 8 different subtypes of secondary structure, but
    ↪  we work with just helices, sheets and linkers
    # create simplification dictionary; go from 8 SSEs to 3 (H, S, -)
    simplify_dict = {
        'H': 'H',
        'B': 'S',
        'E': 'S',
        'G': 'H',
```

```
    'I': 'H',
    'T': '-',
    'S': '-',
    '-': '-'
}

# change the line below to wherever dssp is installed on your
↪ machine
dssp_location = '/home/dan/Apps/xssp-3.0.10/mkdssp'

for chain_in in hit_dict:
    pdb_in = hit_dict[chain_in][0]
    # grab PDB entry from PDB (I think this defaults to wwPDB but
    ↪ this shouldn't matter)
    # default file format warning suppressed
    pdb_list = PDB.PDBList()
    pdb_list.retrieve_pdb_file(pdb_in, file_format='mmCif',
    ↪ overwrite=True, pdir='pdb_downloads')

    fin = os.path.join('pdb_downloads', pdb_in.lower() + '.cif')
    parser = PDB.MMCIFParser()
    structure = parser.get_structure(pdb_in, fin)

    # identify which residue numbers belong to the chain of
    ↪ interest
    chain_number = chain_in.split('_')[-1]

    dssp_output = []
    for model in structure:
        print('\nProcessing model', os.path.split(fin)[-1])
        dssp = PDB.DSSP(model, fin, dssp=dssp_location)
```

```
        for key in list(dssp.keys()):
            record = dssp[key]
            to_add = {
                'Resn': record[1],
                'SSE': simplify_dict[record[2]]
            }
            dssp_output.append(to_add)

    dssp_df = pd.DataFrame(dssp_output)
    dssp_df.to_csv('{}_dssp_df_raw.csv'.format(pdb_in))

    # s1 is the input file's sequence
    s1 = list(seq_in.values())[0]

    # s2 is the entire single letter aa sequence of dssp output
    s2 = ''.join(dssp_df['Resn'].values)

    # find sequence overlap between fasta input and dssp output
    pattern = get_overlap(s1, s2)

    # santity check to make sure that overlap is present in the
     ↪  user sequence
    assert pattern in s1
    # get row numbers of pattern start and end to slice dataframe
    sequence_match_start = s1.index(pattern)
    sequence_match_end = sequence_match_start + len(pattern) - 1

    fout_df = dssp_df.loc[sequence_match_start:sequence_match_end]

    # write a file with sequence name, single letter aa seq,
     ↪  secondary structure prediction in 3 rows.
```

```
        with open('dssp_assignment_{}.txt'.format(pdb_in), 'a') as
         ↪  fout:
            fout.write('{}\n'.format(list(seq_in.keys())[0]))
            fout.write(''.join(fout_df['Resn'].values) + '\n')
            fout.write(list(seq_in.values())[0] + '\n')
            fout.write(''.join(fout_df['SSE'].values))

        print('dssp_assignment_{}.txt generated'.format(pdb_in))

    return 'dssp_assignment_{}.txt'.format(pdb_in)



def parse_pdb_biomts(fin, verbose=False):
    """
    Parses REMARK350 rows of PDB file to get matrix operators to
↪   generate biological assembly from asymmetric unit
    coordinates. The output of this is pretty complicated and
↪   horrible, check the returned variable description below.
    There is definitely a better way of organising this information-
↪   sorry!
    :param fin: opened pdb file
    :param verbose: prints extra information if True
    :return matrix_dict:
    :return all transformations: 3 x n array; pos 0 = biological
↪   assembly label and descriptor (string)
    pos 1 = dictionary of {biomt_number: biomt} pairs, with biomts
↪   being 4x3 arrays
    pos 2 = ([chain_names], [operations_to_perform_on_chains])
↪   tuples. E.g. [(['A, B'], ['1'])] means to perform
    transformation 1 on chains A and B.
    """
```

```
if verbose:
    print('Parsing symmetry operations to generate biological
     ↪  assembly.\n')

remarks = [l[11:].strip() for l in fin.readlines() if
 ↪  l.startswith('REMARK 350')]
all_transformations = []
operations = False
for i, l in enumerate(remarks):
    if l.startswith('BIOMOLECULE:'):
        # records transformation matrices for different
         ↪  assemblies separately
        # example:
        # 'BIOMOLECULE: 1'
        if operations:
            chain_assignments = list(set(operations))
            chain_assignments.sort()
            chain_array.append((chains, chain_assignments))
            all_transformations.append([label, matrix_dict,
             ↪  chain_array])
        assembly = []
        matrix_dict = {}
        chain_array = []
        operations = []
        label = l + ', ' + remarks[i+1]
    elif l.startswith('APPLY THE FOLLOWING TO CHAINS:'):
        # example:
        # 'APPLY THE FOLLOWING TO CHAINS: A, B, C'
        if operations:
            chain_array.append((chains, list(set(operations))))
```

```
                chains = []
            operations = []
            chains = l[31:].split(', ')
        elif l.startswith('AND CHAINS: '):
            # example (stripped):
            # 'AND CHAINS: E, F, G'
            chains.extend(l[12:].split(', '))
        elif l.startswith('BIOMT'):
            # example (remembering REMARK 350 is stripped):
            # 'BIOMT3    1   0.000000   0.000000   1.000000
            ↪   0.00000'
            matrix_dict.setdefault(l[9], []).append(l[11:].split())
            operations.append(l[9])

    chain_assignments = list(set(operations))
    chain_assignments.sort()
    chain_array.append((chains, chain_assignments))
    all_transformations.append([label, matrix_dict, chain_array])

    if verbose:
        print('Biological units found:', len(all_transformations),
        ↪   '\n')
        for transformation in all_transformations:
            print(transformation[0])

    return all_transformations



def parse_custom_biomts(fin, verbose=False):
    """
```

```
    Parses custom user submission to get matrix operators to generate biological assembly from assymetric unit
    coordinates.
    :param fin: unopened csv file containing transformation instructions. Format: first row is a comment containing the
    ([chain_names], [operations_to_perform_on_chains]) tuples. Subsequent rows form 4 x 3n matrix of transformations
    (3x3 rotation matrix and 1x3 translation vector).
    :param verbose: prints extra information if True
    :return all transformations: 3 x n array; pos 0 = biological assembly label and descriptor (string)
    pos 1 = dictionary of {biomt_number: biomt} pairs, with biomts being 4x3 arrays
    pos 2 = ([chain_names], [operations_to_perform_on_chains]) tuples. E.g. [(['A, B'], ['1'])] means to perform
    transformation 1 on chains A and B.
    """
    matrix_dict = {}
    # top row should be a comment with the chain assignments. Read from 2nd character to avoid '#' character
    with open(fin, 'r') as f:
        assignments = ast.literal_eval(f.readlines()[0][1:])
    transformations = np.loadtxt(fin, delimiter=',', skiprows=1)
    assert np.shape(transformations)[0] % 3 == 0, 'Check dimensions of transformation array.'
    for i, row in enumerate(transformations):
        matrix_dict.setdefault(str(int(np.ceil((i+1)/3))), []).append(list(row))

    return ['User-generated Symmetry Operations', matrix_dict, assignments]
```

```
def generate_assembly(asymmetric_unit, symmetry='parse',
↪   verbose=False, unit_test=False):
    """

