Design and discovery of novel bacterial nanocompartment proteins using deep learning tools

Design and Discovery of Novel Bacterial Nanocompartment Proteins Using Deep Learning Tools

Author: Naail Kashif-Khan

Primary supervisors:

Dr Renos Savva

Prof. Katherine Thompson

***Chair*:** Prof. Mark Williams

Secondary supervisor:

Dr Stefanie Frank

This dissertation is submitted for the degree of Doctor of Philosophy

Birkbeck, University of London

Abstract

Design and Discovery of Novel Bacterial Nanocompartment Proteins Using Deep Learning Tools

Naail Kashif-Khan

Encapsulins are prokaryotic protein-based organelles. These icosahedral protein compartments display diverse natural functions, including mineral storage and stress responses. Encapsulins also have applications in synthetic biology, drug delivery, vaccines, and metabolic engineering. There are over 6000 known encapsulins, but only a small sample have been characterised experimentally. The work presented probes the limits of current knowledge of encapsulins via two avenues of interrogation:

(1) A dataset of over 1000 novel encapsulin candidates is presented, discovered from metagenomics data using bioinformatics and deep learning tools. These novel encapsulins may display new structural features or biological functions. Three of these novel encapsulin candidates were recombinantly expressed in *E. coli* and one of the recombinant proteins was purified for biophysical characterisation.

(2) Deep learning tools for protein structure prediction and design were applied to encapsulin proteins, to design putative encapsulins with promising physicochemical properties. A computational pipeline was used to select 48 encapsulin candidates from thousands of initial designs. The resulting recombinant proteins were investigated experimentally using laboratory automation techniques. One chimeric variant of an encapsulin from *Quasibacillus thermotolerans* was characterised biophysically.

Overall, this work presents an account of the opportunities for the discovery or design of self-assembling protein particles and offers a synopsis of the challenges this task presents.

Table of Contents

List of Abbreviations

AI - Artificial intelligence

BGC: Biosynthetic gene cluster

CNN: Cargo loading peptide

CNN: Convolutional neural network

DL: Deep learning

DLS: Dynamic light scattering

DNA: Deoxyribonucleic acid

IPTG: Isopropyl β-1-thiogalactopyranoside

MCMC: Markov-chain Monte Carlo

MCP: Major capsid protein

ML: Machine learning

MSA: Machine sequence alignment

NLP: Natural language processing

pLDDT: Predicted local distance difference test

pLM: Protein language model

TM-Score: Template modelling Score

XAI: Explainable artificial intelligence

Chapter 1: Introduction

1.1 Organelles and Compartmentalisation in Biology

The physical separation of biological matter from the surrounding environment is a defining property of all known life on Earth. Within living cells, the presence of specialised compartments with defined functions is also universal and observed across all domains of life [1]. Separating different areas of the cell creates distinct chemical environments adapted to different functions, such as energy metabolism, storage and regulation of genetic material, or disposal of waste products. Separation of these processes in space also allows them to be regulated in a more complex manner. Such specialised compartments or structures are known as organelles, and can either be delineated by lipid membranes [2] or other biological macromolecules like proteins [3]. In this work "organelle" is defined as any subcellular compartment.

1.1.1 Compartmentalisation in Eukaryotes

Eukaryotic cells display a complex level of compartmentalisation compared to the other domains of life, with multiple highly organised membrane-bound organelles. The nucleus is a double membrane-bound organelle responsible for storing and regulating the genetic material of a cell [4]. The presence of a nucleus is often stated as the key difference between eukaryotes and simpler prokaryotes, which lack such a defined compartment for their genetic material. Continuous with the nuclear membrane is the Endoplasmic Reticulum (ER), a large network of membrane compartments where protein folding and post-translational modification take place [5].

In addition to a nucleus, eukaryotes are also the only known organisms to possess mitochondria and chloroplasts – membrane bound organelles with their own genome, which are responsible for energy metabolism and photosynthesis respectively [6]. Crucially, these organelles are referred to as "endosymbiotic" organelles and are thought to have originated from free-living prokaryotes that were likely internalised, domesticated and repurposed by an ancestral host cell [7]. Indeed, it is thought that the origin of eukaryotes directly corresponds with the emergence of the mitochondrion, since all known eukaryotes have mitochondria, either currently or at some point in evolutionary history [8].

1.1.2 Compartmentalisation in Prokaryotes

Traditionally, it was thought that only eukaryotes possess membrane-bound organelles [9], however recent evidence has firmly disproven this rule. Ammonia-oxidizing bacteria make use of membranous organelles known as annamoxosomes for energy metabolism [10], whilst photosynthetic prokaryotes use specialised membranes known as thylakoid membranes for light harvesting and electron transport [11]. Some bacteria can align themselves with the Earth's magnetic field lines using a membranous organelle called a magnetosome [12], while some nitrogen fixing bacteria display intricate, stack-like cytoplasmic membrane structures under nitrogen-fixing conditions [13]. These examples have thoroughly disproved the idea that membrane-bound organelles are unique to eukaryotes. Prokaryotes also make use of proteinaceous organelles to compartmentalise cellular processes. The enzyme lumazine synthase assembles into an icosahedral compartment in bacteria and archaea [14], whilst bacterial ferritins and bacterioferritins form spherical protein capsules used for cytoplasmic storage of iron [15].

1.2 **Encapsulins**

Encapsulins are a class of icosahedral protein-based organelle found across both bacterial and archaeal phyla. These protein compartments encapsulate so-called "cargo" proteins in a specific manner (Figure 1.2.1). Packaging is dependent on a "cargo loading peptide" (CLP) present in the specific cargo protein(s) of any given encapsulin compartment [16]. Encapsulins have many natural functions, including bulk storage of iron [17], oxidative stress resistance [18], and sulphur metabolism [19]. A large-scale computational search published in 2021 revealed over 6000 known encapsulin sequences with a multitude of different cargo types, including ferritin-like proteins for iron storage, dye-decolourizing peroxidases, cysteine desulfurases, hydrogenases, and xylose kinases [20].

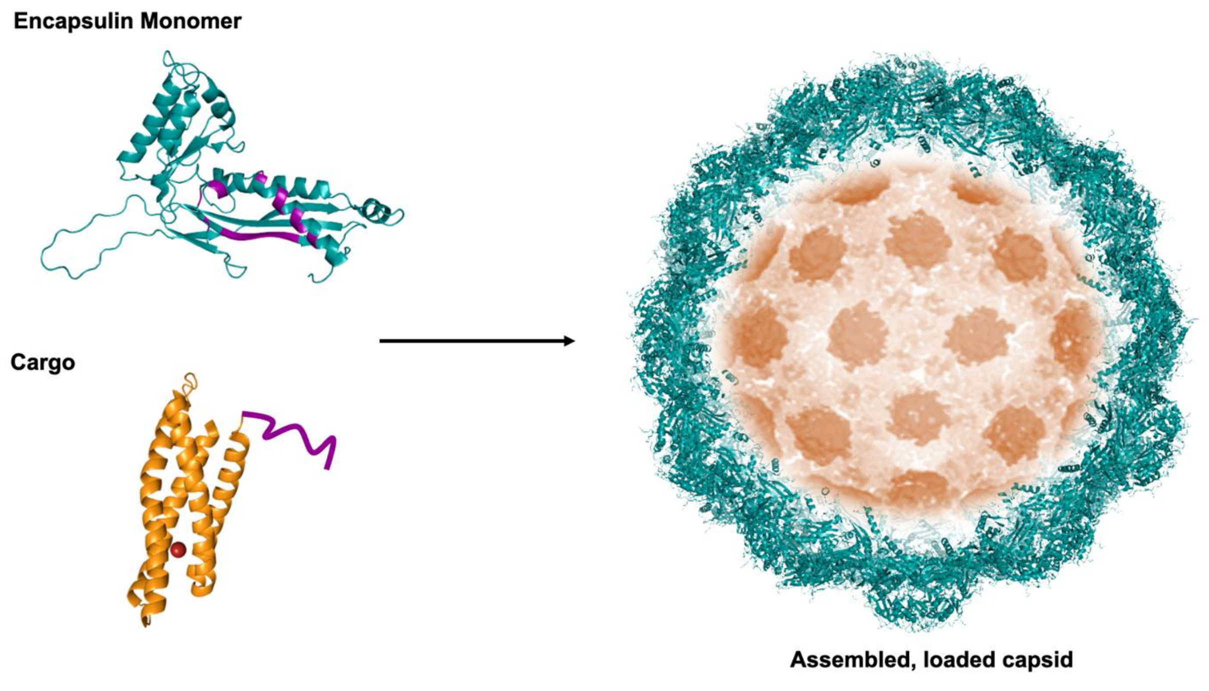


Figure 1.2.1: **Encapsulin Cargo Loading**
Encapsulins are icosahedral protein shells that package molecular cargo in a specific manner. Encapsulin proteins shown in cyan, with cargo proteins shown in orange. A C-terminal loading peptide present in the cargo protein recognizes a region in the encapsulin monomer (both shown in purple). Simultaneous expression of both proteins results in formation of assembled capsids loaded with functional cargo, in this case a mineralization protein for storage of iron. Encapsulins and cargo proteins shown are from *Quasibacillus thermotolerans* (PDB codes for encapsulin and cargo are 6NJ8 and 6N63 respectively). Loaded cargo density is not representative of actual atomic structure and is for schematic purposes only (adapted from ***[17]***). The cargo loading peptide was not modelled in the atomic resolution structure and is shown for diagrammatic purposes only.

That published work also categorised extant encapsulins into 4 different families based on function and operon layout. Family 1 contains "classical" encapsulins, whose cargo proteins are found upstream of the encapsulin coding sequence and targeted to the shell via a C-terminal CLP in almost all cases [1]. However, some family 1 systems have cargo proteins which are directly fused to the termini of the encapsulin protein. Cargo proteins encapsulated in these systems are typically dye-decolourizing (DyP) type peroxidases, or iron-binding proteins from the ferritin or rubrethyrin family, and are involved in oxidative stress response and iron storage respectively [21, 22].

Family 2 encapsulins display more diverse operon types, with some containing multiple encapsulin proteins, or fusions of the encapsulin monomer with a cyclic NMP-binding domain. These systems make use of a longer, disordered targeting domain in the cargo protein N-terminus for encapsulation [20]. Family 2 cargo proteins belong to one of four enzyme families – cysteine desulfurase, xylulose kinase, polyprenyl transferase, or terpene cyclases. Encapsulins containing cysteine desulfurase enzymes have been shown to sequester elemental sulphur [19, 23], and those that encapsulate terpene cyclase enzymes have been demonstrated to be involved in the synthesis pathways of natural products such as the soil odorant molecule 2-methylisoborneol [24]. The biological function of xylulose kinase and polyprenyl transferase encapsulins is currently unknown.

Family 3 encapsulins are known as "natural product encapsulins" due to their presence within large gene clusters associated with the synthesis of bioactive natural secondary metabolites (see Section 1.4) [20, 25]. Finally, family 4 contains "A-domain" encapsulins, so-called because of their truncated nature, containing only a single domain found in full-length encapsulins (see next section for more details). These encapsulins are mostly found in archaea and are around 30% of the length of a canonical encapsulin monomer [26].

1.2.1 Encapsulin Structure

Encapsulin compartments are all complexes of a single protein monomer. As seen in Figure 1.2.2, encapsulins share the same fold as the HK97 bacteriophage major capsid protein [27], suggesting that the two share a common ancestor. This fold is formed of three domains: axial and proximal domains ("A" and "P" domains) and an extended loop known as the "E" loop.

Encapsulin monomers assemble into an icosahedral, symmetrical compartment. As shown in Figure 1.2.1, encapsulin shells can form different icosahedral arrangements. An encapsulin shell can be assigned a triangulation number (T-number) based on shape and number of subunits. T-number classification was originally used to classify viruses based on the number of "structural units" that make up an icosahedral face [28], however it also applies to encapsulins which are themselves virus-derived icosahedral capsids.

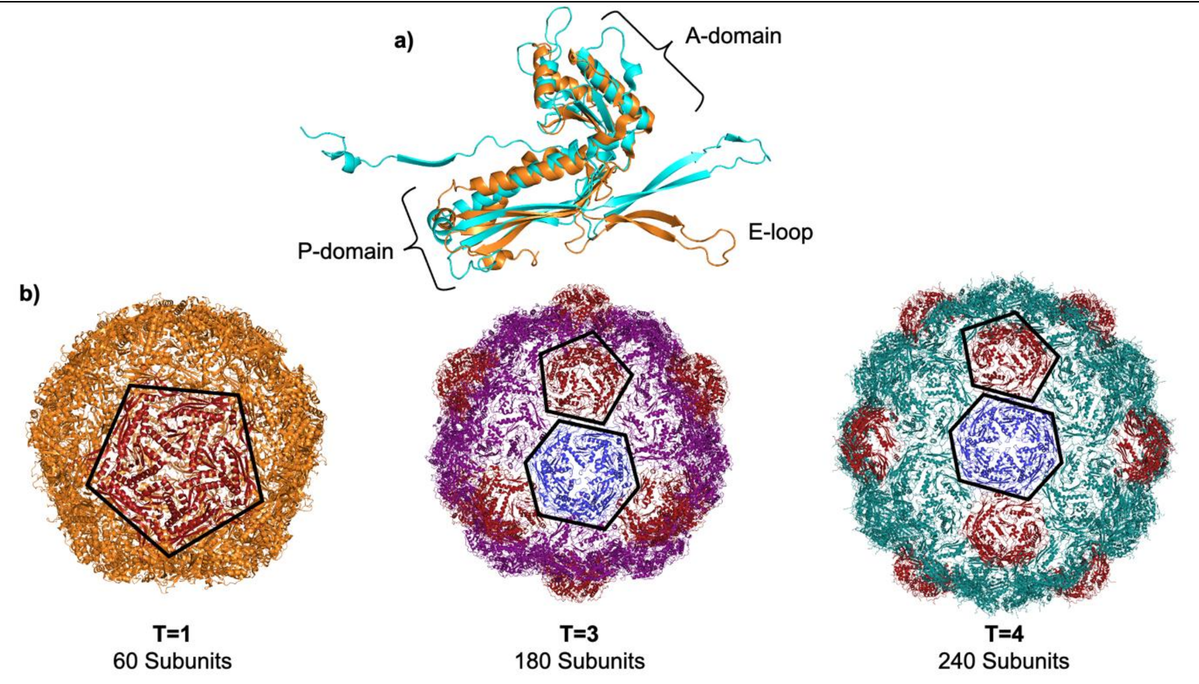


Figure 1.2.2: **Structure and Assembly of Encapsulin Microcompartments**
a) A structural alignment of *Thermotoga maritima* encapsulin (orange) with phage HK97 major capsid protein (cyan) shows that the two share a conserved fold and domain architecture. Proximal and axial (P- and A-domains) are connected by an extended loop (E-loop). b) Encapsulin compartments all form symmetrical icosahedral shells, with different triangulation numbers (T-numbers). T=1 capsids such as that of Thermotoga maritima (orange) are only formed of pentameric encapsulin assemblies (red). Higher T-number also shells use hexameric assemblies (blue), such as the T=3 encapsulin from *Myxococcus xanthus* (purple) or the T=4 encapsulin of *Q. thermotolerans* (cyan). PDB accessions: 7MU1 (*T. maritima*), 7S21 (*M. xanthus*), 6NJ8 (*Q. thermotolerans*), 1OHG (HK97).

The number of subunits in an encapsulin shell is equal to the T number multiplied by 60, and as such the surface area and volume of an encapsulin shell increases with T number. Crucially, T=1 encapsulins are only formed of pentamers, whereas T=3 and T=4 capsids contain pentamers and hexamers, despite still only using one protein monomer to form the shell. Capsids with a T-number higher than 1 are known as "pseudo-symmetrical" because their subunits are not all identical – the monomers that form hexamer and pentamer units are in different conformations. T number can also vary under different conditions - the *M. xanthus* encapsulin has always been considered a T=3 capsid, however recent crystal structures show that it can also form a T=1 capsid in the absence of any cargo proteins [22].

The precise sequence and structural elements that mediate capsid assembly and geometry are still largely unknown. Experimentally determined encapsulin structures typically show the most variation in the E-Loop region (Figure 1.2.3), suggesting that this region may be a determinant of capsid geometry. Indeed, the angle of the E-loop does appear to be related to T-number, with T=1 encapsulins generally showing a different conformation to higher T-number capsids. However, the *S. elongatus* encapsulin is an exception to this rule, showing a T=1 geometry but with an E-loop angle resembling higher T-number capsids.