    :param asymmetric_unit: path to PDB file containing asymmetric
↪   unit. Requires REMARK350 info to generate assembly
    through parsing.
    :param symmetry: string to specify path to file containing
↪   user-made symmetry operation to perform. If 'parse',
    the method will try to parse the BIOMTs from the REMARK350
↪   section of the PDB.
    :param verbose: prints out some extra info to the console if not
↪   None
    :return biological_assembly: pdb file containing coordinates and
↪   secondary structure of full biological assembly
    based on the symmetry operations that are either parsed from the
↪   REMARK350 info or specified by the user
    """
    if verbose:
        print('Parsing file:', asymmetric_unit, '\n')

    # parse symmetry operations
    if symmetry == 'parse':
        if asymmetric_unit.endswith('.pdb') or
        ↪   asymmetric_unit.endswith('.pdbx'):
            with open(asymmetric_unit) as fin1:
                # matrix_dict, assignment_array =
                ↪   parse_pdb_biomts(fin1, verbose=verbose)
```

```
            transformations = parse_pdb_biomts(fin1,
             ↪  verbose=verbose)
            if len(transformations) > 1:
                if not unit_test:
                    print('Multiple possible biological
                     ↪  assemblies detected.')
                    for i, transformation in
                     ↪  enumerate(transformations):
                        print('To select {}, type
                         ↪  {}'.format(transformation[0], i+1))
                user_selection = input('Selected biological
                 ↪  assembly: ')
                user_selection = int(user_selection)
                matrix_dict, assignment_array =
                 ↪  transformations[user_selection-1][1],
                 ↪  transformations[user_selection-1][2]
            else:
                matrix_dict, assignment_array =
                 ↪  transformations[0][1], transformations[0][2]
    else:
        with open(symmetry) as fin2:
            matrix_dict, assignment_array = parse_custom_biomts(fin2,
             ↪  verbose=verbose)

    # apply symmetry operations
    # todo: add some error catching to cope with mmcif files
    parser = PDB.PDBParser(QUIET = True)
    structure = parser.get_structure('structure', asymmetric_unit)
    original_chains = [c.get_id() for c in structure.get_chains()]
    # will update current chains throughout loops
    current_chains = original_chains.copy()
```

```
original_length = len([a for a in structure.get_residues()])
if verbose:
    print('original_chains:', original_chains)
    print('original_length:', original_length)
for assignment_tuple in assignment_array:
    # assumes unit matrix is given as first REMARK350
    for biomt_id in assignment_tuple[1][1:]:
        biomt = np.array(matrix_dict[biomt_id], dtype=float)
        rot = biomt[:, :3]
        trans = biomt[:, 3]
        for chain_assignment in assignment_tuple[0]:
            new_chain_id = chr(ord(max(list(current_chains))) + 1)
            current_chains.append(new_chain_id)
            new_chain = structure[0][chain_assignment].copy()
            new_chain.id = new_chain_id
            new_chain.transform(rot, trans)
            structure[0].add(new_chain)

final_chains = [c.get_id() for c in structure.get_chains()]
final_length = len([a for a in structure.get_residues()])
if verbose:
    print('final_chains:', final_chains)
    print('final length:', final_length)

if unit_test:
    return (original_chains, original_length, final_chains,
     ↪  final_length)

# get path
directory = os.path.split(asymmetric_unit)[:-1]
# write new pdb
```

```
    pdb_name = os.path.split(asymmetric_unit)[-1][:-4]
    io = PDB.PDBIO()
    io.set_structure(structure)
    output_path = os.path.join(*directory,
     ↪  '{}_Preprocessing_Biological_Assembly.pdb'.format(pdb_name))
    io.save(output_path)

    return output_path


def pdb_to_coord(fin, chains='all', homomultimer=False,
 ↪  seq_and_ss=False):
    """
    Takes input of pdb file and spits out Chris format coordinate
 ↪  data. Option to include all or subset of chains
    as an argument.

    :param fin: path to pdb file
    :param chains: optional dictionary of chains for multiple
 ↪  selection (THIS DOESN'T WORK CONSISTENTLY YET)
    :param multimer: set to True if structure is a multimer of
 ↪  single-chain proteins. (CANNOT HANDLE MULTIMERS OF
    MULTI-CHAIN PROTEINS YET).
    :param seq_and_ss: set to True to also generate aa sequence and
 ↪  ss file from pdb info.
    :return: path to coordinates
    :return: path to seq and ss file (if seq_and_ss == True)
    """
    # strip pdb file extension
    structure_name = fin[:-4]
    # standard biopython pdb parse
```

```
parser = PDB.PDBParser(QUIET = True)
structure = parser.get_structure(structure_name, fin)

# some proteins are divided into separate chains in the record
↪ despite being one chain. info in header of pdb.
protein_subchains = []
# work out the gaps in the chain. Segments will be recorded as a
↪ sorted list of (start, end) integer tuples.
segment_terminals = []
# this bit parses the pdb file text line by line to look for
↪ records with specific words at the start and stores the
# info in the protein_subchains and segment_terminals lists.
with open(fin, 'r') as pdb_text:
    text = pdb_text.readlines()
    for line in text:
        if line.startswith('COMPND') and line[11:20] ==
        ↪ 'MOLECULE:':
            protein_name = line[21:].strip().replace(';', '')
        if line.startswith('COMPND') and line[11:17] == 'CHAIN:':
            # regex operation strips non alphanumeric characters
            chain_list = list(re.sub(r'\W+', '',
            ↪ line[18:].strip()))
            protein_subchains.append((protein_name, chain_list))
        if line.startswith('HELIX'):
            segment_start = int(line[21:25].strip())
            segment_end = int(line[33:37].strip())
            segment_type = 'H'
            assert line[19:21].strip() == line[
```

```
                                                      31:33].strip(), 'Chain
                                                   ↪  ID mismatch in
                                                   ↪  HELIX record.
                                                   ↪  Check
                                                   ↪  {}'.format(fin)
                chain_id = line[19:21].strip()
            elif line.startswith('SHEET'):
                segment_start = int(line[22:26].strip())
                segment_end = int(line[33:37].strip())
                segment_type = 'S'
                assert line[21] == line[32], 'Chain ID mismatch in
                 ↪  SHEET record.  Check {}'.format(fin)
                chain_id = line[21]
            else:
                continue
            segment_terminals.append((segment_start, segment_end,
             ↪  segment_type, chain_id))
    # sort terminals so when we loop through them they're in
     ↪  ascending aa order (we filter by chain id later).
    segment_terminals.sort()

    with open('{}_chris_format.txt'.format(structure_name), 'a') as
     ↪  fout:
        for model in structure:
            chain_count = 1
            if chains == 'all':
                chains = list(chain.get_id() for chain in
                 ↪  structure.get_chains())
            else:
                chains = list(chains)
            for chain_id in chains:
```

```
        chain = model[chain_id]
        segment_terminals_subset = [x for x in
         ↪ segment_terminals if x[3] == chain_id]
        # look up sequence id of first resi in chain in case
         ↪ chains don't start at terminal residue
        first_resi = next(chain.get_residues()).id[1]
        resi_count = first_resi
        for resi in chain:
            if resi.has_id("CA"):
                ca = resi["CA"]
                fout.write(
                    str(ca.get_coord()[0]) + ' ' +
                     ↪ str(ca.get_coord()[1]) + ' ' +
                     ↪ str(ca.get_coord()[2]) + '\n')
                resi_count += 1
                prevline = 'coord'
            for terminals in segment_terminals_subset:
                if (resi_count == terminals[0] or resi_count
                 ↪ == terminals[1] + 1) and prevline != '\n':
                    fout.write('\n')
                    prevline = '\n'

        # terminate chains
        if homomultimer:
            fout.write('End of chain
             ↪ {}\n\n'.format(chain_count))
            chain_count += 1
            prevline = '\n'
        else:
            for protein in protein_subchains:
```

```
                    if protein[1][-1] == chain_id:  # todo: this won't work properly if the chain list doesn't contain the terminal chain- make an assertion to stop people doing this
                        fout.write('End of chain {}\n\n'.format(chain_count))
                        chain_count += 1
                        prevline = '\n'
                        break
    fout.close()
print('Coordinate file {}_chris_format.txt created.'.format(structure_name))


if seq_and_ss == True:
    with open('{}_seq_and_ss.txt'.format(structure_name), 'a') as fout:
        chain_info = []
        for model in structure:
            if chains == 'all':
                chains = list(chain.get_id() for chain in structure.get_chains())
            else:
                chains = list(chains)
            for chain_id in chains:
                chain_seq = []
                chain = model[chain_id]
                segment_terminals_subset = [x for x in segment_terminals if x[3] == chain_id]
                print(segment_terminals_subset)
                # adjust segment terminals to match numbering from first resi
```

```
            first_resi = next(chain.get_residues()).id[1]
            for resi in chain:
                if resi.has_id("CA"):
                    chain_seq.append(resi.get_resname())
            chain_seq = ''.join(chain_seq)
            chain_seq = seq1(chain_seq)
            chain_ss = []
            # residues indexed from 1 instead of 0 to match
            ↪   pdb resi numbering
            for resi in range(1, len(chain_seq) + 1):
                ss_element = '-'
                for terminals in segment_terminals_subset:
                    if resi in range(terminals[0] + 1 -
                    ↪   first_resi, terminals[1] + 2 -
                    ↪   first_resi):
                        ss_element = terminals[2]
                        break
                chain_ss.append(ss_element)
            chain_info.append((chain_id, chain_seq, chain_ss))

        for full_chain in protein_subchains:
            print(full_chain)
            if homomultimer:
                chain_ids = list(chains)
            else:
                chain_ids = full_chain[1]
                full_chain_seq = []
                full_chain_ss = []
            for chain_id in chain_ids:
                if homomultimer:
                    full_chain_seq = []
```

```
                        full_chain_ss = []
                    match = [x for x in chain_info if x[0] ==
                    ↪   chain_id]
                    print(match)
                    # assert len(match) == 1, 'Chain ID clashes
                    ↪   detected. Check all chains are labelled
                    ↪   uniquely.'
                    full_chain_seq.append(match[0][1])
                    full_chain_ss.append(''.join(match[0][2]))
                    if homomultimer:
                        fout.write(''.join(full_chain_seq) +
                        ↪   '\n\n')
                        fout.write(''.join(full_chain_ss) +
                        ↪   '\n\n')
                if not homomultimer:
                    fout.write(''.join(full_chain_seq) + '\n\n')
                    fout.write(''.join(full_chain_ss) + '\n\n')

        print('Sequence and secondary structure file
        ↪   {}_seq_and_ss.txt generated.'.format(structure_name))

        return '{}_chris_format.txt'.format(structure_name),
        ↪   '{}_seq_and_ss.txt'.format(structure_name)

    else:
        return '{}_chris_format.txt'.format(structure_name)


def make_fingerprint(fasta_in, ss_in):
    """
```

```
    Generates fingerprint file from fasta and secondary structure
↪   files.

    :param fasta_in: path to fasta file
    :param ss_in: path to secondary structure file
    :return: path to fingerprint file
    """

    sequences = [str(entry.seq) for entry in
     ↪  list(SeqIO.parse(fasta_in, 'fasta'))]
    with open(ss_in, 'r') as fin:
        secondaries = list(filter(None, fin.read().splitlines()))
    assert len(sequences) == len(secondaries)
    data_dir = os.path.dirname(fasta_in)
    #project_root = os.path.dirname(data_dir)
    output_filename = os.path.join(data_dir, 'preprocessed',
     ↪  'Fingerprint.dat')
    with open(output_filename, 'w+') as fout:
        fout.write(str(len(sequences))+'\n\n')
        for seq, ss in zip(sequences, secondaries):
            assert len(seq) == len(ss)
            fout.write(seq+'\n\n')
            fout.write(ss+'\n\n')

    return output_filename



def write_sh_file(project_name, preprocessed_data_dir, saxs_data_path,
fingerprint_path, coords_path = None, contacts_path = None,
fixed_sections_path = None, kmin = 1, kmax = 10, fit_steps = 1000):
```

```
    """Writes the config shell script based on parameters given by
    ↪  the user on data upload.