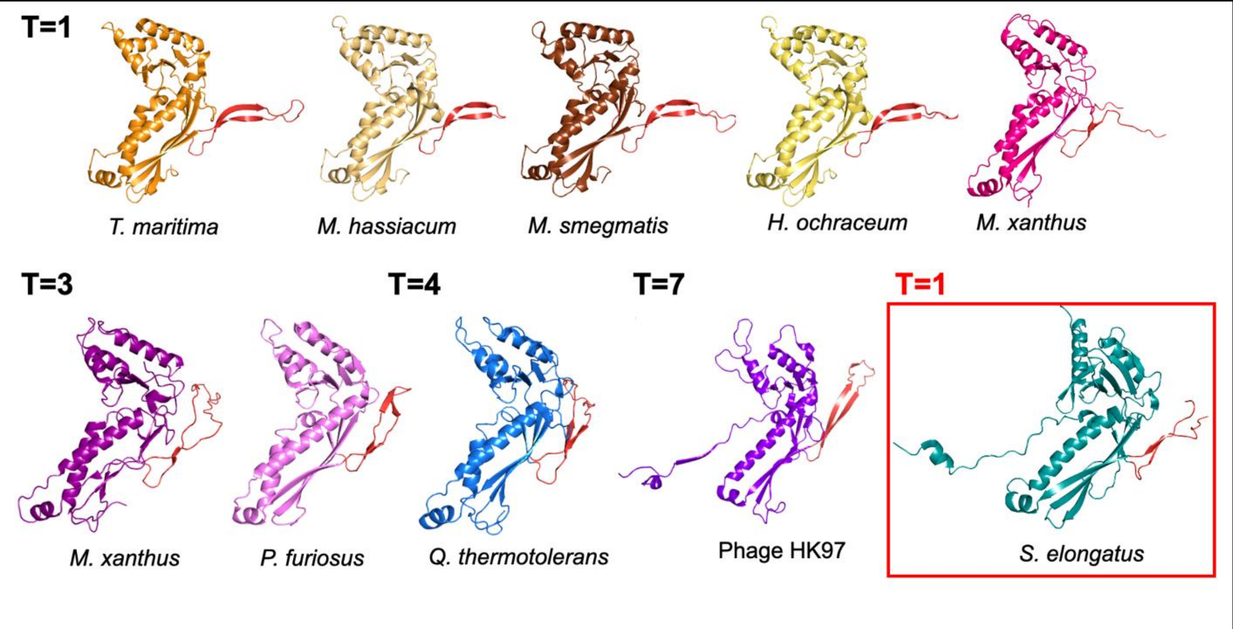


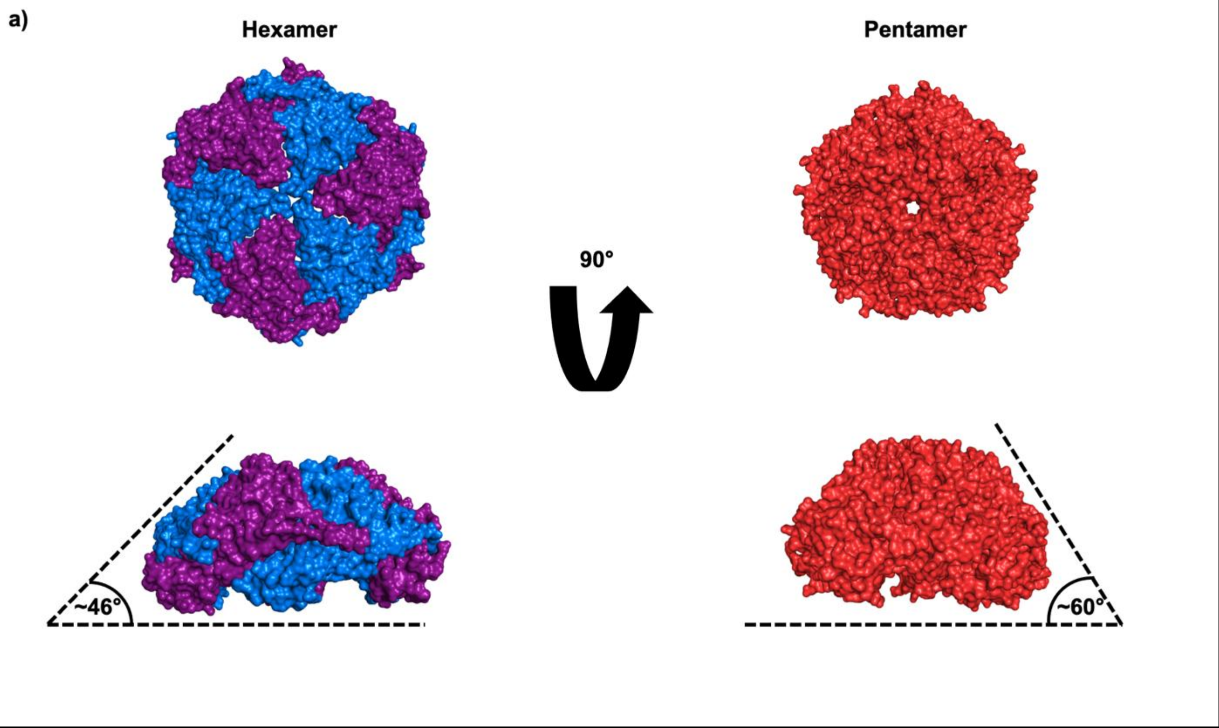
Figure 1.2.3: **Encapsulin Monomers with Different T-Numbers**
E-loop regions highlighted in red. In general, T=1 encapsulins show a characteristic conformation with the E-loop angled out away from the P and A domains. However, higher T number encapsulins (as well as the HK97 phage capsid protein) show a different conformation with the E-loop lying more parallel with the body of the encapsulin. However, the S. elongatus encapsulin exhibits T=1 geometry but with the E-loop parallel to the body as in T=3,4, and 7 proteins.

However, recent work has highlighted the importance of other structural regions in encapsulin assembly. In the *M. xanthus* T=3 encapsulin protein, mutations in a flexible loop at the tip of the A-domain result in so-called "symmetry reduction" of the capsid shell; instead of a major T=3 and a minor T=1 population of particles as seen in the wild-type, loop mutants now show a major T=1 and a new, minor population of particles with tetrahedral symmetry [29]. Another recent study on the *M. xanthus* encapsulin showed that insertion of a short peptide from the SARS-CoV2 spike protein into the encapsulin monomer can disrupt icosahedral assembly. In this case, the majority of assembled particles belong to several different classes of non-icosahedral cage, with only minority populations of T=1 and T=3 icosahedral shells observed [30]. Interestingly, this effect is seen when the peptide is inserted either in the E-Loop, or in a flexible loop in the P-domain spine helix.

Taken together, the findings from these two studies suggest that there are multiple symmetry- or assembly-determining regions in the *M. xanthus* encapsulin monomer.

T=1 capsids are formed only of pentameric units, whereas higher T-number shells contain both hexameric and pentameric units. As shown in Figure 1.2.4, the hexamer takes on a flatter shape whilst the pentamer displays a steep pyramid-like shape. In icosahedral capsids, the T-number corresponds to the number of non-equivalent monomers in the shell. For example, T=1 encapsulins only contain a single identical protein subunit, which assembles into pentameric complexes that form the icosahedral shell. However, T=3 encapsulins contain three different monomer conformations – one that forms the pentamer and two that form the hexameric unit [22]. Interestingly, as shown in Figure 1.2.4, there are only minor conformational differences between these three different non-equivalent subunits in the T=3 encapsulin shell.

T=1, 3, and 4 are the only native, icosahedral encapsulin geometries that have been observed so far (not counting the engineered non-icosahedral *M. xanthus* variants described previously). This is in stark contrast to phage capsid proteins, which share a similar fold but exhibit far greater diversity in T-number [31]. There are over 6000 known encapsulin sequences, but less than 20 of these sequences have been described at atomic resolution with experimental structural biology methods [17, 19, 22, 32–39]. Given this small sample size, there may be a wealth of unexplored structural and functional diversity present in natural encapsulins.



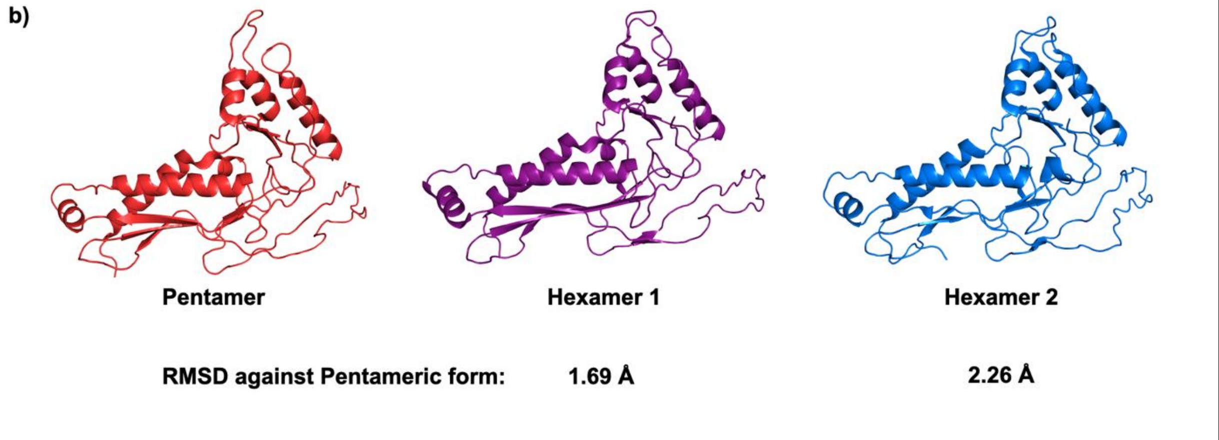


Figure 1.2.4: **Comparison of Pentamer and Hexamer Units in *M. xanthus* T=3 Encapsulin.**
a) Surface depiction of the hexameric (left) and pentameric (right) units that form the *M. xanthus* encapsulin shell (PDB **7S2T**), coloured by non-equivalent subunits – red for pentamer, and blue/purple for the two different monomers that make up the hexamer. As shown, the hexamer takes on a flatter shape compared to the pentamer, whose outer surfaces make a shallower angle against the base of the structure. **b)** Comparison of the three different non-equivalent units. There are only minor conformational differences between the three, mainly found in the β sheet region of the P-domain. This region is formed of broken β strands in the pentamer, but longer strands in the "hexamer 1" monomer, and shorter strands in the "hexamer 2" form. Note that this structure was solved by cryoEM at 3.45 Å resolution.

1.2.2 Encapsulin Evolution

Encapsulins show structural and sequence homology with HK97-fold bacteriophage major capsid proteins (hereafter referred to as MCPs), implying that the two families share a common ancestor [40]. Phylogenetic and structural comparisons show that encapsulins form a monophyletic clade, whereas MCPs are more diverse [33]. These comparisons also showed that encapsulin proteins are more closely related to each other than to MCPs (ignoring the truncated family 4 sequences). Bacterial and archaeal encapsulins are more closely related to each other than to MCPs, indicating ancestral horizontal gene transfer between these two kingdoms [26]. This might indicate that encapsulins appeared in bacteria before archaea, since horizontal gene transfer occurs more often from bacteria to archaea than vice versa [41]. However, this is simply speculation and has not been confirmed in the literature.

It is currently unclear whether encapsulins or MCPs diverged first from their shared ancestor, and thus whether the HK97 fold originated in a cellular host or a virus. Andreas and Giessen [26] argue that encapsulins evolved via bacterial domestication of an ancestral phage MCP. Indeed, MCPs are primarily found in tailed dsDNA phages of the order *Caudovirales*, which infect a broader range of prokaryotic hosts than those containing encapsulin proteins [40]. This might suggest a broad viral origin of the HK97 fold before divergence of encapsulin function in a narrower set of prokaryotes. However, there is also some evidence to suggest that the HK97 fold may have originated in encapsulins before being co-opted by viruses. Encapsulins are found in some archaeal phyla (such as *Crenarchaeota)* which are not known to be colonised by *Caudovirales* phages [42]. From a structural perspective, encapsulin shells also appear more primitive than their complex phage capsid counterparts. All known experimental encapsulin structures are either T=1, T=3, or T=4, whereas phage capsids can reach T-numbers up to T= 52 [43]. Phage capsids also have a more complex assembly pathway then encapsulins, involving scaffolding proteins or insertion domains, proteases and chemical crosslinking, and multiple intermediate structures before the mature shell is formed [44].

This gap in complexity is best illustrated with an example. As shown in Figure 1.2.5, the bacteriophage HK97 T=7 MCP shows similar structural features with the *Synechococcus elongatus* encapsulin. The two share a similar E-loop conformation and have their N-termini facing the outside of the shell, in contrast with all other encapsulin structures. However, the encapsulin only forms a T=1 shell and self-assembles with no scaffolding proteins, where the phage HK97 MCP forms a larger T=7 capsid and requires a scaffolding domain and a protease to form a mature capsid [45]. Despite their structural similarities, the gap in assembly complexity implies that phage MCPs might have evolved from and expanded upon their more primitive encapsulin counterparts.

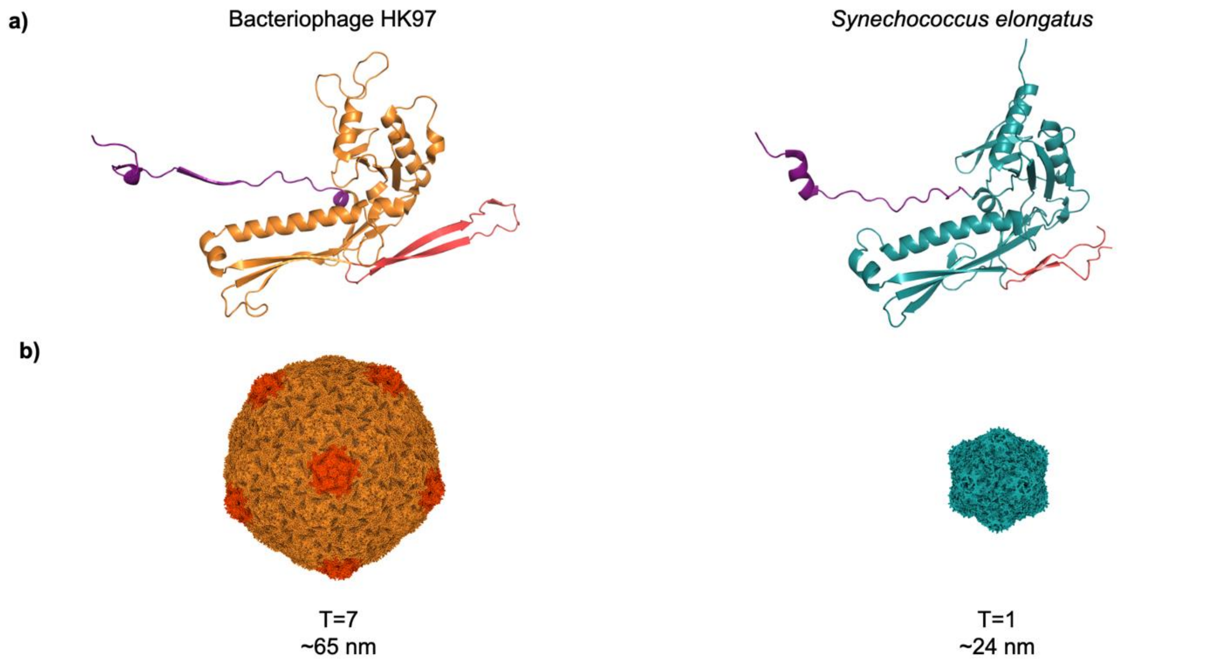


Figure 1.2.5: **Structural Complexity of HK97-type Phage Capsids and Encapsulins**
a) Monomer structures of the Bacteriophage HK97 MCP (orange) and *Synechococcus elongatus* (cyan). As well as sharing the same overall fold, the two structures also share a similar E-loop angle (highlighted in red) and both expose the N-terminal arm (shown in purple) to the outside of the shell, which is uniquely seen in this encapsulin structure. **b)** Comparison between fully assembled shell geometries, with HK97 pentamers highlighted in red for contrast. Scales depicted here are only approximate. Despite sharing structural features in the monomer, the HK97 MCP assembles into a large T=7 capsid, whereas the *S. elongatus* encapsulin forms a much smaller T=1 shell. PDB accessions **1OHG** (HK97) and **6X8M** (*S. elongatus*).

These two pieces of evidence point to a cellular, possibly prokaryotic origin of encapsulins before the divergence and increase in complexity of HK97-type MCPs. This aligns well with a wider evolutionary theory that points to a cellular origin of many viral proteins, which may potentially include MCPs [26, 42, 46]. However, it must be stressed that currently, the overall evolutionary history of encapsulins and MCPs is poorly understood, and there is no widely accepted theory as to which came first.

1.3 Biotechnology Applications of Encapsulins

Encapsulins have many potential applications in synthetic biology, biotechnology, and biomedicine. Engineering these compartments to allow packaging and delivery of DNA or RNA may be a route towards novel gene therapies and vaccines [47]. The encapsulin surface may also serve as a functional site for antigen display in protein-based vaccines [48]. Indeed, encapsulin nanoparticles displaying the SARS-CoV-2 spike protein have recently been shown to induce immunogenic responses in animal models [49]. DNA loading into encapsulins may also serve as a new type of genotype-phenotype linkage for use in directed evolution experiments. Directed evolution has previously been used to repurpose a bacterial enzyme into an RNA capsule [50]. In this work, the packaging ability of an RNA capsule library was selected against by testing their ability to protect their own mRNA molecules from nuclease degradation. A similar approach could be used for encapsulins, selecting for function by subjecting mutants to nuclease treatment.

Heterologous loading of non-natural protein cargos has been previously demonstrated with encapsulins [34] and could have interesting applications for metabolic engineering, for example in the isolation of difficult enzymatic reactions producing toxic intermediates in microbial cell factories. Encapsulins are highly tolerant to extremes in temperature and pH, and can be reversibly disassembled and reassembled with cargo loading *in vitro* [51] – these properties could be exploited for drug or vaccine delivery, as has already been shown in proof-of-concept studies [52]. Encapsulins have also been fused with metal-binding domains for use as imaging agents in correlative electron microscopy [53].