    Args:
        project_name (str): [description]
        preprocessed_data_dir (str): [description]
        saxs_data_path (str): [description]
        fingerprint_path (str): [description]
        coords_path (str, optional): [description]. Defaults to None.
        contacts_path (str, optional): [description]. Defaults to
↪   None.
        fixed_sections_path (str, optional): [description]. Defaults
↪   to None.
        kmin (int, optional): [description]. Defaults to 1.
        kmax (int, optional): [description]. Defaults to 10.
        fit_steps (int, optional): [description]. Defaults to 1000.

    Returns:
        output_filepath (str): path to the finished shell script
    """
    project_root = os.path.dirname(preprocessed_data_dir)
    output_filepath = os.path.join(preprocessed_data_dir,
    ↪  project_name + '_config.sh')
    with open(output_filepath, 'w') as fout:
            fout.write('#!/bin/bash')
            fout.write('\nScatterFile={}'.format(saxs_data_path))
            fout.write('\nSequenceFile={}'.format(fingerprint_path))
            if coords_path is not None:
                    fout.write('\ninitialCoordsFile={}'.format(coords
                    ↪  _path))
            else:
```

```
            fout.write('\ninitialCoordsFile=none')
        if contacts_path is not None:
            fout.write('\npairedPredictions={}'.format(contac⌋
            ↪ ts_path))
        else:
            fout.write('\npairedPredictions=none')
        if fixed_sections_path is not None:
            fout.write('\nfixedsections={}'.format(fixed_sect⌋
            ↪ ions_path))
        else:
            fout.write('\nfixedsections=none')
    # no options for crystal symmetry or or hydro cover in
    ↪ prototype
        fout.write('\ncrystalSymmetry=none')
        fout.write('\nwithinMonomerHydroCover=none')
        fout.write('\nbetweenMonomerHydroCover=none')
        fout.write('\nkmin={};'.format(str(kmin)))
        fout.write('\nkmax={};'.format(str(kmax)))
        fout.write('\nmaxNoFitSteps={}'.format(str(fit_steps)))
        fout.write('\n\nmkdir {}/outputs'.format(project_root))
        fout.write('\n\n\nfor i in {1..40}')
        fout.write('\ndo')
        fout.write('\n    predictStructure $ScatterFile
        ↪ $SequenceFile '
        '$initialCoordsFile $pairedPredictions $fixedsections
        ↪ $crystalSymmetry '
        '$withinMonomerHydroCover $betweenMonomerHydroCover $kmin
        ↪ $kmax '
        '$maxNoFitSteps {}/outputs/mol$i.dat
        ↪ {}/outputs/scatter$i.dat'.format(project_root,
        ↪ project_root))
```

```
            fout.write('\ndone')


    return output_filepath
```

The following is an example config shell script that is generated by SAXShund.

```
#!/bin/bash
ScatterFile=D:/dan_preprocessing_validation/test_data/pyl2_SS.txt
SequenceFile=D:/dan_preprocessing_validation/test_data/pyl2.fasta
initialCoordsFile=none
pairedPredictions=none
fixedsections=none
crystalSymmetry=none
withinMonomerHydroCover=none
betweenMonomerHydroCover=none
kmin=1;
kmax=10;
maxNoFitSteps=1000