1.4 Biosynthetic Gene Clusters and Secondary Metabolites

Biosynthetic gene clusters (BGCs) are groups of genes found in close proximity, which encode a biosynthetic pathway whose product is a specialised metabolite [54]. These specialised metabolites are described as "secondary metabolites" or "natural products" - small molecules which are not essential for an organism's survival, but which convey an evolutionary benefit in a competitive environment [55]. Secondary metabolites are ubiquitous and produced by prokaryotes, fungi and other eukaryotic microbes, and plants, however this work will focus specifically on microbial BGCs, mainly in prokaryotes, and their secondary metabolite products.

Antibiotics and antimicrobials are perhaps the most well-known class of secondary metabolite produced by BGCs, however these gene clusters produce molecules that are implicated in a wide range of biological processes, including symbiosis, host-microbe interactions, and immunity [56]. The biotechnology and pharmaceutical industries have historically shown great interest in secondary metabolites, for use as antibiotics and therapeutics, pesticides, dyes, flavourings, and cosmetics [57–59]. Traditionally, these molecules were discovered in microbiological experiments. For example, penicillin (perhaps the most important secondary product of the 20th century) was serendipitously discovered when *Penicillium* mould was observed to inhibit the growth of *Staphylococcus* bacteria on agar plates [60]. Similarly, the cephalosporin class of antibiotics was first discovered when the growth of *Salmonella typhi* was seen to be inhibited by *Cephalosporium* fungi on agar plates [61].

Secondary metabolites are frequently discovered using so-called bioactivity assays, where model organisms are exposed to crude extracts of plants or microbes and screened for a desired phenotype [62]. For example, the prolific anticancer drug paclitaxel was discovered when cancer cell lines were systematically exposed to extracts of the bark of the *Taxus brevifolia* tree [63]. The psychedelic drug lysergic acid diethylamide (LSD) was discovered in an unconventional variation on this technique, when the synthetic chemist Albert Hofmann was accidentally exposed to a derivative of an alkaloid extracted from ergot fungal species grown on rye [64].

The mid-20th century was a productive period in natural product discovery, with the 1950s and 1960s sometimes referred to as the "golden age" of antibiotics, when most antibiotics in use today were first discovered [65]. Following this period, many major pharmaceutical companies set up extensive natural product discovery programs, targeting infectious diseases and cancer. However, by the 1990s and early 2000s, many of these companies had closed their discovery programs, believing that the results of such discovery campaigns mostly recapitulated known compounds, and that any new compounds discovered were too difficult to synthesize *de* novo [66]. Despite this downturn in commercial interest, the early 2000s also heralded a new era for natural product discovery in the genomic age. In particular, full genome sequences for bacteria of the genus *Streptomyces* revealed a plethora of potential BGCs and their associated natural products [67]. This new "genome mining" approach promised a much more systematic way of discovering natural products by examining clusters of genes involved in their biosynthesis.

Indeed, the fields of BGC and secondary metabolite discovery have only continued to grow since the early 2000s, for several reasons. The advent of faster, cheaper genome sequencing technology has resulted in the near-exponential growth of genomic databases [68], and thus more raw data from which to discover new BGCs and products. Of particular interest in this study is the growth in metagenomics databases (see Section 1.5.3) which may represent a particularly rich and diverse source of new BGCs. In line with the increasing amounts of data available, computer hardware has become exponentially faster and cheaper over recent decades [69], and new bioinformatics tools for discovering BGCs and their products have been released [70]. In particular, the proliferation of deep learning (see Section 1.5.2) has produced advances in tools not only for general protein bioinformatics, but also for BGC prediction [71]. Finally, BGCs are becoming easier to study experimentally due to the falling costs of DNA synthesis, and the development of modern DNA assembly protocols [72], high throughput screening and automation techniques [73], and cell-free protein synthesis [74].

1.4.1 Encapsulins and Biosynthetic Gene Clusters

As mentioned previously, Family 3 encapsulin proteins are associated with predicted BGCs [26]. Whilst none of these BGCs have been experimentally investigated, they are predicted to encode a diverse range of putative enzymes. Several enzyme families commonly appear in these putative gene clusters, chief among which are short-chain dehydrogenase/reductase (SDR) and sulfotransferase enzymes. Sulfotransferases are known to catalyse the conjugation of sulphate groups to a wide range of secondary metabolites [75], and as such the co-occurrence of these two enzyme families suggests that these BGCs may produce sulphur-containing compounds. Some of these putative encapsulin BGCs contain amino group carrier proteins (AmCPs), which are involved in the biosynthesis of lysine and arginine in bacteria, but can also be found in *Streptomyces* BGCs producing non-proteinogenic amino acid secondary metabolites [76]. Other encapsulin-associated putative BGCs contain polyketide synthases (PKS), which are large multifunctional enzymes or enzyme complexes involved in the synthesis of complex natural products containing long polyketide acyl chains [77]. Lastly, a subset of these putative encapsulin BGCs are found to contain non-ribosomal peptide synthetases (NRPS), another group of large, modular enzyme complexes responsible for producing modified peptides in a stepwise "assembly line"-like fashion [78].

This summary of encapsulin-associated BGC function is deliberately broad, since none of these gene clusters have been experimentally investigated in any detail, and as such there is almost no evidence of function or product formation for these BGCs. Only one single encapsulin-containing BGC (from *Myxococcus* sp. MCy10608) has been studied experimentally [79]. Functional annotations show this BGC to contain an NRPS enzyme, and mass spectrometry revealed its products to be a mixture of chlorinated peptide products known as chloromyxamides. Whilst this study did describe the structure of the final BGC product, no experiments on its biosynthetic pathway could be performed, owing to the difficulty of genetic manipulation in the *Myxococcus* host. Isotope labelling studies showed that the chloromyxamide products are derived from glutamic acid, but more precise evidence of enzymes in the biosynthetic pathway and the reactions they catalyse is yet to be presented.

The exact role of encapsulin proteins in these BGCs is unclear, however Andreas and Giessen [26] speculate that these BGCs contain specifically encapsulated cargos, due to the presence of large disordered regions in these proteins. It is suggested that encapsulation may aid enzyme catalysis, through the sequestration of reactive pathway intermediates. This concept of sequestering unstable or volatile intermediates is employed by bacterial microcompartments (BMCs), another type of icosahedral proteinaceous organelle [80]. This hypothesis of specific encapsulation of BGC enzymes is plausible, but currently lacks supporting experimental evidence. On the other hand, it is entirely possible that BGC-associated encapsulins do not encapsulate enzymes at all. Andreas and Giessen also describe unique Family 3 encapsulins with a putative C-terminal transmembrane domain [26]. This could implicate encapsulins in transport or membrane localization roles in these BGCs. In summary, encapsulin-associated BGCs are currently vastly understudied, and there is much scope for wet lab characterisation and investigation of these gene clusters and their products.

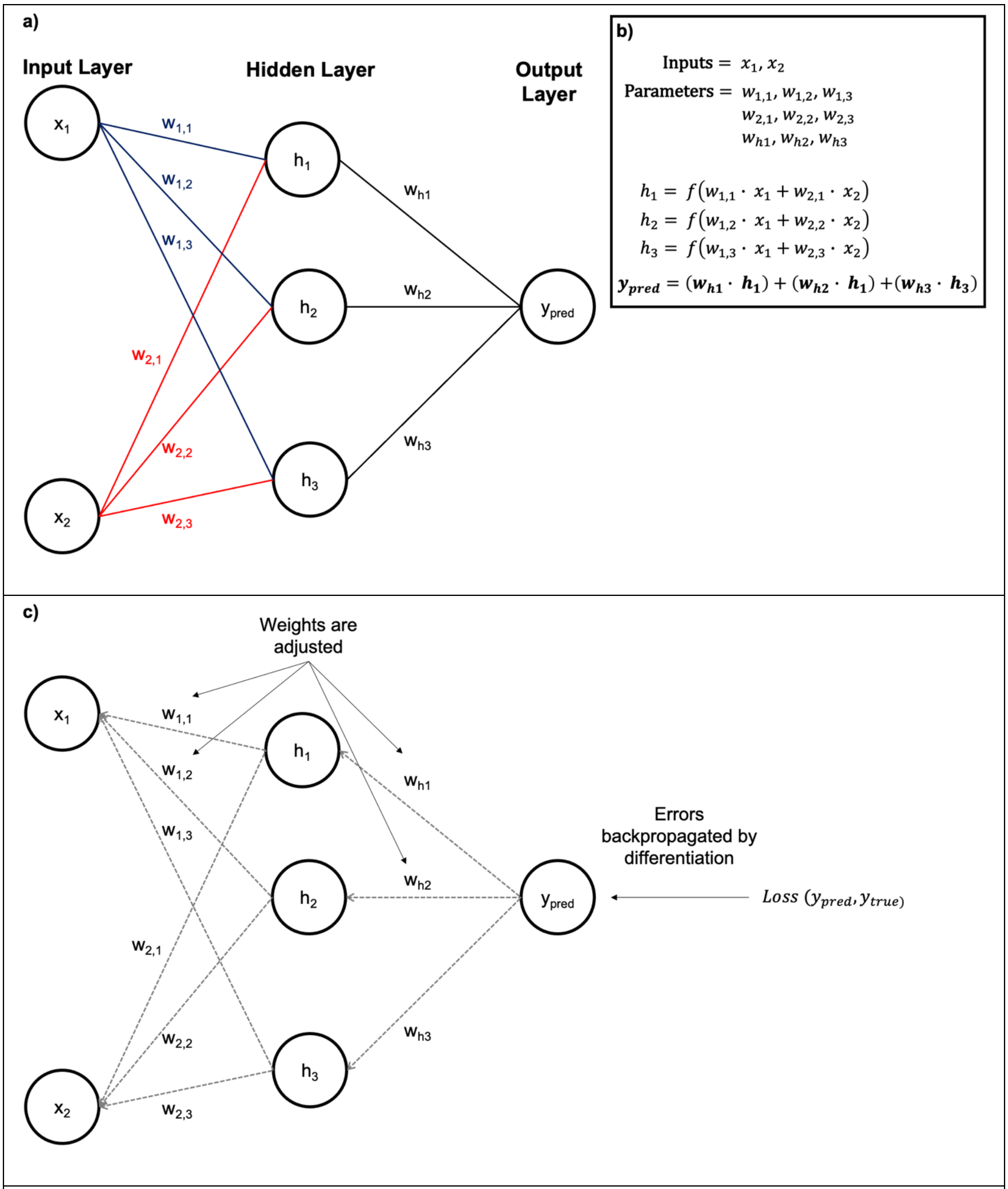
1.5 **Computational Techniques**

1.5.1 Machine Learning

Machine Learning (ML) is a broad area of computer science whose primary focus is on recognizing patterns in large, complex datasets [81]. Traditionally, a programmer will solve a problem by manually devising the rules and logic which govern the solution space for that problem [82]. In contrast, machine learning aims to forego manual rules, and instead learn them directly from the dataset. This process is called training, whereby a mathematical model is presented with many input examples which are used to optimise its parameters and infer patterns from the data [83]. The term "machine learning" is used interchangeably with Artificial Intelligence (AI) although the consensus in the field is that ML is a subfield of AI research [82].

The field has its origins in the 1950s, starting with Alan Turing's seminal paper *Computing Machinery and Intelligence* where he pondered the question: "Can machines think?" [84]. In this paper Turing formulated his famous "Turing test" whereby a machine can pass the test if it can convince an observer that it is human. The actual term "machine learning" is widely believed to have been coined in 1959 by Arthur Samuel, a researcher at IBM who stated that "a computer can be programmed so that it will learn to play a better game of checkers than can be played by the person who wrote the program" [85]. A year earlier, Frank Rosenblatt published a paper detailing the "Perceptron" – an analogue computer designed to detect visual stimuli modelled after the human brain [86].

The Perceptron was the earliest example of what is referred to today as a neural network – and indeed the term "multi-layer perceptron" is still regularly used to describe modern neural networks [87]. A neural network can be thought of as a nested mathematical function, with fully connected layers of nodes or "neurons" (Figure 1.5.1). Signals propagate through the network via weighted connections and activations [88]. During training, the weights of all the neurons in the network are optimised to minimise a cost function over the training data, using a process known as backpropagation [89].



b)

$$\text{Inputs} = x_1, x_2$$
$$\text{Parameters} = w_{1,1}, w_{1,2}, w_{1,3}$$
$$w_{2,1}, w_{2,2}, w_{2,3}$$
$$w_{h1}, w_{h2}, w_{h3}$$

$$h_1 = f(w_{1,1} \cdot x_1 + w_{2,1} \cdot x_2)$$
$$h_2 = f(w_{1,2} \cdot x_1 + w_{2,2} \cdot x_2)$$
$$h_3 = f(w_{1,3} \cdot x_1 + w_{2,3} \cdot x_2)$$
$$\boldsymbol{y_{pred}} = (\boldsymbol{w_{h1}} \cdot \boldsymbol{h_1}) + (\boldsymbol{w_{h2}} \cdot \boldsymbol{h_1}) + (\boldsymbol{w_{h3}} \cdot \boldsymbol{h_3})$$

Figure 1.5.1: **Artificial Neural Networks**
a) A neural network is made up of an input layer, one or more hidden layers, and an output layer. This toy network takes two inputs, has one hidden layer with three nodes, and gives a single output. Layers are fully connected, and connections have weights which are randomly initialised. **b)** Here there are two input values, and 9 weights, which are parameters of the model. The value of any given node in the network is calculated as the weighted sum of all inputs it receives, transformed by a nonlinear "activation function" *f.* **c)** In training, the difference between prediction and true values for given inputs is calculated using a loss function. A process known as backpropagation is used to adjust the weights such that the loss function is minimised over all training data.

Work on the theory and applications of these primitive neural networks continued throughout the 1960s, however the field saw major funding cuts towards the end of the decade, heralding the approach of the first so-called “AI winter” [90]. This slowing of progress has been attributed to several factors. It has been suggested that early AI researchers severely underestimated the difficulty of the problems they were trying to solve (self-driving vehicles, translation and speech recognition, and similar) [90]. Early ML models were aimed at solving scaled-down “toy problems”, and it was widely believed that once these were solved, working up to more difficult real-world problems would be trivial. However, discoveries in computational complexity theory soon revealed the true difficulty of these problems. Around this time, severe limitations were also discovered in the underlying ML models being studied – it was demonstrated that simple neural networks such as the Perceptron are only capable of approximating linear functions, and as such these simple models are not suitable for even the simplest of non-linear problems (which are overwhelmingly common) [91].

Following the first AI winter of the late 1960s and 1970s, there was a revival of interest in AI and ML during the 1980s, with a narrower focus on so-called “expert systems”, where ML models were developed to solve domain-specific problems [90]. This is in contrast with the more general work of the 1960s, where broad methods were developed and applied to many different problems. Many fundamental discoveries in ML were made in the 1980s which are still widely used today, including the training of neural networks with backpropagation [89] and the first convolutional neural network [92]. However, the true power of these new tools was only fully realised from the late 1990s onwards, as computers became faster, storage became cheaper, and lots more data became available to train models on [90].

An important distinction within ML is supervised versus unsupervised learning. In supervised learning, a model is trained on data that has known labels – for example, training a model to classify images of cats and dogs typically requires a large training set of known images of cats and dogs [93]. In contrast, unsupervised learning is employed where the data has no known labels or “ground truth” values – in this case the goal is not to correctly predict an output value for a given input, but instead to uncover patterns within data and/or cluster similar examples together. This approach has been widely used in recommendation systems on the web and on social media platforms [94].