mkdir dan_preprocessing_validation/dan_project_1/outputs

for i in {1..40}
do
    predictStructure $ScatterFile $SequenceFile $initialCoordsFile
     ↪  $pairedPredictions $fixedsections $crystalSymmetry
     ↪  $withinMonomerHydroCover $betweenMonomerHydroCover $kmin
     ↪  $kmax $maxNoFitSteps /outputs/mol$i.dat /outputs/scatter$i.dat
done
```

Bibliography

1. Sukiran, N. *Exploring plant hormonal signalling through chemical perturbation* PhD thesis (Durham University, 2018).
2. Harris, D *et al.* On-farm seed priming: using participatory methods to revive and refine a key technology. *Agricultural systems* **69,** 151–164 (2001).
3. Zheng, M. *et al.* Seed priming in dry direct-seeded rice: consequences for emergence, seedling growth and associated metabolic events under drought stress. *Plant growth regulation* **78,** 167–178 (2016).
4. Umezawa, T. *et al.* Molecular basis of the core regulatory network in aba responses: sensing, signaling and transport. *Plant and cell physiology* **51,** 1821–1839 (2010).
5. Khan, A. *et al.* Coping with drought: stress and adaptive mechanisms, and management through cultural and molecular alternatives in cotton as vital constituents for plant stress resilience and fitness. *Biological research* **51** (2018).
6. Leng, P., Yuan, B. & Guo, Y. The role of abscisic acid in fruit ripening and responses to abiotic stress. *Journal of experimental botany* **65,** 4577–4588 (2013).
7. Pantoliano, M. W. *et al.* High-density miniaturized thermal shift assays as a general strategy for drug discovery. *Journal of biomolecular screening* **6,** 429–440 (2001).
8. Grøftehauge, M. K., Hajizadeh, N. R., Swann, M. J. & Pohl, E. Protein-ligand interactions investigated by thermal shift assays (TSA) and dual polarization interferometry (DPI). *Acta crystallographica section D: biological crystallography* **71,** 36–44 (2015).

9. Bruce, D., Cardew, E., Freitag-Pohl, S. & Pohl, E. How to stabilize protein: stability screens for thermal shift assays and nano differential scanning fluorimetry in the Virus-X project. *Journal of visualized experiments,* 58666 (Feb. 2019).

10. Fägerstam, L. G., Frostell-Karlsson, Å., Karlsson, R., Persson, B. & Rönnberg, I. Biospecific interaction analysis using surface plasmon resonance detection applied to kinetic, binding site and concentration analysis. *Journal of chromatography A* **597,** 397–410 (1992).

11. Wienken, C. J., Baaske, P., Rothbauer, U., Braun, D. & Duhr, S. Protein-binding assays in biological liquids using microscale thermophoresis. *Nature communications* **1,** 100 (2010).

12. López-Méndez, B., Uebel, S., Lundgren, L. P. & Sedivy, A. Microscale thermophoresis and additional effects measured in NanoTemper Monolith instruments. *European biophysics journal* **50,** 653–660 (2021).

13. Jumper, J. *et al.* Highly accurate protein structure prediction with AlphaFold. *Nature* **596,** 583–589 (Aug. 2021).

14. Baek, M. *et al.* Accurate prediction of protein structures and interactions using a three-track neural network. *Science* **373,** 871–876 (2021).

15. Abramson, J. *et al.* Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature,* 1–3 (2024).

16. Varadi, M. *et al.* AlphaFold protein structure database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic acids research* **50,** D439–D444 (2022).

17. Brookes, E., Rocco, M., Vachette, P. & Trewhella, J. AlphaFold-predicted protein structures and small-angle x-ray scattering: insights from an extended examination of selected data in the Small-Angle Scattering Biological Data Bank. *Journal of applied crystallography* **56,** 910–926 (2023).

18. Berman, H. M. *et al.* The protein data bank. *Nucleic acids research* **28,** 235–242 (2000).

19. Jerabek-Willemsen, M., Wienken, C. J., Braun, D., Baaske, P. & Duhr, S. Molecular interaction studies using microscale thermophoresis. *Assay and drug development technologies* **9,** 342–353 (Aug. 2011).

20. Murashige, T. & Skoog, F. A revised medium for rapid growth and bio assays with tobacco tissue cultures. en. *Physiologia plantarum* **15,** 473–497 (1962).

21. Schneider, C. A., Rasband, W. S. & Eliceiri, K. W. NIH image to ImageJ: 25 years of image analysis. en. *Nature methods* **9,** 671–675 (July 2012).

22. Howe, K. L. *et al.* Ensembl 2021. *Nucleic acids research* **49,** D884–D891 (Jan. 2021).

23. Andrews, S. *FastQC: a quality control tool for high throughput sequence data* 2010.

24. Ewels, P., Magnusson, M., Lundin, S. & Käller, M. MultiQC: summarize analysis results for multiple tools and samples in a single report. *Bioinformatics* **32,** 3047–3048 (Oct. 2016).

25. Li, H. Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. *arXiv preprint arXiv:1303.3997* (2013).

26. Danecek, P. *et al.* Twelve years of SAMtools and BCFtools. eng. *Gigascience* **10,** giab008 (Feb. 2021).

27. McKenna, A. *et al.* The genome analysis toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. en. *Genome research* **20,** 1297–1303 (Jan. 2010).

28. Pedersen, B. S. & Quinlan, A. R. Mosdepth: quick coverage calculation for genomes and exomes. *Bioinformatics* **34,** 867–868 (Oct. 2017).

29. Cingolani, P. *et al.* A program for annotating and predicting the effects of single nucleotide polymorphisms, SnpEff: SNPs in the genome of *Drosophila melanogaster* strain w1118; iso-2; iso-3. *Fly* **6,** 80–92 (2012).

30. Cingolani, P. *et al.* Using *drosophila melanogaster* as a model for genotoxic chemical mutational studies with a new program, SnpSift. *Frontiers in genetics* **3** (2012).

31. Cock, P. J. *et al.* Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics* **25,** 1422–1423 (2009).

32. Schaarschmidt, J., Monastyrskyy, B., Kryshtafovych, A. & Bonvin, A. M. Assessment of contact predictions in CASP12: co-evolution and deep learning coming of age. *Proteins* **86,** 51–66 (Mar. 2018).

33. Kabsch, W. & Sander, C. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers: original research on biomolecules* **22,** 2577–2637 (1983).

34. Touw, W. G. *et al.* A series of PDB-related databanks for everyday needs. *Nucleic acids research* **43,** D364–D368 (2015).

35. Bradski, G. The OpenCV library. *Dr. dobb's journal of software tools* (2000).

36. Paszke, A. *et al.* in *Advances in neural information processing systems 32* 8024–8035 (Curran associates, Inc., 2019).

37. Ren, S., He, K., Girshick, R. & Sun, J. Faster R-CNN: towards real-time object detection with region proposal networks. *Advances in neural information processing systems* **28** (2015).

38. Schwartz, S. H., Tan, B. C., Gage, D. A., Zeevaart, J. A. D. & McCarty, D. R. Specific oxidative cleavage of carotenoids by VP14 of maize. *Science* **276,** 1872–1874 (1997).

39. Park, S.-y. *et al.* Abscisic acid inhibits type 2C protein phosphatases via the PYR/PYL family of START proteins. *Science* **324,** 1068–1069 (2009).

40. Ma, Y. *et al.* Regulators of PP2C phosphatase activity function as abscisic acid sensors. *Science* **324,** 1064–1069 (2009).

41. Yin, P. *et al.* Structural insights into the mechanism of abscisic acid signaling by PYL proteins. *Nature structural & molecular biology* **16,** 1230–1236 (Dec. 2009).

42. Xu, Z.-J., Nakajima, M., Suzuki, Y. & Yamaguchi, I. Cloning and characterization of the abscisic acid-specific glucosyltransferase gene from adzuki bean seedlings. *Plant physiology* **129,** 1285–95 (July 2002).

43. Saito, S. *et al.* *Arabidopsis* CYP707As encode (+)-abscisic acid 8′-hydroxylase, a key enzyme in the oxidative catabolism of abscisic acid. *Plant physiology* **134,** 1439–49 (Apr. 2004).

44. Brabham, D. E. & Biggs, R. H. Cis-trans photoisomerization of abscisic acid. *Photochemistry and photobiology* **34,** 33–37 (July 1981).

45. Liu, W. C., Carnsdagger, H. R., Addicott, F. T. & Smith, O. E. Isolation of abscisin, an abscission accelerating substance. *Science (New York, N.Y.)* **134,** 384–5 (Aug. 1961).

46. Eagles, C. F. & Wareing, P. F. Dormancy regulators in woody plants: experimental induction of dormancy in *Betula pubescens*. *Nature* **199,** 874–875 (Aug. 1963).

47. Porterab, N. G. & van Steveninckab, R. An abscission-promoting factor in *Lupinus luteus*. *Life sciences* **5,** 2301–2308 (Dec. 1966).

48. Szostkiewicz, I. *et al.* Closely related receptor complexes differ in their ABA selectivity and sensitivity. *The plant journal* **61,** 25–35 (Jan. 2010).

49. Hao, Q. *et al.* The molecular basis of ABA-independent inhibition of PP2Cs by a subclass of PYL proteins. *Molecular cell* **42,** 662–672 (June 2011).

50. Fujii, H., Verslues, P. E. & Zhu, J.-K. Identification of two protein kinases required for abscisic acid regulation of seed germination, root growth, and gene expression in *Arabidopsis*. *The plant cell online.*

51. Melcher, K. *et al.* Identification and mechanism of ABA receptor antagonism. *Nature structural & molecular biology* **17,** 1102–1108 (2010).

52. Park, S.-Y. *et al.* Agrochemical control of plant water use using engineered abscisic acid receptors. *Nature* **520,** 545–548 (2015).

53. Nishimura, N. *et al.* Structural mechanism of abscisic acid binding and signaling by dimeric PYR1. **326,** 1373–1379 (2010).

54. Santiago, J. *et al.* Modulation of drought resistance by the abscisic acid receptor PYL5 through inhibition of clade A PP2Cs. *Plant journal* **60,** 575–588 (2009).

55. Meyer, K, Leube, M. P. & Grill, E. A protein phosphatase 2C involved in ABA signal transduction in *Arabidopsis thaliana*. *Science* **264,** 1452–5 (1994).

56. Leung, J. *et al.* *Arabidopsis* ABA response gene ABI1 : features of a calcium-modulated protein phosphatase. *Science* **264,** 1448–1452 (1994).

57. Mustilli, A.-C., Merlot, S., Vavasseur, A., Fenzi, F. & Giraudat, J. *Arabidopsis* OST1 protein kinase mediates the regulation of stomatal aperture by abscisic acid and acts upstream of reactive oxygen species production. *The plant cell* **14,** 3089–99 (Dec. 2002).

58. Yoshida, R. *et al.* The regulatory domain of SRK2E/OST1/SnRK2.6 interacts with ABI1 and integrates abscisic acid (ABA) and osmotic stress signals controlling stomatal closure in *Arabidopsis*. *Journal of biological chemistry* **281,** 5310–5318 (Feb. 2006).

59. Ng, L.-M. *et al.* Structural basis for basal activity and autoactivation of abscisic acid (ABA) signaling SnRK2 kinases. *Proceedings of the national academy of sciences of the United States of America* **108,** 21259–64 (Dec. 2011).

60. Guiltinan, M. J., Marcotte, W. R. & Quatrano, R. S. A plant leucine zipper protein that recognizes an abscisic acid response element. *Science* **250** (1990).

61. Finkelstein, R. R. & Lynch, T. J. The *Arabidopsis* abscisic acid response gene ABI5 encodes a basic leucine zipper transcription factor. *The plant cell* **12,** 599–609 (2000).

62. Uno, Y *et al. Arabidopsis* basic leucine zipper transcription factors involved in an abscisic acid-dependent signal transduction pathway under drought and high-salinity conditions. *Proceedings of the national academy of sciences of the United States of America* **97,** 11632–7 (Oct. 2000).

63. Furihata, T. *et al.* Abscisic acid-dependent multisite phosphorylation regulates the activity of a transcription activator AREB1. *Proceedings of the national academy of sciences* **103,** 1988–1993 (Feb. 2006).

64. Geiger, D. *et al.* Activity of guard cell anion channel SLAC1 is controlled by drought-stress signaling kinase-phosphatase pair. *Proceedings of the national academy of sciences of the United States of America* **106,** 21425–30 (Dec. 2009).

65. Geiger, D *et al.* Guard cell anion channel SLAC1 is regulated by CDPK protein kinases with distinct Ca2+ affinities. *Proceedings of the national academy of sciences of the United States of America* **107,** 8023–8 (Apr. 2010).

66. Weiner, J. J., Peterson, F. C., Volkman, B. F. & Cutler, S. R. Structural and functional insights into core ABA signaling. *Current opinion in plant biology* **13,** 495–502 (Oct. 2010).

67. Ponting, C. P. & Aravind, L. START: a lipid-binding domain in StAR, HD-ZIP and signalling proteins. *Trends in biochemical sciences* **24,** 130–132 (1999).

68. McWilliam, H. *et al.* Analysis tool web services from the EMBL-EBI. *Nucleic acids research* **41** (2013).

69. Sievers, F. *et al.* Fast, scalable generation of high-quality protein multiple sequence alignments using clustal omega. *Molecular systems biology* **7,** 539–539 (2014).

70. Li, W. *et al.* The EMBL-EBI bioinformatics web and programmatic tools framework. *Nucleic acids research* **43,** W580–W584 (2015).

71. Robert, X. & Gouet, P. Deciphering key features in protein structures with the new ENDscript server. *Nucleic acids research* **42,** W320–W324 (July 2014).

72. Melcher, K. *et al.* A gate-latch-lock mechanism for hormone signalling by abscisic acid receptors. *Nature* **462,** 602–608 (2009).