1.5.2 Deep Learning

Deep Learning (DL) is a subfield of ML whose focus is learning meaningful representations from very complex datasets, using a layered approach as shown in Figure 1.5.2. DL models typically have many parameters, in the order of millions or even billions [40]. A big recent breakthrough in deep learning was the discovery of the Transformer architecture (Figure 1.5.3a), a specialised type of neural network which uses a mechanism known as "self-attention" to assign weights to different parts of the input and learn long-range interactions in different data types [95]. Transformers have since achieved state-of-the-art performance in many different tasks, including image recognition [96] and language modelling and text generation [97]. DL models can also be applied to the problem of generating new synthetic data, such as life-like photographs of faces [98] or newspaper articles that read exactly like natural text written by a human [99]. Large language models have received unprecedented mainstream interest with the release of ChatGPT, a deep learning model from OpenAI based on the Transformer, which achieves remarkable

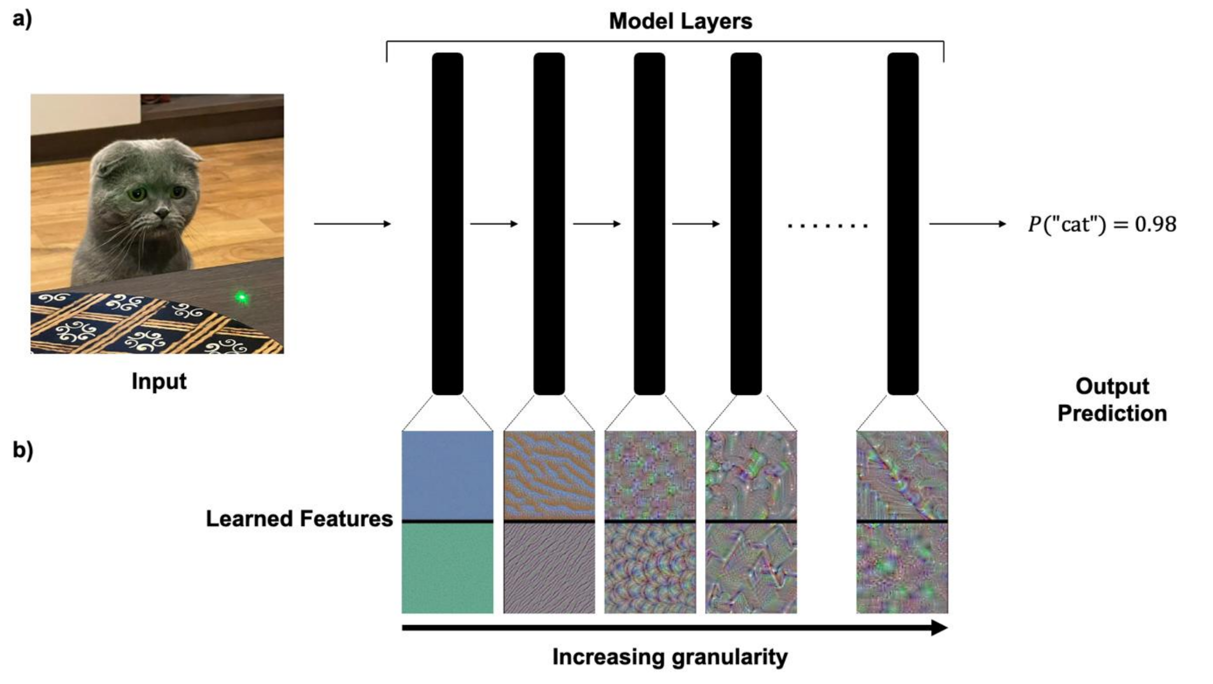


Figure 1.5.2: **Deep Learning Models**
Deep Learning models are neural networks made up of many layers – this is known as the "depth" of a model. This toy example shows a neural network which takes an input image, passes it through the layers of the model, and outputs the probability of the image containing a cat. Earlier layers learn more coarse-grained features of the input data, and the deeper layers gradually learn increasingly fine details and features [93]. Here earlier layers learn simpler features like blocks of colour and edges, while later layers interpret complex patterns of edges and colours.

performance in question answering and text generation in a conversational, human-like style [100].

The field of generative DL (models that produce new text, images, etcetera) has vastly benefitted from the discovery of the Transformer, as well as a more recent innovation known as the diffusion model (Figure 1.5.3b). Diffusion models are generative DL models that are trained on input data (usually images) with different amounts of random noise added. Given a set of noisy images and their noise-free counterparts, the model is trained to denoise the images, but can then be used to generate new images from random noise [101]. The input noise can be parameterised and modified to produce controllable output images. These powerful diffusion models have been combined with large language models to produce incredibly performant generative DL models such as Stable Diffusion [102] and DALLE-2 [103]. These models can produce lifelike images and convincing artwork given a text-based prompt as input, and have sparked fierce debate around the ethics of AI-generated artwork [104].

DL models are near-ubiquitous in the technology industry, due to their ability to learn useful patterns and representations from complex data which are otherwise intractable by simple computational tools. These models are used in smartphone cameras for image processing [105], language processing for voice assistants such as Apple's *Siri* [106], object detection in surveillance cameras [107], automatic spam detection and filtering in email [108], and improving online search engine results [109], amongst countless other examples.

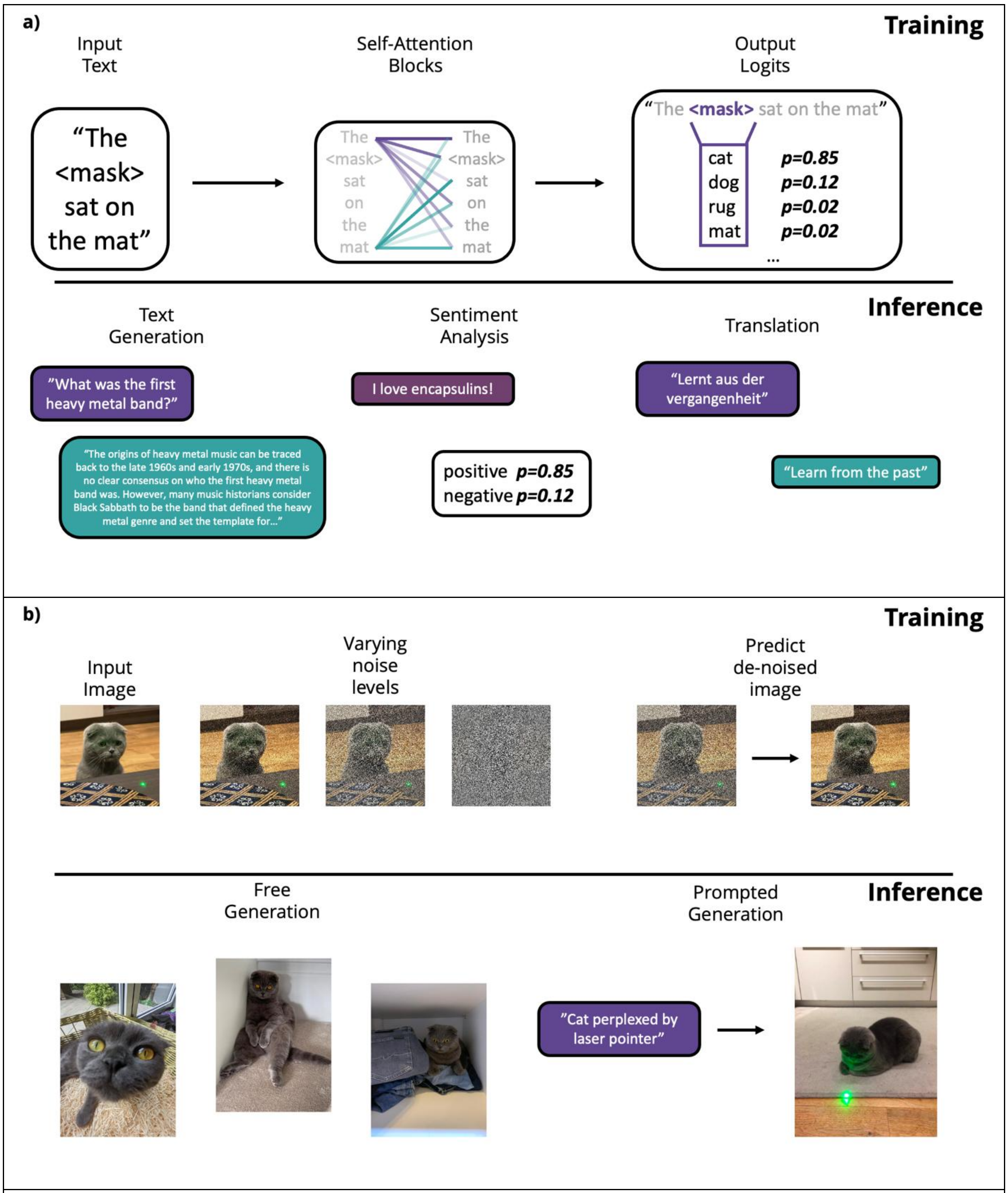

a)
Training
Input Text
Self-Attention Blocks
Output Logits
"The <mask> sat on the mat"
The <mask> sat on the mat
The <mask> sat on the mat
"The <mask> sat on the mat"
cat p=0.85
dog p=0.12
rug p=0.02
mat p=0.02
...
Inference
Text Generation
Sentiment Analysis
Translation
"What was the first heavy metal band?"
"The origins of heavy metal music can be traced back to the late 1960s and early 1970s, and there is no clear consensus on who the first heavy metal band was. However, many music historians consider Black Sabbath to be the band that defined the heavy metal genre and set the template for..."
I love encapsulins!
positive p=0.85
negative p=0.12
"Lernt aus der vergangenheit"
"Learn from the past"
b)
Training
Input Image
Varying noise levels
Predict de-noised image
Inference
Free Generation
Prompted Generation
"Cat perplexed by laser pointer"

***Figure 1.5.3:* Generative Deep Learning Models**
a) Transformer models use self-attention to assign weights to different parts of the input text and decipher long-range dependencies. They are trained to predict words that are removed ("masked") from the input. After pre-training they can be used to generate text in response to prompts or questions, classify sentences and analyse sentiments, and translate between different languages. **b)** Diffusion models are trained on images with varying amounts of gaussian noise added. In training the model predicts a denoised image from a noisier one. At inference time these models can be used to generate images from noise "freely" or can be combined with language models to generate images when given an input prompt in text.

1.5.2.1 Limitations of Deep Learning Models

Whilst DL models have transformed science and technology, they are not without their drawbacks. Since these models use deep neural networks with many millions or billions of parameters, they require huge amounts of training data accordingly. Large language models are often trained on hundreds of gigabytes of text [110], requiring substantial compute resources and incurring significant monetary and energy cost [111]. Indeed, it is estimated that the entire process of developing and training a large language model produces more CO_2 emissions than the average American citizen produces in two years [112]. The vast financial resources and infrastructure needed to train state-of-the-art models has led to a loss of transparency and democracy in the field, as only a handful of companies can afford to produce them [113].

DL approaches also suffer from a lack of interpretability. In contrast with simple mathematical models such as linear regression, DL models are often treated as "black boxes" where it is difficult to understand how or why the model has arrived at a particular output for any given input [114]. This may cause unpredictable faults in safety-critical scenarios where the decisions that a DL model makes could impact human lives, such as self-driving vehicles [115] or biomedicine [116–118]. As such, explainable AI (XAI) and interpretability of DL models is a rapidly growing field of inquiry [119]. Furthermore, DL models are fundamentally limited by the quality and biases of the data they are trained on. Larger models demand more and more training data, which is more difficult to filter and curate. This is particularly noticeable in generative DL models which are frequently shown to display racial and political biases [120, 121].

Finally, in the wider context of AI research, there is considerable scepticism surrounding DL models and their evaluation. DL models are typically trained on large datasets and evaluated on task-specific benchmarks; for example a language model may be tested on question-answering, logical reasoning, or translation [122], whilst image generation models are evaluated on object segmentation or "recognition in context" benchmarks [123]. This focus on strictly defined tasks and datasets can sometimes lead researchers to chase incremental improvements in benchmark scores, at substantial cost in terms of money, time, and carbon emissions, in exchange for negligible improvements in "real world" performance. This is an example of Goodhart's Law, where a metric "ceases to be a good measure” when it is used as a target [124]. To this end, there is some uncertainty among AI experts as to whether endless scaling of DL models will lead to an artificial general intelligence [125, 126].

1.5.2.2 Deep Learning in Protein Science

Traditionally, bioinformaticians have used empirically derived statistical models to understand protein sequence data. For example, simple evolutionary models like the Jones-Taylor-Thornton model are derived from natural sequence data and used in phylogenetic tree inference [127]. Protein functional annotation is often done by comparing a sequence with homologs of known function [128]. Protein structure prediction classically relied on homologs with known structure [129], and proteins are classified into families using multiple sequence alignments (MSAs) calculated using pre-determined scoring matrices [130] or Hidden Markov Models (HMMs) [131]. However, DL models are incredibly well suited to bioinformatics problems – there is an abundance of sequence data available in databases such as UniProt [132], as well as structure data in the Protein Data Bank [133]. As such, the application of DL models to protein data has been a rapidly growing field of research in recent years.

The release of the AlphaFold2 protein structure prediction model was arguably the greatest recent advance in this field (Figure 1.5.4, top), and indeed one of the biggest breakthroughs in protein science and bioinformatics in recent decades. This model achieves state-of-the-art accuracy in protein structure prediction from sequence data, by incorporating many cutting-edge DL innovations such as the Transformer architecture, self-attention, and learning from several different data modalities (sequences, structures, and multiple sequence alignments) [134]. This model represents a huge paradigm shift for structural biology. While this work was being completed, AlphaFold3 was released, building on the success of its predecessor and now allowing the prediction of protein/nucleic acid and protein/small molecule complexes [135]. However, the source code and model weights are not yet available for AlphaFold3 at the time of this work, and the model is only available as a limited-access web server, so only AlphaFold2 is discussed and used here.

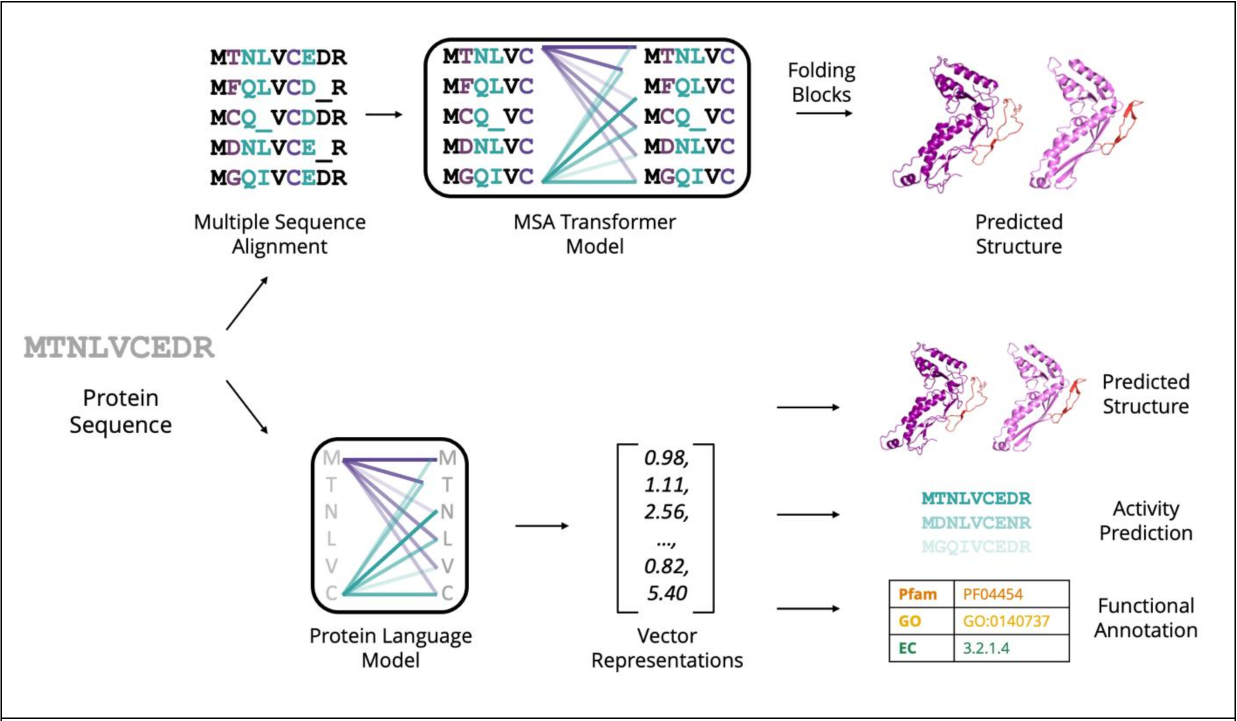


Figure 1.5.4: **Deep Learning Models in Protein Science**
Deepmind's AlphaFold2 (top) uses a Transformer model to learn evolutionary information from MSAs, which can be used to predict protein structures from sequences with near-experimental accuracy ***[134]***. Protein language models such as ESM from Meta AI (bottom) use Transformers on sequences only, to generate meaningful vector representations of protein sequences ***[136]***. These can then be used as inputs to "head" models to predict protein structure ***[137]***, mutational activity, and functional annotations ***[138]***.

Large Transformer models have also been trained on protein sequence databases in a manner analogous to the large language models used in Natural Language Processing (NLP) [136, 139]. These models have been shown to learn meaningful representations of protein sequences (Figure 1.5.4, bottom) which can be used for residue-residue contact prediction [140, 141] as well as prediction of mutational effects on protein function [142]. The state-of-the-art in large protein language models shows that these models follow similar scaling laws to human language models, and that learned representations continue to improve with more training data, as models reach billions of parameters [137]. This culminates in ESM-2, a 15 billion parameter model whose representations capture rich structural and evolutionary information about proteins from their sequence alone. These representations contain enough information to be used as input to a protein folding model, ESMFold. ESMFold can attain near-atomic accuracy in some cases, and whilst not as accurate overall as AlphaFold2, it has the crucial advantage of predicting structure from a single sequence input only. This avoids the need to generate MSAs, making prediction much faster and more accurate for proteins with few or no known homologs, such as those observed in metagenomics databases.

Similarly to AlphaFold3, an updated ESM3 protein language model was released as this work was finalised [143]. This new model is trained simultaneously on protein sequence, structure, and function information and promises vast improvements in performance. The number of model parameters and the amount of training data were both scaled by orders of magnitude compared to the largest ESM-2 model, and synthetically generated protein sequences and structure predictions were used in training. However, as with AlphaFold3, the source code and model weights were not released for the largest ESM3 model, which is only available via a limited-access API. As such, and as with AlphaFold3, only ESM-2 is discussed and used as part of this work.

Breakthroughs in DL models for protein science have revolutionised bioinformatics by vastly increasing the amount of publicly available structure data. The European Bioinformatics Institute provide a publicly available database of AlphaFold2 predicted structures covering the UniProt KB sequence database as well as selected model organism proteomes [144]. Similarly, Meta AI provide ESM Atlas, a database of predicted structures from the MGnify protein database, a dataset of protein sequences from metagenomics studies. Whilst the Protein Data Bank has only just reached 200,000 structures [145], the AF2 DB contains over 200 million [144], and the ESM Atlas contains over 600 million [137]. In addition to structure databases, DL approaches are also used for protein functional annotation in sequence databases [138].

1.5.2.3 In Silico Protein Design and Sequence Generation

The rational design of synthetic protein sequences has been studied for decades [146] and has many potentially useful outcomes. The ability to engineer proteins with desired structure or function *de novo* may allow the production of enzymes with increased stability, activity, or even novel chemistries [147]. It could be used to produce novel antibodies and biologic drugs with affinity against a specific target for use in therapeutic applications [148], or even develop new biomaterials [149]. Additionally, advances in protein design may also help answer a more fundamental scientific question, namely the mysterious relationship between protein sequence and structure, a problem which has puzzled scientists for over 60 years [150].

Up until very recently, there were two main approaches to tackle the protein design problem computationally, shown in Figure 1.5.5. Sequence-based approaches use evolutionary information from MSAs and HMMs to generate novel sequences [151]. However, they completely ignore protein structure data, instead using sequence as a proxy for structure. This not only ignores the wealth of information encoded in the three-dimensional structure of a protein, but it can also be confounded by the so-called "twilight zone" [152]. Proteins with less than 30% sequence identity can share the same fold (they are in the "twilight zone") [153], whereas a single residue mutation can be enough to completely destabilise a protein and remove function [154].

In contrast, structural methods (such as the widely used Rosetta package) use physics-based force fields, scoring functions, and energy functions to assess the stability of a given protein sequence or structure. This also allows generation of novel sequences predicted to fit within this protein structure, a process known as fixed-backbone design [155]. Generally, methods

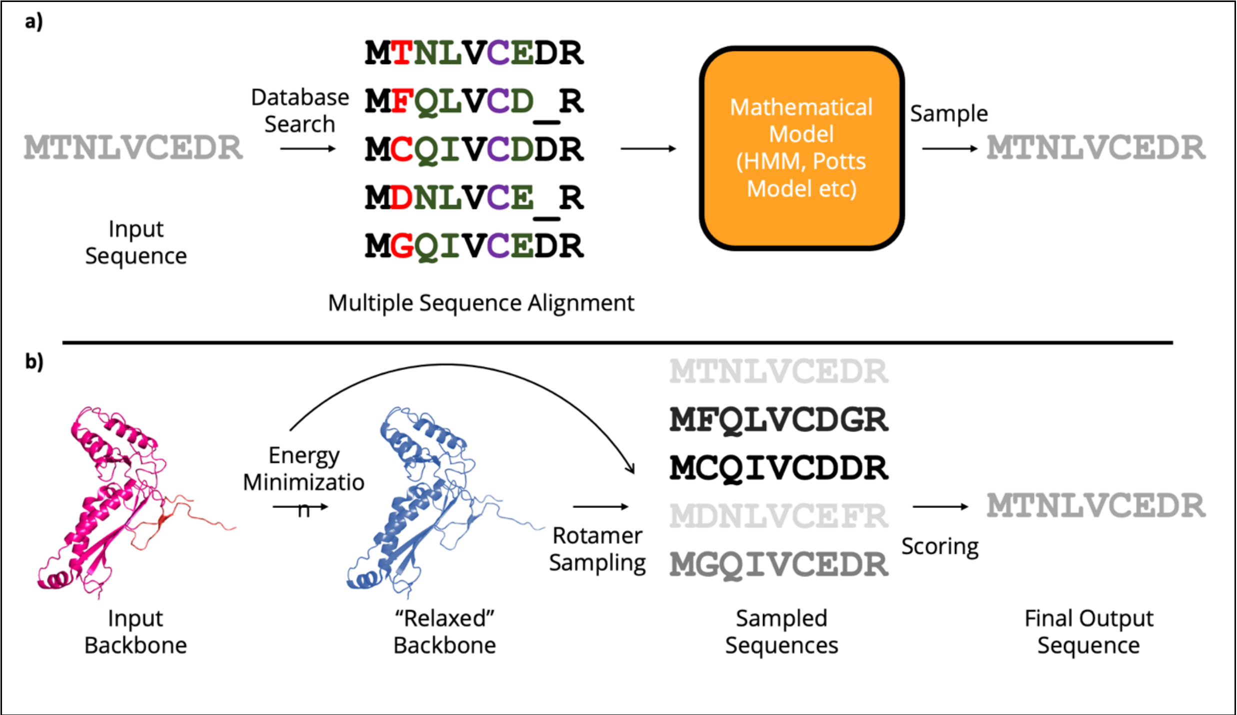


***Figure 1.5.5:* Traditional Approaches to Protein Design**
Sequence-based methods a) build a multiple sequence alignment (MSA) from an input sequence, by searching genetic databases. MSAs are used to infer evolutionary information, such as which positions are diverse, partially, or fully conserved (shown in red, green, and purple respectively). A mathematical model such as a Potts model can be inferred from the MSA, and the parameters of this model sampled to output sequences. Structure-based methods such as Rosetta's fixbb b) take an input backbone and use energy minimisation to produce a relaxed low-energy version of this backbone. This uses force fields and energy/scoring functions to find optimal rotamer combinations. These rotamer combinations can be used to produce new sequences.

like Rosetta have been used with some success in the past [156], however these bespoke physics-based simulations are very computationally expensive – some Rosetta protocols can take hours or even days to run [157]. Packages like Rosetta can also be difficult to learn and use, especially for non-experts.

1.5.2.4 Deep Learning for Protein Design

Protein design is yet another example of a domain where DL models can be applied successfully, involving complex relationships and patterns in protein sequence and structure data, but with an abundance of such data available today. Indeed various DL model architectures have been applied to protein design, including Recurrent Neural Networks (RNNs) [158] and Convolutional Neural Networks CNNs [159]. Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs) are two popular DL model architectures for generating text and images, and these have also been applied to protein design [160, 161]. Graph representations of protein structures have also been used to train deep neural networks for controllable fixed-backbone design [162].

More recently, the Transformer architecture has been used to generate new protein sequences, in fixed-backbone design [163] and in a controllable manner using keywords [164]. Diffusion models have also been applied to protein design. By training a diffusion model to reconstruct protein structures with noise added to the atomic coordinates, a deep learning model can learn to "hallucinate" new protein structures *de novo* in a controllable manner. These new backbones can then be used as input to a fixed-backbone design model to produce sequences predicted to fold into the desired structure. To this end, several groups have applied diffusion models to different protein design problems [165–168].

Crucially, many studies on DL models for protein design omit any experimental characterisation of their generated protein sequences. This is a big problem since most in silico sequence scoring or model evaluation metrics are imperfect and do not truly reflect the utility of protein design models. Models are often evaluated based on their ability to recreate natural sequences, by measuring percentage identity of generated sequences compared to held-out natural sequences [169]. Given the previously explained "twilight zone" problem in protein sequences, identity or similarity to known sequences is a poor metric for assessing protein sequence generation. A designed sequence could feasibly have very low identity to any known natural sequence but still exhibit the desired structure and function. Conversely, a designed sequence could differ by only a few residues from a wild-type protein, and yet fail to express or fold properly.

However, several recent landmark studies have presented DL methods for protein design where sequences were expressed and characterised experimentally. Salesforce Research trained a Transformer language model called Progen to generate protein sequences in a controllable manner using keywords [164]. They present functional assay data for designed lysozyme sequences and claim to have obtained the first crystal structure of an AI-generated protein. This work breaks new ground for protein design using DL, and its novelty and impact is not to be understated, however lysozyme sequences represent low-hanging fruit for protein scientists. Lysozymes are a good proof-of-concept because they are monomeric and easily crystallisable but have few (if any) interesting applications in the real world. Shin et al train a CNN to generate protein sequences and validate their model by experimentally screening a library of nanobodies [170]. This proof-of-concept is closer to a real application of protein design, but again only involves small, monomeric design. More recently, Verkuil and colleagues have adapted the ESM-2 protein language model to generate sequences in fixed backbone design [171] and experimentally validate 79 protein designs for 6 different targets using size-exclusion chromatography. This work presents an alternate angle on protein design using models trained on sequences only, however it only considers small monomeric proteins and so suffers from similar limitations as the above methods.

In contrast, Dauparas and coworkers present ProteinMPNN, a graph neural network for fixed-backbone protein design [162]. They design sequences for several monomeric and homo-oligomeric protein backbones and characterise their designs using size-exclusion chromatography, circular dichroism, X-ray crystallography, and negative-stain cryoEM. ProteinMPNN was also shown to be able to “rescue” insoluble protein designs from other methods. Later work from the same group presented RFDiffusion, a deep diffusion model capable of hallucinating protein structures in an unconditional manner or based on a given topology, oligomeric symmetry, or motif [165]. ProteinMPNN was then used to generate sequences for various monomeric and oligomeric backbones, and the resulting designs experimentally verified by size-exclusion chromatography and negative stain cryoEM.

More recently, Ingraham and colleagues have presented Chroma, a set of deep learning models for *de novo* design of protein backbones and sequences [172]. Their system jointly models structure and sequence, using a diffusion model to generate backbones and a graph neural network to design sequences for these backbones. Design can be constrained using a variety of inputs, including symmetry, secondary structure, domain labels from Pfam or CATH, and even text captions using natural language prompts. Notably, the authors screened 310 protein designs experimentally, and demonstrate that Chroma designs are soluble and well folded. Designs were validated using CD spectroscopy and two designs were crystallised and solved structures indicate close agreement with the design templates.

When taken together, the findings from Progen, ESM-2, ProteinMPNN, RFDiffusion, and Chroma represent a step change in *de novo* protein design, showing the utility of DL models for a variety of protein design tasks with real experimental evidence. However, as with all scientific claims, the experimental validation of DL models for protein design must be viewed with some scepticism. The models described above were all tested on arbitrary sets of proteins (lysozymes, nanobodies, and various *de novo* designed backbones) which may have been carefully chosen to optimise model performance. Since DL models suffer from poor interpretability, it is impossible to predict *a priori* how well these models will generalize to more difficult protein design targets which may have more interesting biotechnology or therapeutic applications. The authors of these models provide cursory data on the success rate of their designs – usually the total number of proteins expressed experimentally, along with how many of these proteins were soluble or showed the desired properties. Again, it is easy to manipulate these figures to make models appear more performant; it is unknown how many failed designs were tested outside of those presented in publications. Studies define success differently depending on the designed target; ProteinMPNN and RFDiffusion were evaluated based on the number of soluble designs, whereas Progen was evaluated based on protein expression level and solubility as well as enzymatic function. Given that different models are tested on different protein targets, it is impossible to compare their quoted success rates. Lastly, publications give no indication of how straightforward the *in silico* design process is, in terms of how many different model settings need to be tweaked and optimised before designs are chosen for experimental characterisation.

1.5.3 Metagenomics

Metagenomics is broadly defined as the study of microbial communities using genomic sequencing technology [173]. Traditionally, microorganisms are studied by isolating them from their natural environment, culturing them in the laboratory, and then sequencing and assembling their genomes (Figure 1.5.6a). This approach has a fundamental limitation - it can only be applied to organisms that can be easily cultured; and yet this set of culturable organisms is thought to be a tiny fraction of the microbial diversity seen on Earth [174]. In contrast, metagenomics approaches focus on analysing all nucleic acids purified directly from environmental samples (Figure 1.5.6b), regardless of their species of origin. Metagenomic samples can be taken from any environment, and thus the approach can shed light on microbial communities from diverse and often extreme locations, including soil, oceans, thermal springs, Antarctic ice, host-associated biomes such as human and animal digestive systems, and engineered biomes like wastewater treatment plants and laboratories [175].

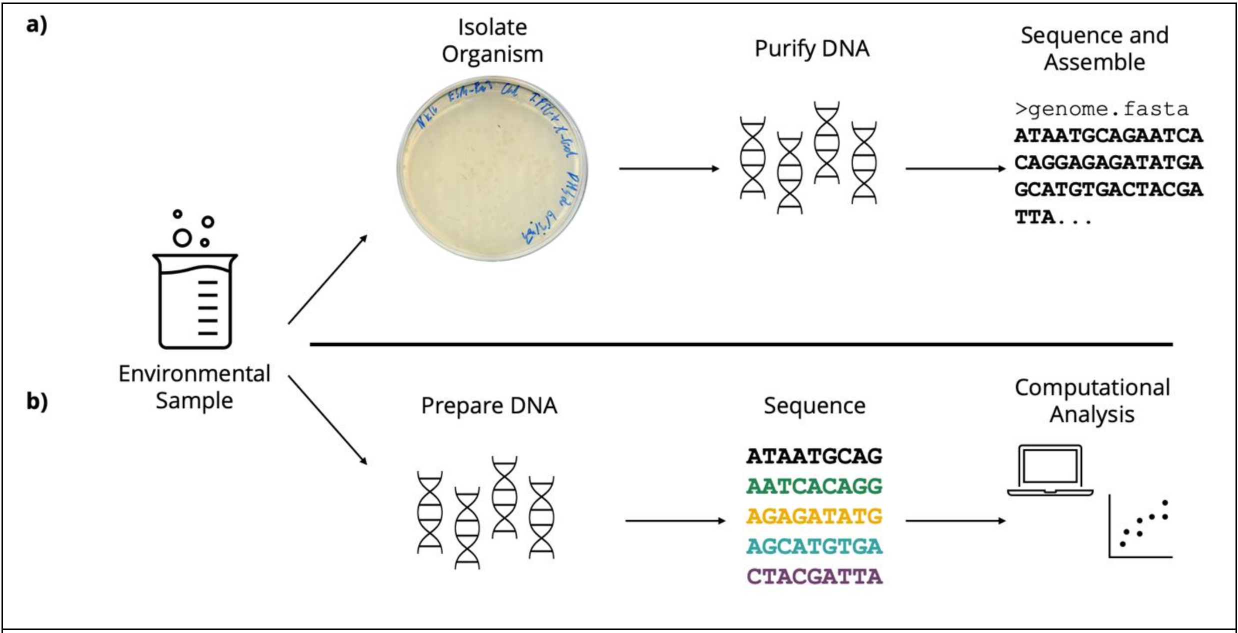


Figure 1.5.6: Metagenomics Versus Genome Sequencing
a) In single genome sequencing, the microbe of interest is isolated from an environmental sample and cultured in the laboratory under standardised growth conditions. Its genomic DNA can then be purified and prepared for whole genome sequencing. In contrast, **b)** metagenomics studies involve isolation of all DNA from the environmental sample before sequencing. This gives a broader picture of the microbial ecosystem but requires more involved data processing.

Experimental methods used in metagenomics studies can be grouped into two main classes. In marker gene methods, specific genomic regions are amplified and sequenced to reveal the species composition and diversity in a given sample [176]. For example, the 16S and 18S ribosomal RNA and internal transcribed spacer (ITS) regions are often used to identify prokaryotes and fungi present in a given sample [177]. Alternatively, whole genome shotgun (WGS) approaches aim to sequence all DNA in a sample and assemble short raw reads into

longer contiguous sequences (contigs). These contigs can be analysed to provide taxonomic and functional information about the organisms and genes present in a sample, including enrichment for certain protein functional annotations or pathways [178]. WGS methods can reveal more information across wider taxonomic groups than marker gene analyses and are free from bias that is introduced when amplifying marker genes – in a mixed population of organisms, marker genes from some organisms may be overrepresented when amplified. However, WGS also requires better quality sequencing data with a higher read count, is more computationally expensive, and requires much more involved data processing [179].

The growth of metagenomics as a field has resulted in an explosion of interest in microbiomes – the population of microorganisms that live inside or on the surface of the human body or other animals. This so-called "microbiome revolution" [180] has revealed that these microbial communities have a profound effect on human development, health, and in disease. WGS and marker gene experiments (and their accompanying computational data analysis steps) can reveal population-level information about the microbiome of the human digestive system, skin, and circulatory system – over different time courses, under different environmental conditions [181], after different drug treatments [182], or in certain disease states [183]. However, these experiments produce huge amounts of sequencing data which must be processed using complicated, computationally intensive metagenomic pipelines [184].

The rapid development of the metagenomics field has led to the continual growth of publicly available databases specializing in metagenomic data. Perhaps the most well-known metagenomics database is MGnify, which is run by the European Bioinformatics Institute and provides easy public access to thousands of metagenomics studies, including raw data, sample and analysis information, and assembled genomes [185]. Of particular interest for this study is the availability of large metagenomic protein sequence database such as the Big Fantastic Database [134] and the MGnify Protein Database [185]. Proteins in these databases are often referred to as biological "dark matter" since many show remote homology to experimentally characterised or functionally annotated proteins [186].

It has been hypothesised that this "dark matter" consists of proteins with novel functions, from new evolutionary families (or distantly related to known families), and containing undiscovered structural features [137, 187]. This wealth of proteins which explore uncharted sequence space has powered the deep learning revolution in protein science, providing the necessary training data for large, data-hungry deep learning models such as AlphaFold2 [134], ESMFold [137], and protein language models such as ESM2 [137] and Progen2 [188].