73. Li, W. *et al.* Molecular basis for the selective and ABA-independent inhibition of PP2CA by PYL13. *Cell research* **23,** 1369–1379 (Dec. 2013).

74. Peterson, F. C. *et al.* Structural basis for selective activation of ABA receptors. *Nature publishing group* **17** (2010).

75. Seidel, S. A. *et al.* Microscale thermophoresis quantifies biomolecular interactions under previously challenging conditions. *Methods* **59,** 301–315 (Mar. 2013).

76. Zhang, X. *et al.* Complex structures of the abscisic acid receptor PYL3/RCAR13 reveal a unique regulatory mechanism. *Structure* **20,** 780–790 (May 2012).

77. Reuven, J. *Probing the mode of action of eW5 - a small molecule plant growth promoter* PhD thesis (Durham University, 2022).

78. Chen, K. *et al.* Abscisic acid dynamics, signaling, and functions in plants. *Journal of integrative plant biology* **62,** 25–54 (2020).

79. Friml, J. Auxin transport - shaping the plant. eng. *Current opinion in plant biology* **6,** 7–12 (Feb. 2003).
80. MacMillan, J. & Suter, P. J. The occurrence of gibberellin A1 in higher plants: isolation from the seed of runner bean (*phaseolus multiflorus*). en. *Naturwissenschaften* **45,** 46–46 (Jan. 1958).
81. Hedden, P. & Sponsel, V. A century of gibberellin research. eng. *Journal of plant growth regulation* **34,** 740–760 (2015).
82. Harberd, N. P. Relieving DELLA restraint. *Science* **299,** 1853–1854 (Mar. 2003).
83. Ruan, J. *et al.* Jasmonic acid signaling pathway in plants. *International journal of molecular sciences* **20,** 2479 (May 2019).
84. Chinnusamy, V., Schumaker, K. & Zhu, J.-K. Molecular genetic perspectives on cross-talk and specificity in abiotic stress signalling in plants. *Journal of experimental botany* **55,** 225–236 (Jan. 2004).
85. Sahu, P. K. *et al.* Next generation sequencing based forward genetic approaches for identification and mapping of causal mutations in crop plants: a comprehensive review. en. *Plants* **9,** 1355 (Oct. 2020).
86. Sega, G. A. A review of the genetic effects of ethyl methanesulfonate. eng. *Mutation research* **134,** 113–142 (Nov. 1984).
87. Maglott, D. Entrez gene: gene-centered information at NCBI. *Nucleic acids research* **33,** D54–D58 (Dec. 2004).
88. Klepikova, A. V., Logacheva, M. D., Dmitriev, S. E. & Penin, A. A. RNA-seq analysis of an apical meristem time series reveals a critical point in *arabidopsis thaliana* flower initiation. *BMC genomics* **16,** 466 (June 2015).
89. Klepikova, A. V., Kasianov, A. S., Gerasimov, E. S., Logacheva, M. D. & Penin, A. A. A high resolution map of the *Arabidopsis thaliana* developmental transcriptome based on RNA-seq profiling. en. *The plant journal* **88,** 1058–1070 (2016).
90. Jarad, M. *et al.* The lamin-like LITTLE NUCLEI 1 (LINC1) regulates pattern-triggered immunity and jasmonic acid signaling. eng. *Frontiers in plant science* **10,** 1639 (2019).

91. Fukao, Y., Hayashi, M. & Nishimura, M. Proteomic analysis of leaf peroxisomal proteins in greening cotyledons of *arabidopsis thaliana*. eng. *Plant & cell physiology* **43,** 689–696 (July 2002).

92. Patil, S. S., Prashant, R., Kadoo, N. Y., Upadhyay, A. & Gupta, V. S. Global study of MFS superfamily transporters in *Arabidopsis* and grapes reveals their functional diversity in plants. en. *Plant gene* **18,** 100179 (June 2019).

93. Tamaki, H. *et al.* Identification of novel meristem factors involved in shoot regeneration through the analysis of temperature-sensitive mutants of *Arabidopsis*. eng. *The plant journal: for cell and molecular biology* **57,** 1027–1039 (Mar. 2009).

94. Sakamoto, T., Kamiya, N., Ueguchi-Tanaka, M., Iwahori, S. & Matsuoka, M. KNOX homeodomain protein directly suppresses the expression of a gibberellin biosynthetic gene in the tobacco shoot apical meristem. en. *Genes & development* **15,** 581–590 (Jan. 2001).

95. Hay, A. *et al.* The gibberellin pathway mediates KNOTTED1-type homeobox function in plants with different body plans. en. *Current biology* **12,** 1557–1565 (Sept. 2002).

96. Chen, H., Banerjee, A. K. & Hannapel, D. J. The tandem complex of BEL and KNOX partners is required for transcriptional repression of ga20ox1. en. *The plant journal* **38,** 276–284 (2004).

97. Jasinski, S. *et al.* KNOX action in *Arabidopsis* is mediated by coordinate regulation of cytokinin and gibberellin activities. en. *Current biology* **15,** 1560–1565 (Sept. 2005).

98. Yanai, O. *et al. Arabidopsis* KNOXI proteins activate cytokinin biosynthesis. en. *Current biology* **15,** 1566–1571 (Sept. 2005).

99. Blatch, G. L. & Lässle, M. The tetratricopeptide repeat: a structural motif mediating protein-protein interactions. en. *BioEssays* **21,** 932–939 (1999).

100. Dittmer, S., Kleine, T. & Schwenkert, S. The TPR- and J-domain-containing proteins DJC31 and DJC62 are involved in abiotic stress responses in *Arabidopsis thaliana*. *Journal of cell science* **134,** jcs259032 (Oct. 2021).

101. Sun, J. *et al.* WRKY41/WRKY46-miR396b-5p-TPR module mediates abscisic acid-induced cold tolerance of grafted cucumber seedlings. *Frontiers in plant science* **13** (2022).

102. Kim, J.-E., Cheng, K. M., Craft, N. E., Hamberger, B. & Douglas, C. J. Over-expression of *arabidopsis thaliana* carotenoid hydroxylases individually and in combination with a beta-carotene ketolase provides insight into in vivo functions. eng. *Phytochemistry* **71,** 168–178 (Feb. 2010).

103. Chen, C. *et al. Arabidopsis* SAG protein containing the MDN1 domain participates in seed germination and seedling development by negatively regulating ABI3 and ABI5. *Journal of experimental botany* **65,** 35–45 (Jan. 2014).

104. Li, W. *et al.* NAC family transcription factors in tobacco and their potential role in regulating leaf senescence. *Frontiers in plant science* **9** (2018).

105. Furutani, M. *et al.* Polar recruitment of RLD by LAZY1-like protein during gravity signaling in root branch angle control. en. *Nature communications* **11,** 76 (Jan. 2020).

106. Digby, J. & Firn, R. D. The gravitropic set-point angle (GSA): the identification of an important developmentally controlled variable governing plant architecture. eng. *Plant, cell & environment* **18,** 1434–1440 (Dec. 1995).

107. Roychoudhry, S., Del Bianco, M., Kieffer, M. & Kepinski, S. Auxin controls gravitropic setpoint angle in higher plant lateral branches. English. *Current biology* **23,** 1497–1504 (Aug. 2013).

108. Zhao, Y. *et al.* The ABA receptor PYL8 promotes lateral root growth by enhancing MYB77-dependent transcription of auxin-responsive genes. *Science signaling* **7,** ra53–ra53 (2014).