Metagenomic protein databases such as BFD and MGnify represent a goldmine of novel proteins which may have exciting uses in synthetic biology and biotechnology. However, this novelty is both a blessing and a curse; the remote or non-existent homology displayed by these proteins makes it difficult to predict their function and structure using traditional bioinformatics methods. Additionally, metagenomics studies usually do not produce fully assembled genomes, meaning that the genomic context surrounding any given protein is limited to the length of its contig, which can sometimes be barely longer than the protein itself.

This means that the organism of origin for a metagenomic protein cannot be easily determined, nor its surrounding genes. Despite these challenges, deep learning models have emerged as a viable alternative to classical bioinformatics tools for annotating metagenomic protein function [138], predicting structure [137], and finding biosynthetic gene clusters from metagenomics contigs [71].

1.6 **Experimental Techniques**

1.6.1 Protein Preparation for Biophysical Studies

1.6.1.1 Recombinant Protein Expression

Experimental structural biology and biophysics methods require abundant and pure, homogenous protein [189]. Traditionally, proteins were isolated and crystallised from natural sources, for example the red blood cells of humans or animals. [190]. However, advances in the field of molecular biology in the last quarter of the 20th century gave scientists the ability to manipulate any DNA sequence and introduce it into a model organism, such as the bacterium *Escherichia coli* [191]. This represented a paradigm shift in the field of structural biology, since any natural protein could now be expressed and purified in the laboratory using a well-characterised model organism. The choice of model organism depends on the properties of the protein being studied – *E. coli* is generally considered a "workhorse" of recombinant protein science, since it is fast-growing and easy to work with in the laboratory.

E. coli is suitable for recombinant expression of many bacterial and viral proteins, however it is naturally unable to perform many post-translational modifications present in eukaryotic proteins, such as glycosylation [192]. As such, recombinant expression of eukaryotic proteins is often most successful when performed in yeasts such as *Saccharomyces cerevisiae, Schizosaccharomyces pombe,* or *Hansenula polymorpha*, in filamentous fungi such as *Komagataella phaffii*, or in mammalian or insect cell cultures. However, sometimes bacterial proteins are also more stable and soluble when expressed in eukaryotic cells than in *E. coli.* This is seen in the case of the *S. elongatus* T1 encapsulin which is insoluble when expressed in *E. coli* but soluble when expressed from *K. phaffi* cells (Frank Lab unpublished data). The work presented here will focus solely on *E. coli* protein expression.

The maturity of genetic engineering and synthetic biology have led to a wealth of strategies for recombinant gene expression in *E. coli.* There are many options for controlling the expression level of a recombinant protein, at the DNA, RNA, and amino acid level, making recombinant expression tuneable in line with the characteristics of the target molecule. This is critical since protein expression experiments often involve a three-way trade-off between achieving a high expression level, minimising any adverse effects on *E. coli* growth and physiology from toxicity effects, and solubility and correct folding of the target protein. While heterologous genes can be stably inserted into the bacterial chromosome [193], the most common method of introducing foreign genes is via plasmids – circular, double-stranded DNA elements that replicate independently of the bacterial genome [194]. Plasmid replication is regulated by a region known as the origin of replication, which controls the copy number – the average number of plasmid molecules present in each *E. coli* cell [195]. The choice of a different plasmid origin can be used to tune protein expression levels. Plasmids also encode antibiotic resistance genes, which allow for selection of cells harbouring the plasmid of interest and removal of any "background" cells without it.

Expression levels and timing can also be controlled using gene regulatory elements. On a plasmid, the gene of interest is flanked by an upstream promoter and ribosome binding site (RBS). These elements facilitate the transcription initiation and translation of the gene respectively. Different promoter and RBS sequences have different "strengths" and can be used to fine tune protein expression. In addition, some promoter sequences are "inducible" and only allow the target gene to be transcribed when an external signal is present [196]. The de-facto standard in many structural biology labs is to use the *T7* promoter system (Figure 1.6.1). The recombinant protein is not transcribed until IPTG is added to the culture, allowing tight control of the timing of protein expression. This decoupling of cell growth and protein expression allows activation of protein production at the optimal point in the bacterial growth curve, which in *E. coli* is usually the mid-logarithmic phase of growth when protein translation is maximal [197].

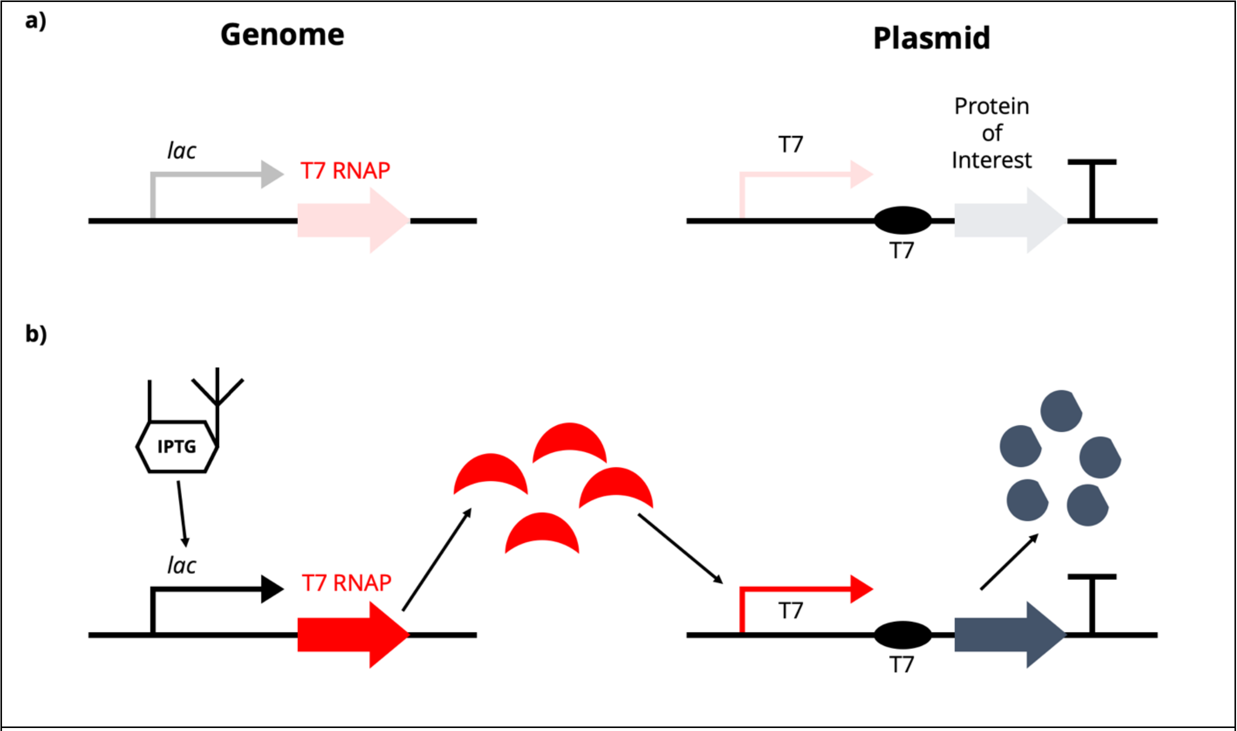


Figure 1.6.1: **The *T7* Promoter System for Inducible Protein Expression**
An *E. coli* strain is used containing a chromosomal prophage encoding the phage T7 RNA polymerase. This gene is driven by a *lac* promoter inducible by the small molecule Isopropyl ß-D-1-thiogalactopyranoside (IPTG). The protein of interest is introduced to this strain on a plasmid, and the gene encoding this protein driven by a *T7* promoter, which is not recognised by the wild-type polymerases. a) In the absence of IPTG, the T7 RNAP is not expressed, and neither is the protein of interest. However, when IPTG is added b) transcription and translation of the T7 RNAP can proceed, which drives expression of the recombinant protein.

The sequence of the recombinant protein itself is another variable in a protein expression experiment. The genetic code contains inbuilt redundancy, with some amino acids encoded by 2-6 different “synonymous” codons. However, different organisms use different synonymous codons in protein coding genes – this is known as codon bias [198]. When expressing recombinant genes in *E. coli*, the codons in the native DNA sequence are often swapped for codons that are more preferentially used in *E. coli*. This process is known as “codon optimisation” and has been shown to improve expression of genes from different organisms [199]. Mutants of the wild-type protein sequence can also be made to study function and biological mechanism, and fusion proteins added to the C- or N-terminus to improve solubility, activity, or add new functions [200]. In protein crystallography, truncations of the wild-type protein are often used, since the presence of disordered or flexible regions at the termini of a protein can reduce the likelihood of obtaining well-diffracting crystals [201]. Lastly, so-called “affinity tags” can be added to the termini of the protein (or sometimes internally) to aid in purification, as explained below [202].

1.6.1.2 Protein Purification

Once a recombinant protein of interest has been successfully expressed to a high level in *E. coli*, it must be recovered and purified for further studies, as shown in Figure 1.6.2. Before purification, lysed cells are separated into soluble and insoluble fractions by centrifugation or filtration, and these fractions analysed by SDS-PAGE. If the protein of choice is in the soluble fraction, then this can immediately be used for downstream purification, however if the protein is insoluble then conditions and buffers may need optimisation. In some cases insoluble protein in *E. coli* can form aggregates known as “inclusion bodies”, which can be solubilised and used for downstream purification thereafter [203].

Several different chromatography methods are used for protein purification in structural biology, the most common of which is affinity chromatography. In this method, an “affinity tag” is added to the C- or N-terminus of the protein-coding region in the expression plasmid. When the protein is expressed, this affinity tag is capable of specific binding to a ligand incorporated into a solid matrix on a chromatography column [204]. A commonly used affinity tag is the poly-histidine tag, usually a run of 6 consecutive histidine residues conferring specific adsorption to nickel ions on a chromatography column. The cell lysate is passed over the affinity column, where the His-tagged recombinant protein binds specifically to the column. Contaminants either do not bind at all or bind weakly in a non-specific manner and can be washed away. Finally, the protein of interest can be eluted from the column using a high concentration of imidazole (typically 0.5 M [204]). Other chromatography methods are also used instead of or in combination with affinity chromatography. These include ion-exchange chromatography which separates proteins based on their isoelectric point [201] and hydrophobic interaction chromatography, which separates based on the hydrophobicity of proteins [205].

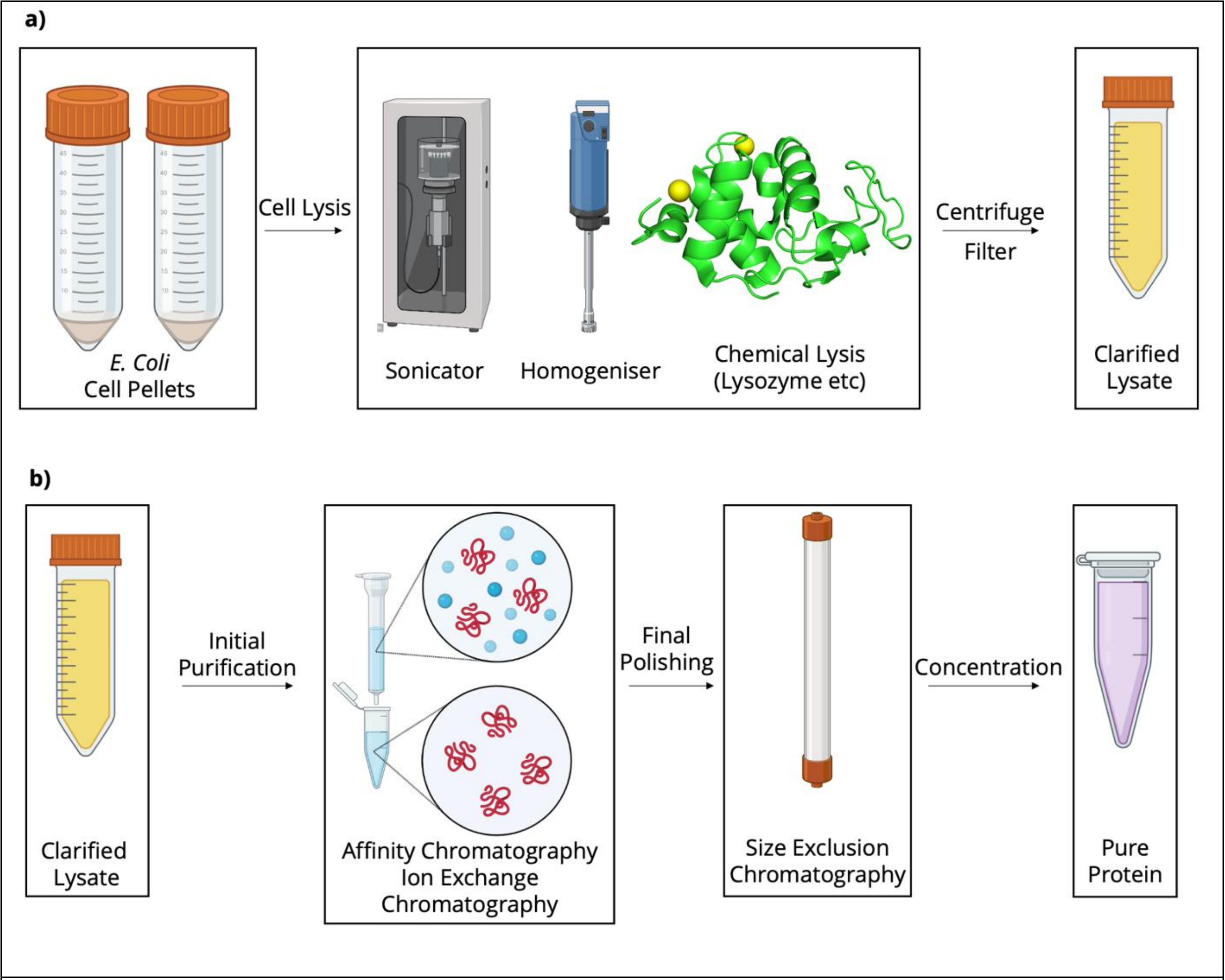


***Figure 1.6.2:* Purifying Recombinant Proteins from E. coli**
a) Following a protein expression experiment, *E. coli* cell pellets containing the protein of interest are obtained. These cells must be resuspended in a suitable buffer and lysed. Lysis can be carried out mechanically using a sonicator or a homogeniser, or chemically using lysozyme or similar chemical lysis buffers [107]. This crude lysate is centrifuged, and supernatant filtered to recover clarified, soluble lysate. **b)** Protein purification from clarified lysates usually begins with an initial purification step of affinity chromatography or ion-exchange chromatography. One or more of these initial steps are used to remove bulk contaminants before size exclusion chromatography is used as a final "polishing" step.

Following initial chromatography steps, proteins are often subjected to a final “polishing” step to ensure a high degree of purity for crystallography. This is typically size exclusion chromatography (SEC), sometimes known as gel filtration. Here, protein samples are passed through a column containing a porous gel matrix, which separates proteins by their hydrodynamic volume (a proxy for molecular weight). Larger proteins will tend to be “excluded” from the pores and pass quicker through the column, whereas smaller species will pass through the pores more slowly [206]. Some very large species may not fit through the pores at all and elute very early in the “void volume”. As such, SEC can be used to separate proteins based on their size and is commonly used as a final chromatographic step in a protein purification experiment, to yield a highly pure sample for crystallography. The shape of SEC absorbance peaks can also give some indication of the homogeneity of a sample.

1.7 Aims of this Study

This study has two main aims pertaining to encapsulins. The first is to discover new encapsulin systems from metagenomics databases. These databases have grown rapidly in size over recent years and are thought to be a rich source of novel proteins with interesting structural and functional properties. Concurrent advances in deep learning methods for protein structure and function prediction can be applied to these large, challenging datasets to shed light on new areas of protein sequence space. This may result in the discovery of new encapsulins with novel structural or physical features, or new cargo protein types. These findings may improve understanding of the biology and evolution of protein-based organelles, or perhaps even viral proteins overall. Any new encapsulins discovered may also have properties amenable to biotechnology or therapeutic applications.

The second aim of this work is to re-design an existing encapsulin protein to increase its solubility and expression yield. These properties are of great importance in industrial biotechnology applications. Multiple deep learning models and methods are used to generate encapsulin sequences at scale *in silico*. The resulting sequences are scored and evaluated using a variety of different computational techniques, to better understand the behaviour of these models on an industrially relevant “real world” use case. Successful designs are screened experimentally in a high-throughput manner using liquid handling robots and automation techniques, and stable candidates characterised further using biophysical and structural biology techniques. The utility of this aspect of the work is twofold; it will aid in understanding the behaviours and limitations of deep learning models for protein design, as well as potentially generate novel encapsulin proteins with desirable properties for biotechnology and synthetic biology.

A final, secondary aim of this study is to investigate the effect of the E-loop region on encapsulin expression and assembly.

1.7.1 Previous work

There are several previous studies focused on encapsulin discovery using well-established bioinformatics methods [18, 19, 26, 207, 208]. However, this work purposefully ignored proteins from metagenomic databases, thus neglecting a potentially valuable source of data. These studies also took place before widespread availability of deep learning tools for protein structure and function prediction [134, 138]. To date, no encapsulin bioinformatics study has leveraged metagenomics databases, protein structure and function prediction tools, and biosynthetic gene cluster prediction methods on a large scale.

Whilst there are countless studies on protein design in the literature, particularly in the age of deep learning, precious few of these works show any kind of laboratory validation of their designs. However, there are several examples in the literature of *de novo* design of symmetrical protein oligomers, with experimental validation. Deep learning tools have been used to design tetrahedral protein nanoparticles [162] as well as small (≈15 nm diameter) icosahedral protein assemblies [165], and in both cases successful designs were expressed, purified, and validated by cryoEM. However, these nanoparticles are of a different size and shape to encapsulins.

Bale et al report *in silico* design of larger 24-40 nm icosahedral capsids which are closer in scale to encapsulins [156], but with different assembly properties. These capsids were formed from two different protein monomers assembling into 120-subunit shells as confirmed by small-angle X-ray scattering, negative stain electron microscopy, and crystal structures. However, this work was done before the advent of deep learning techniques for protein design, and as such made use of a complicated manual design procedure. This process involved retrieving and filtering hundreds of protein structures from the PDB, as well as protein-protein docking, and manual interface design using the Rosetta software package.

More recently, Lutz and colleagues [209] demonstrated *de novo* design of 60-subunit icosahedral capsids using reinforcement learning, a type of machine learning technique, and validated their method with two cryoEM structures of 10 and 13 nm capsids respectively, as well as some proof-of-concept work in displaying antigens on the surface of these capsids for potential vaccine design. Whilst this work is very impressive, these designs are much smaller than encapsulin nanocompartments, and the reinforcement learning method used to design the novel protein backbones isn't strictly a deep learning technique. Whilst deep learning tools were used to design sequences for these backbones, the design process still relied on manually derived scoring methods and screens, using scoring functions adapted from tools like RPXDock [210], a symmetric protein-protein docking package.

A previous study in the literature has also demonstrated re-design of existing proteins for increased function, solubility, and yield, using ProteinMPNN, a deep learning method for protein design [211]. Multiple protein designs were experimentally confirmed to show higher soluble yield and function following expression in *E. coli*, purification, functional assays, and X-ray crystallography. Whilst these aims align closely to the goals of this study, there are still some key differences and limitations between this previous work and the present study. For one, the ProteinMPNN paper focused on two relatively small, single subunit proteins; a tobacco etch virus (TEV) protease, and myoglobin. These will obviously present different design challenges compared to large, symmetrical encapsulin assemblies. A central theme of the ProteinMPNN paper was the trade-off between increasing soluble protein yield and preserving protein function. Again, the dynamics of this trade-off are likely to be different in a self-assembling protein complex compared to an enzyme or cofactor-binding protein. Lastly, this work only investigated ProteinMPNN as a design tool, whereas this work aims to characterise several different deep learning tools.

The present work, in contrast, aims to perform re-design of encapsulin nanocompartments using deep learning methods. This may result in the discovery of an optimal "pipeline" for protein design and *in silico* validation which is much more approachable by non-expert protein designers compared to conventional tools. Whilst the literature offers many examples of deep learning-based protein design, no previous work has characterised and compared multiple deep learning models on the same design problem and included experimental results. This work aims to achieve this goal by testing deep learning tools of different architectures that are trained on diverse datasets, and then experimentally validate design quality.

Chapter 2: **Materials and Methods**

2.1 Metagenomic Encapsulin Discovery

This section describes methods pertaining to experiments described in detail in Chapter 3. While the focus in this section is on providing a comprehensive overview of the methodological approach, the reader is encouraged to refer to Chapter 3 for detailed exposition of how methods were used in experiments, along with supporting figures (to aid comprehension, a schematic workflow diagram is shown in Figure 3.2.1).

2.1.1 Database Searches

To discover novel encapsulin sequences, a combined sequence annotation and structure-based search approach was used. The 2022/05 release of the MGnify Protein Database [185] was filtered to recover all accessions with Pfam annotations from clan CL0373 (phage coat), which contains all HK97 fold-associated Pfam families. These annotations are generated using a convolutional deep neural network tool instead of the traditional Hidden Markov Model (HMM) method used by Pfam [212], which has been demonstrated to assign function more accurately in cases where sequence homology is remote or non-existent [213]. In tandem, structure searches were performed against the 2023/02 release of ESM Atlas [137] using experimentally solved structures of the T=1 encapsulin from *T. maritima*, the T=3 encapsulin from *M. xanthus*, the T=4 encapsulin from *Q. thermotolerans*, and the T=1 encapsulin from *S. elongatus* (PDB codes 7MU1, 7S2T, 6NJ8, and 6X8M respectively). ESM Atlas structures with pTM scores of 0.7 and higher were downloaded using aria2c, and structure searches were performed using Foldseek [214] with the "easy search" workflow and a minimum coverage of 0.5. Structure database search was carried out on VM.Standard.E4.Flex cloud instance with 64 cores and 1024 GB of RAM (Oracle Cloud).

2.1.2 Removing Phage-Associated Sequence Contamination

Genomic contigs for each search hit were retrieved using a combination of text-based filtering using bash, and API calls using Python. To retrieve contigs for each search hit, an MGYC contig accession was obtained from the MGnify Protein Database for each search hit. For each MGYC accession, a corresponding European Nucleotide Archive analysis accession (ERZ) and MGnify contig name was also retrieved from the MGnify Protein Database. Lastly, a MGnify analysis accession (MGYA) was obtained for each ERZ accession using the MGnify API. This API was also used to obtain protein coding sequences (CDS) for each search hit, using the hit's respective ERZ and MGYA accession and contig name. Any search hit with missing MGYC accession contig CDS was removed from the dataset. Lastly, any search hit with contigs under 25 kilobases in length or containing fewer than 10 protein sequences was also removed from the dataset – this is to ensure that every search hit has enough genomic context available to confidently filter out phage-associated proteins and provide functional information for encapsulin hits.

As in previous work [26], a custom mmseqs2 database [215] was prepared from two phage proteome datasets: one containing all proteins from Bacteriophage HK97, and one containing proteins from a broader set of prokaryotic tailed dsDNA viruses (UniProt proteome accessions UP000002576 and UP000391682 respectively). This database was searched using candidate contig protein sequences as query; searches were performed using mmseqs2 with the iterative search function, a starting sensitivity of 4, a final sensitivity of 7, and 5 sensitivity steps (all further searches in this study used these parameters unless otherwise stated). Any search hits whose contigs contained mmseqs2 hits against these two phage proteomes were removed from the dataset. Additionally, every contig protein's Pfam annotations were retrieved from the MGnify Protein Database. These annotations were screened against a manually curated set of 279 phage-associated Pfam families, and any search hits whose contigs contained these proteins were removed from the dataset.

Finally, to ascertain maximum sequence identity with proteins in conventional databases, putative encapsulin hits were searched against the UniRef90 database (downloaded 2023-03-30) [216] using mmseqs2 with previously mentioned parameters, and "--max-accept" set to 1. UniRef90 database search was carried out on a VM.Standard.E4.Flex cloud instance with 64 cores and 1024 GB of RAM (Oracle Cloud). The taxonomy ID of each encapsulin candidate was used to retrieve its taxonomic lineage using the UniProt API, and 2 encapsulin sequences showing >95% identity to UniRef90 sequences from the superkingdom "viruses" were removed from the dataset.

2.1.3 Encapsulin Structure Prediction and Analysis

Where available, encapsulin search hit structure predictions were retrieved from ESM Atlas using the public API. However, most candidate sequences had no available structure prediction data. For these candidates, structure prediction was carried out using ESMFold [137] in Google Colaboratory [217] with a chunk size of 64 for sequences larger than 700 amino acids, and 128 for sequences smaller than 700 amino acids. Structures for putative encapsulins longer than 900 amino acids were not predicted due to computational constraints, and any structure predictions with a mean predicted local distance difference test (pLDDT) value under 70% were removed from the dataset.

Confident predicted structures were analysed using DALI [218] to compute all-against-all pairwise Z-scores. Experimentally solved structures for four well-characterised encapsulin proteins were also included, from *T. maritima*, *M. xanthus*, *Q. thermotolerans*, and *S. elongatus* (PDB codes **7MU1**, **7S2T**, **6NJ8,** and **6X8M** respectively). The similarity matrix was manually inspected, and a set of 130 structures showing extremely low similarity to all others were removed and manually assigned to their own dissimilar cluster. The remaining matrix was used as input for hierarchical clustering with complete linkage using the scipy.cluster.hierarchy package [219]. The protein sequences within each cluster were then clustered at 80% sequence identity cutoff using mmseqs2 to reduce redundancy and facilitate easier manual inspection [215] and predicted structures for each cluster were visually inspected using PyMOL [220]. All plots were created and inspected using the Plotly package in Python [221].

Representative sequences from each cluster of ESMFold predicted structures were also predicted using AlphaFold2 [134]. This was done to demonstrate that structure prediction with ESMFold is comparable and does not lead to exclusion of encapsulin structures due to inferior performance. Structures were predicted with AlphaFold2 v2.3.0 using default MSA settings, and a maximum template cutoff date of 1st December 2023. For structure clusters with fewer than 15 sequences (after mmseqs2 clustering at 80% identity), all sequences were predicted. In the case of larger structure clusters, a single representative sequence was chosen based on the lowest mean DALI Z-Score to every other member of the cluster.

2.1.4 Encapsulin Cargo Protein Annotation

Initially, all contig protein Pfam annotations were manually inspected to assign encapsulin cargo type and biological function. Two sets of Pfam annotations were considered in this study: Pfam family annotations from the MGnify Protein Database which are generated using ProtENN, a deep learning tool based on convolutional neural networks [138], and more conventional HMM-based Pfam assignments generated using HMMScan as part of DeepBGC [71]. A comprehensive set of cargo types has been previously published [26], however this work did not assign every cargo type a Pfam family or set of Pfam families. For this current study, the known cargo Pfam families were enriched with further manually curated Pfam families which were used to annotate some family 1, 2, and 4 encapsulin cargo proteins. However, since most putative encapsulins still had no family or cargo protein assigned, a more involved strategy was needed.

Additional Family 1 cargo proteins were identified by searching an mmseqs2 database containing all contig proteins, using as query the family 1 cargo loading peptide (CLP) consensus sequences and secondary cargo CLP sequences from [18]. mmseqs2 search parameters were optimised for short query sequences by using the PAM30 Matrix, an E-value cutoff of 200,000, and setting "spaced-kmer-mode" to 0.

Additional Family 2 cysteine desulfurase (CyD) cargo proteins were identified using the same search parameters with a conserved motif (LARLANEFFS) found in the disordered N-terminal domain (NTD) of CyD from the *S. elongatus* family 2 encapsulin system [19]. Further Family 2 cargo proteins were discovered using Hidden Markov Model (HMM)-based searches. For the four known cargo types (cysteine desulfurase, xylulose kinase, polyprenyl transferase, and terpene cyclase) sequence accessions from [26] were collected and sequences retrieved from UniProt [216]. MSAs for each cargo type were built using Clustal Omega with default parameters [130], and HMMs produced from these MSAs using the hmmbuild utility from HMMer with default settings [131]. The hmmsearch utility from HMMer was used to search these profile HMMs against all putative cargo proteins and any hits with E-value less than 1 were reported.

Further cargo annotations were carried out using sequence similarity by searching all putative cargo proteins against the NCBI non-redundant protein database [222] using mmseqs2 with the previously mentioned parameters and --max-accept set to 30. NCBI non-redundant database search was carried out on a VM.Standard.E4.Flex cloud instance with 64 cores and 1024 GB of RAM (Oracle Cloud).

2.1.5 Biosynthetic Gene Cluster Prediction

BGCs were predicted from encapsulin-containing metagenomic contigs using two different approaches. The antiSMASH 6.1.1 package [70] was used to predict BGCs from contigs with the following settings: Prodigal was used as the genefinding tool, ClusterBLAST was used with the general, subclusters, and knownclusters settings, active site finder was enabled, and the pfam2go and clusterhmmer options were enabled. antiSMASH outputs in HTML format were parsed using the Beautifulsoup4 package in Python [223]. In parallel, DeepBGC [71] was also used to predict BGCs, using Prodigal in metagenomic mode for gene finding, the "deepbgc" detector, and classifiers "product_class" and "product_activity". Predicted clusters were filtered to remove any clusters ending more than 10 kb upstream of the putative encapsulin gene or beginning more than 10 kb downstream of the gene.

2.2 *In silico* Encapsulin Design

Similarly to Section 2.1, this section describes methods used in experiments from Chapter 4. Again, the reader is encouraged to refer to Chapter 4 for detailed exposition of how methods were used in experiments, along with supporting figures. A schematic workflow diagram is presented in Chapter 5, Figure 5.2.1.

Unless otherwise stated, analyses in this section were run on BM.GPU.A10.4 cloud machines with 64 CPU cores, 1024 GB of RAM, and 4x Nvidia A10 GPUs with 24 GB VRAM each (Oracle Cloud). Experimental logs and notes along with accompanying plots and code snippets are publicly available at https://github.com/naailkhan28/bioinformatics_lab_notebook .

2.2.1 Structure Prediction Benchmarking

The performance of three different structure prediction methods (AlphaFold2, ESMFold, OmegaFold) was benchmarked across a representative set of experimentally solved protein structures containing the HK97-fold (see main text for details). AlphaFold2 [134] v2.3.2 was installed using the non-Docker method documented at https://github.com/kalininalab/alphafold_non_docker and predictions were run using default settings, with the max template cut-off date set to 1st January 2023. ESMFold [137] and OmegaFold [224] were installed using instructions from their respective GitHub repos, and predictions run with default settings. Note that the "—model 2" setting for OmegaFold was not used, since this drastically increased GPU memory usage. TM-Scores were measured against the template using TMalign [225] and mean pLDDT values were calculated from the B-factors in the PDB file. All code, scripts, protein sequences and structures, predicted structures, and analysis notebooks are publicly available at https://github.com/naailkhan28/hk97_structure_prediction_benchmarking .

2.2.2 Inverse Folding Methods

Inverse folding experiments were carried out using ESM-IF [163] or ProteinMPNN [162]. Models were installed and run according to the authors' instructions on GitHub. The sampling temperature value for sequence generation was varied according to the experiment (see text for details). The standard “vanilla” ProteinMPNN model weights were used as opposed to the “soluble” model weights (model trained only on soluble proteins), since these “soluble” weights didn't significantly alter the quality of generated sequences. The ProteinMPNN model trained with 0.2 Å noise was used in all experiments. No masking was used for either model.

2.2.3 Markov-Chain Monte Carlo Methods

MCMC protein design was carried out using either the “Protein Programming Language” model described in [226] or the “ProteinLM” method from [171]. The number of Monte Carlo iterations for each model was varied based on the experiment (see text). ProteinLM was used with default parameters and random seed set to 0. Protein Programming Language design was carried out using the default energy constraints for fixed backbone design, namely pLDDT, pTM, surface hydrophobicity, CRMSD, and DRMSD. Scripts were based on examples available in the ESM GitHub repo [227]. Documentation and code for Monte Carlo design testing is available at https://github.com/naailkhan28/monte_carlo_design .

2.2.4 ESM-2 Protein Language Models

2.2.4.1 Fine-Tuning

PLMs were fine-tuned on a representative dataset of HK97-fold sequences. UniRef accessions for ≈55,000 sequences with Pfam annotations from clan CL0373 (phage-coat) were obtained from the InterPro database [228] and downloaded from UniProt (accessed April 18th 2023). Sequences were clustered at 30% identity using mmseqs2 [215] and split into train, validation, and test sets (75%, 15%, and 10% of sequences respectively). Splits were made randomly, but constructed such that the training and validation sets contained no sequences sharing more than 30% identity with any solved HK97 fold protein structure. This is to allow proper evaluation of fine-tuned models on these experimentally validated sequences. Sequences were truncated to a maximum length of 768 residues during fine-tuning due to memory constraints.

ESM-2 models were trained using the HuggingFace Transformers library in Python [229] on the masked language modelling task. All model parameters were trained during fine-tuning, with no frozen layers or parameters. Models were fine-tuned using cross entropy loss and the Adamw [230] optimiser with parameters as follows: $\beta 1 = 0.9$, $\beta 2 = 0.99$, $\varepsilon=10^{-8}$. Hyperparameter sweeps were also carried out to determine the optimal learning rate, weight decay, and number of epochs; hyperparameter sweeps and monitoring of training runs was carried out using the Weights & Biases library [231]. The final ESM-2 650M fine-tuning run was carried out for 10 epochs, with learning rate of 10^{-5} and weight decay of 0.05. Models were fine-tuned on the training set only, with the validation set used to monitor overfitting.

2.2.4.2 Model Evaluation

Both existing pre-trained and fine-tuned ESM-2 models were evaluated on the unseen test set. Perplexity was calculated on this test set by taking the exponential of the cross-entropy loss calculated over all sequences. Models were also evaluated on contact prediction, using the HK97-fold structure dataset described in Table 4.1. Contact prediction was carried out using the contact prediction head for each ESM-2 model, and accuracy reported as the P@L5 metric as in [137].

2.2.4.3 Sequence Generation

Protein sequences were generated using ESM-2 models via an iterative masking procedure as in [232]. In brief, the input template sequence has a fraction of its residues replaced with mask tokens. At each of these masked positions, the language model produces a probability distribution over all 20 amino acids. This probability distribution is then sampled from, and the mask token replaced with a sampled residue. The fraction of residues replaced with mask tokens varies depending on the experiment. Notably, and unlike [232], for a given masking fraction, the number of residues masked is not fixed, but is chosen from a binomial distribution where n is equal to the length of the sequence and p is equal to the desired masking fraction.