109. Jander, G. *et al.* Ethylmethanesulfonate saturation mutagenesis in *Arabidopsis* to determine frequency of herbicide resistance. *Plant physiology* **131,** 139–146 (2003).

110. Abadi, M. *et al. TensorFlow: large-scale machine learning on heterogeneous systems* Software available from tensorflow.org. 2015.

111. Chollet, F. *et al. Keras* https://github.com/fchollet/keras.

112. Dobos, O., Horvath, P., Nagy, F., Danka, T. & Viczián, A. A deep learning-based approach for high-throughput hypocotyl phenotyping. *Plant physiology* **181,** 1415–1424 (2019).

113. Hubel, D. H. & Wiesel, T. N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The journal of physiology* **160,** 106–1542 (Jan. 1962).

114. Deng, J. *et al. Imagenet: a large-scale hierarchical image database* in *2009 IEEE conference on computer vision and pattern recognition* (2009), 248–255.

115. Shorten, C. & Khoshgoftaar, T. M. A survey on image data augmentation for deep learning. *Journal of big data* **6,** 60 (July 2019).

116. Zhuang, F. *et al.* A comprehensive survey on transfer learning. *Proceedings of the IEEE* **109,** 43–76 (Jan. 2021).

117. Bird, J. J., Kobylarz, J., Faria, D. R., Ekárt, A. & Ribeiro, E. P. Cross-domain MLP and CNN transfer learning for biological signal processing: EEG and EMG. *IEEE access* **8,** 54789–54801 (2020).

118. Li, Z., Liu, F., Yang, W., Peng, S. & Zhou, J. A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE transactions on neural networks and learning systems,* 1–21 (2021).

119. Dumoulin, V. & Visin, F. *A guide to convolution arithmetic for deep learning* tech. rep. arXiv:1603.07285 [cs, stat] type: article (arXiv, Jan. 2018).

120. Agarap, A. F. *Deep learning using rectified linear units (ReLU)* tech. rep. arXiv:1803.08375 [cs, stat] type: article (arXiv, Feb. 2019).

121. Dubey, A. K. & Jain, V. *Comparative study of convolution neural network 's ReLU and leaky-ReLU activation functions* en. in *Applications of computing, automation and wireless systems in electrical engineering* (eds Mishra, S., Sood, Y. R. & Tomar, A.) (Springer, Singapore, 2019), 873–880.

122. Canny, J. A computational approach to edge detection. *IEEE transactions on pattern analysis and machine intelligence* **PAMI-8,** 679–698 (Nov. 1986).

123. Girshick, R., Donahue, J., Darrell, T. & Malik, J. Region-based convolutional networks for accurate object detection and segmentation. *IEEE transactions on pattern analysis and machine intelligence* **38,** 142–158 (Jan. 2016).

124. Girshick, R. *Fast R-CNN* in *Proceedings of the IEEE international conference on computer vision (ICCV)* (Dec. 2015).

125. He, K., Gkioxari, G., Dollar, P. & Girshick, R. *Mask R-CNN* in *Proceedings of the IEEE international conference on computer vision (ICCV)* (Oct. 2017).

126. Gkioxari, G., Malik, J. & Johnson, J. *Mesh R-CNN* in *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)* (Oct. 2019).

127. Leibe, B., Seemann, E. & Schiele, B. *Pedestrian detection in crowded scenes* in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)* **1.** ISSN: 1063-6919 (June 2005), 878–885 vol. 1.

128. Stewart, R., Andriluka, M. & Ng, A. Y. *End-to-end people detection in crowded scenes* in (2016), 2325–2333.

129. Chu, X., Zheng, A., Zhang, X. & Sun, J. *Detection in crowded scenes: one proposal, multiple predictions* in (2020), 12214–12223.

130. Prior, C., Davies, O. R., Bruce, D. & Pohl, E. Obtaining tertiary protein structures by the ab initio interpretation of small angle X-ray scattering data. *Journal of chemical theory and computation* **16,** 1985–2001 (Mar. 2020).

131. Cowieson, N. P. *et al.* Beamline B21: high-throughput small-angle X-ray scattering at Diamond Light Source. *Journal of synchrotron radiation* **27,** 1438–1446 (Sept. 2020).

132. David, G. & Pérez, J. Combined sampler robot and high-performance liquid chromatography: a fully automated system for biological small-angle X-ray scattering experiments at the Synchrotron SOLEIL SWING beamline. *Journal of applied crystallography* **42,** 892–900 (Oct. 2009).

133. Cordeiro, T. N. *et al.* Small-angle scattering studies of intrinsically disordered proteins and their complexes. en. *Current opinion in structural biology. Folding and binding • Proteins: Bridging theory and experiment* **42,** 15–23 (Feb. 2017).

134. Brosey, C. A. & Tainer, J. A. Evolving SAXS versatility: solution X-ray scattering for macromolecular architecture, functional landscapes, and integrative structural biology. en. *Current opinion in structural biology. Cryo electron microscopy ? Biophysical and computational methods ? Biophysical and computational methods - Part B* **58,** 197–213 (Oct. 2019).

135. Putnam, C. D., Hammel, M., Hura, G. L. & Tainer, J. A. X-ray solution scattering (SAXS) combined with crystallography and computation: defining accurate macromolecular structures, conformations and assemblies in solution. en. *Quarterly reviews of biophysics* **40,** 191–285 (Aug. 2007).

136. Manalastas-Cantos, K. *et al.* ATSAS 3.0: expanded functionality and new tools for small-angle scattering data analysis. *Journal of applied crystallography* **54,** 343–355 (Feb. 2021).

137. Svergun, D. I. Restoring low resolution structure of biological macromolecules from solution scattering using simulated annealing. eng. *Biophysical journal* **76,** 2879–2886 (June 1999).

138. Franke, D. & Svergun, D. I. DAMMIF, a program for rapid ab-initio shape determination in small-angle scattering. en. *Journal of applied crystallography* **42,** 342–346 (Apr. 2009).

139. Svergun, D. I., Petoukhov, M. V. & Koch, M. H. Determination of domain structure of proteins from X-ray solution scattering. eng. *Biophysical journal* **80,** 2946–2953 (June 2001).

140. Ramachandran, G. N., Ramakrishnan, C. & Sasisekharan, V. Stereochemistry of polypeptide chain configurations. en. *Journal of molecular biology* **7,** 95–99 (July 1963).

141. Debye, P. Zerstreuung von röntgenstrahlen. en. *Annalen der physik* **351,** 809–823 (1915).

142. Kryshtafovych, A., Schwede, T., Topf, M., Fidelis, K. & Moult, J. Critical assessment of methods of protein structure prediction (CASP) - round XIII. *Proteins* **87,** 1011–1020 (Dec. 2019).

143. Forbes, C. R. *et al.* Insights into thiol-aromatic interactions: a stereoelectronic basis for S-H/p interactions. *Journal of the American chemical society* **139,** 1842–1855 (Feb. 2017).