The masked sequence is provided as input to the PLM and likelihoods calculated for all 20 amino acids at a randomly chosen masked position. These likelihoods are converted to probabilities (i.e., scaled between 0 and 1 and to sum to 1) using the "torch.softmax" function in PyTorch. In the first instance, a residue is sampled from this probability distribution using the “torch.multinomial” function from PyTorch, with a temperature of 1 (i.e., no scaling of the probability distribution). In sampling experiments, alternative sampling methods such as top-*k* sampling and nucleus sampling are used, as described in [233]. Multinomial sampling was also used with temperature values other than 1. Here, the likelihoods are divided by the temperature value before being passed as input to the softmax function.

2.2.5 Ankh Protein Language Models

2.2.5.1 Sequence Generation

With the Ankh family of PLMs, two different generation schemes were tested: masked language modelling (MLM) and autoregressive generation. MLM generation is analogous to the strategy outlined in Section 2.2.4.3, where an input sequence is randomly masked and “filled in” with new residues according to the probability distribution of a PLM. In autoregressive generation, however, mask tokens are added from the end (i.e., the C-terminus) of a protein sequence. This can be thought of as truncating a protein sequence and then replacing the truncated residues with mask tokens. In both cases, sequences were generated using multinomial sampling with the “model.generate()” function in the HuggingFace Transformers library in Python [229]. The following parameters were used for model.generate(): “repetition_penalty” was set to 3.0, “temperature” was set to 0.3, and “no_repeat_ngram_size” was set to 3, to avoid repeats of more than 3 amino acids. Since this model generates sequences with no fixed length, the minimum and maximum length of generated sequences was set to the same length as the template sequence to constrain the length of generated protein sequences.

2.2.5.2 Fine-Tuning

Ankh PLMs were fine-tuned on the same dataset described in Section 2.2.4.1 and using the same optimiser and settings but using the causal language modelling task instead of masked language modelling. During causal language modelling, sequences are split into chunks, and in training these chunks are presented to the model which is tasked with correctly predicting the next residue. The length of these chunks is defined in the context length parameter. Hyperparameter sweeps were used to determine the optimal context length, as well as learning rate, weight decay, batch size, and number of epochs as before. Final Ankh Base and Large fine-tuning runs were carried out for 20 epochs, with learning rate of 10^{-3}, weight decay of 0.1, context length of 20, and batch sizes of 64 and 24 for the Base and Large models respectively. Models were fine-tuned on the training set only, with the validation set used to monitor overfitting.

2.2.6 Evaluation of Designed Sequences

Designed protein sequences were evaluated using several different methods. Structures were predicted using ESMFold [137] as described in Section 2.2.1. For experimental candidates, structures were also predicted using AlphaFold2 [134] with single sequences (ssAF2) as described in [211]. Here, AlphaFold2 is used as described in Section 2.2.1, but instead of using the default MSA generation pipeline, a Python script is used to make a "dummy" MSA containing only the template sequence. Prediction is run with the max template cutoff set to the date when analysis was run (meaning that all templates can be used for prediction).

To ascertain maximum sequence identity with proteins in conventional databases, designed protein sequences were searched against the UniRef90 database (downloaded 2023-03-30) [216] using mmseqs2 with the iterative search function, a starting sensitivity of 4, a final sensitivity of 7, and 5 sensitivity steps, and "--max-accept" set to 1. Solubility of designed protein sequences was predicted with NetSolP [234] using the "distilled" model, and both solubility and usability predictions were used.

2.3 Experimental Characterisation of Encapsulin Candidates

2.3.1 *E. coli* Cultivation and Microbiology

2.3.1.1 Strains, Growth Media, and Antibiotics

All *in vivo* experiments in this work were carried out in *E. coli*. The *E. coli* strains used are shown in Table 2.1.

Liquid growth media are detailed in

Table 2.2. Solid growth medium was made from Miller's LB-agar powder (Sigma Aldrich) dissolved in distilled water at 37 g/L. The solution was autoclaved, allowed to cool to ~50 °C, and poured into sterile petri dishes (25 ml agar per plate). Solid and liquid growth media were supplemented with 25 µg/ml chloramphenicol, from 25 mg/ml stocks kept at -20 °C. Antibiotics stocks were made up in 100% ethanol and filter sterilised using a 0.2 µm filter (Sartorius).

***Table 2.1:* *E. coli* Strains Used in This Work**
T7 Express cells were used for recombinant protein expression. This strain contains a chromosomal prophage encoding the T7 polymerase, driven by an IPTG-inducible promoter, for inducible protein expression (as explained in Section 1.6.1.1). DH5α cells are deficient in the endonuclease *endA1* and the recombinase *recA1,* making them well-suited for cloning and preparation of pure plasmid DNA.

Strain Name	Genotype	Source	Usage
T7 Express	*fhuA2 lacZ::T7 gene1 [lon] ompT gal sulA11 R(mcr-73::miniTn10--TetS)2 [dcm] R(zgb-210::Tn10--TetS) endA1 Δ(mcrC-mrr)114::IS10*	New England Biolabs	Protein Production
DH5α	*fhuA2Δ(argF-lacZ)U169 phoA glnV44 Φ80Δ(lacZ)M15 gyrA96 recA1 relA1 endA1 thi-1 hsdR17*	New England Biolabs	DNA Assembly and Plasmid Preparation

***Table 2.2:* Liquid Growth Media**
LB and TB media were supplied as powders (Sigma Aldrich), dissolved in distilled water (at 20 g/L and 47.6 g/L), and autoclaved. SOC medium components supplied by Sigma Aldrich. To prepare SOC medium, all components except glucose are dissolved in distilled water and autoclaved. Separately, a 200 mM glucose solution is prepared in distilled water and filter sterilised. Immediately before use, the stock solution is added to a final concentration of 20 mM glucose.

LB	Terrific Broth (TB)	Super-Optimal Broth with Catabolite Repression (SOC)
10 g/L tryptone 5 g/L yeast extract 10 g/L NaCl	20 g/L tryptone 24 g/L yeast extract 4 ml/L glycerol 0.017 M KH2PO4 0.072 M K2HPO4	20 g/L tryptone 5 g/L yeast extract 10 mM NaCl 2.5 mM KCl 10 mM $MgCl_2$ 10 mM $MgSO_4$ 20 mM glucose

2.3.1.2 Bacterial Transformation

Transformations were carried out using commercial competent cell kits (New England Biolabs) in strains listed in Table 2.1. Competent cells were stored at -80 °C and allowed to thaw on ice for 5 minutes before use. 10 ng of plasmid DNA was added, and cells allowed to incubate for 30 minutes on ice, before a 60 second heat shock at 42 °C in a water bath. After a further 5 minutes incubation on ice, 1 ml of liquid SOC medium was added, and cells incubated at 37 °C for 60 minutes at 220 rpm shaking. After incubation, 135 µl of cells were added to a freshly poured agar plate and spread using sterile L-shaped spreaders. Plates were incubated at 37 °C overnight before further use.

2.3.2 DNA Assembly and Molecular Biology

2.3.2.1 Polymerase Chain Reaction (PCR)

Primers for PCR reactions were designed using Benchling [235] and the NEB melting temperature calculator [236]. PCR reactions were carried out using the Q5 proofreading polymerase (NEB) according to the manufacturer's protocol. Reactions were run in 50 µl volume, using the manufacturer's recommended buffer, with 500 nM forward and reverse primers, 200 µM dNTPs, and 1-25ng of template DNA. Optionally, 1M betaine and 3-5% DMSO were used as additives for difficult reactions. Standard cycling parameters were used as follows: 98°C for 30s initial denaturation, followed by 35 cycles of: 98 °C 10s denaturation, 30s annealing, and 72°C extension time, ending with a 72°C final extension for 2 minutes. Annealing temperature is primer-dependent, and an extension time of 30 seconds per 1000bp of template was used. PCR products were analysed using agarose gel electrophoresis (see Section 0) and purified using a Qiaquick PCR Purification Kit (Qiagen) following the manufacturer's instructions.

2.3.2.2 Site-Directed Mutagenesis

For site-directed mutagenesis of purified plasmid samples, primer design and PCR was carried out as in Section 2.3.2.1. Mutagenesis reaction mixtures (KLD mixes) contained kinase, ligase, and DpnI as described in Table 2.3.

Table 2.3: **Reaction Mixture Components for Site Directed Mutagenesis**
Mixtures assembled at room temperature. Purity and presence of the desired dsDNA product from PCR products was established by agarose gel electrophoresis prior to running mutagenesis reactions. ddH2O = double-distilled milliQ pure water.

Component	Volume
PCR Product	1 µl
T4 Polynucleotide Kinase	1 µl
T4 DNA Ligase	1 µl
DpnI	1 µl
T4 Ligase Buffer	1 µl
ddH$_2$O	5 µl

Mixtures were incubated at room temperature for 1 hour and transformed into *E. coli* DH5α (see section 2.3.1.2) for preparation of pure plasmid DNA (see section 2.3.2.5).

2.3.2.3 Gibson Assembly

Gibson assembly reactions were performed using a homemade reaction mix. The buffer and master mix recipe are shown in Table 2.4.

Table 2.4: **Buffer and Master Mix Components for Gibson Assembly**
All mixtures were made in distilled water, pH adjusted with 6M HCl and stored at 4 ˚C. 5X Isothermal Reaction Buffer was made up to volume of 6ml and stored in 500 µl aliquots, and Master Mix stored in 50 µl aliquots. ddH2O = double-distilled milliQ pure water.

Buffer Name	Components
5X Isothermal Reaction Buffer	0.5 M Tris-HCl pH 7.5 50 mM $MgCl_2$ 1 mM dNTP Mix 50 mM DTT 25 % w/v PEG-8000 5 mM NAD
Master Mix	320 µl 5X Isothermal Reaction Buffer 0.64 µl T5 Exonuclease 20 µl Phusion DNA Polymerase 160 µl Taq Ligase 700 µl ddH_2O

DNA fragments containing homologous 20 bp overhangs were obtained by gene synthesis (IDT) or purified from PCR. Fragments were added to 15 µl of Master Mix. In the case of vector assembly, fragments were added in a 3:1 molar ratio of insert to vector backbone (corresponding to 60 fmol insert and 20 fmol vector), otherwise fragments were added in an equimolar ratio (50 fmol each). Reactions were made up to 20 µl volume with distilled water and incubated at 50 °C for 60 minutes in a thermocycler. 15 µl of the reaction mixture was transformed into *E. coli* DH5α (see section 2.3.1.2) for preparation of pure plasmid DNA (see section 2.3.2.5).

2.3.2.4 Golden Gate Assembly

DNA fragments were assembled with Golden Gate assembly using Type-IIS restriction enzymes. Reaction mixture components are shown in Table 2.5.

Table 2.5: **Golden Gate Reaction Mixture Components**
Mixtures are assembled on ice and adjusted to 20 µl final volume with distilled water. ddH2O = double-distilled milliQ pure water.

Component	Amount
BsaI	1 µl
T4 Ligase	1 µl
10X T4 Ligase Buffer	2 µl
Vector DNA	20 fmol
Insert DNA	60 fmol
ddH2O	To 20 µl

Purified plasmid DNA containing the necessary BsaI sites was obtained by miniprep (see section 2.3.2.5). Insert DNA fragments with matching BsaI restriction sites were obtained by gene synthesis (IDT) or from PCR purification. Reactions were incubated in a thermocycler with the following program: 30x cycles of 37 °C for 5 minutes and 16 °C for 5 minutes, followed by a final heat denaturation step of 65 °C for 20 minutes. 15 µl of the reaction mixture was transformed into *E. coli* DH5α (see section 2.3.1.2) for preparation of pure plasmid DNA (see section 2.3.2.5).

2.3.2.5 Plasmid DNA Preparation

6 ml of LB broth (including antibiotics) in a 30 ml Sterilin tube was inoculated with a colony from a freshly grown agar plate of E. coli DH5α transformants. Cells were harvested by centrifugation at 4000xg for 15 minutes at 4 °C, supernatant discarded, and cell pellets stored at -20 °C. Plasmid DNA was purified from cell pellets using a Monarch® Plasmid Miniprep Kit (NEB) following the manufacturer's instructions. Pure plasmid DNA was eluted in 35 µl ddH2O and verified using restriction digest (section 2.3.2.6) and Sanger sequencing (section 2.3.2.8).

2.3.2.6 Diagnostic Restriction Digests

Purified, assembled plasmids were verified by restriction digest. Reaction mixture components are shown in Table 2.6.

Table 2.6: **Restriction Digest Reaction Components**
Mixtures are adjusted to 20 µl final volume with distilled water. ddH2O = double-distilled milliQ pure water.

Component	Amount
Restriction Enzyme(s)	1 µl each
Reaction Buffer (NEB CutSmart/Buffer 2.1/3.1)	2 µl
Plasmid DNA	500 ng
ddH2O	To 20 µl

Reactions were incubated at 37 °C for 60 minutes and visualised with agarose gel electrophoresis (see section 2.3.2.7)

2.3.2.7 Agarose Gel Electrophoresis

1% agarose gels were prepared by adding 0.7g agarose to 70 ml Tris borate EDTA (TBE) buffer – 0.13 M Tris pH 7.6, 45 mM boric acid, 2.5 mM EDTA. The mixture was microwaved at 50% power for approximately 2 minutes until all the agarose was dissolved, and then incubated in a 37 °C water bath to cool. Once cool enough to handle, 7 µl of 10,000X GelRed® Nucleic Acid Gel Stain (Thermo Fisher) was added to the agarose solution and the mixture poured into a mould to solidify. Nucleic acid samples were supplemented with 1X OrangeG loading dye

(Melford) from a 10X stock before loading onto agarose gels. Gels were run at 120 V for 60 minutes.

2.3.2.8 Sanger Sequencing

Purified plasmid DNA samples were sent for Sanger sequencing using the TubeSeq Supreme service (Eurofins Genomics). Samples were prepared according to the service provider's guidelines and sequenced using the standard T7 forward primer (sequence TAATACGACTCACTATAGGG) and a custom reverse primer (GAGAGCGTTCACCGA) covering the sequence region between the *T7* promoter and *rrnB T1* terminator.

2.3.3 Protein Expression

2.3.3.1 Small-Scale Protein Expression

6 ml of LB broth (including antibiotics and 2% w/v glucose) in a 30 ml Sterilin tube was inoculated with a colony from a freshly grown agar plate of E. coli transformants. This culture was incubated at 37 °C with 220 rpm shaking until reaching exponential phase, as measured by optical density at 600 nm reaching 0.6-0.9. The culture was then induced with 1 mM IPTG (from a filter sterilised 1M stock in distilled water). Induced cultures were grown at 18 °C or 25 °C overnight, or 37 °C for 3 hours, with 220 rpm shaking. Cells were harvested by centrifugation at 4000xg for 15 minutes at 4 °C, supernatant discarded, and cell pellets stored at -20 °C.

2.3.3.2 Large-Scale Protein Expression

8 ml of LB broth (including antibiotics and 2% w/v glucose) was added to 2x30 ml Sterilin tubes. Each tube was inoculated with a colony picked from a freshly grown agar plate of *E. coli* transformants and resuspended in distilled water (each tube was inoculated with the same colony). Cultures were incubated at 37 °C with 220 rpm shaking overnight. These overnight cultures were used to inoculate 4x2.5 l flasks each containing 400 ml TB broth (including antibiotics). Each flask was inoculated with 4 ml of overnight culture and incubated at 37 °C, 220 rpm shaking. Flasks were incubated until reaching exponential phase, as measured by optical density at 600 nm reaching 0.6-0.9. Cultures were then induced with 1 mM IPTG (from a filter sterilised 1M stock in distilled water). Induced cultures were grown at 25 °C with 220 rpm shaking overnight. Cell pellets were harvested and stored as described in Section *2.3.3.1*.

2.3.3.3 Cell Lysate Preparation

E. coli cell pellets were resuspended to a fixed final OD, calculated using the equation:

$$Final\ volume = Culture\ OD_{600} \times \frac{Culture\ volume\ (ml)}{Desired\ OD_{600}}$$

Small-scale cell resuspensions were prepared at a final OD of 10, and large-scale cell resuspensions were prepared to final OD of 80. Large-scale cell resuspensions were also supplemented with 1 EDTA-free protease inhibitor tablet (Roche) per 30 ml. Resuspended cells were lysed by sonication at 100% amplitude and 10 second on/off pulses for 5 minutes. The lysate was centrifuged at 16,000xg for 30 minutes at 4 °C and the supernatant decanted. Large-scale samples were filtered using a 0.45 µm filter (Sartorius) before chromatography.

2.3.4 Protein Purification

All samples were kept on ice or stored at 4 °C during preparation. Chromatography was carried out using ÄKTA FPLC systems with the UNICORN 5.31 software, or on the benchtop with gravity flow columns. All columns were supplied by GE Healthcare. Buffers used for purification are outlined in Table 2.7.

Table 2.7: **Buffers Used for Protein Purification**
All buffers were made in distilled water, pH adjusted with 6M HCl, filtered using a 0.22 µm filter (Sartorius), and stored at 4 ˚C.

Buffer Name	Components
Lysis Buffer	50 mM Tris pH 8.0 150 mM NaCl
Elution Buffer	50 mM Tris pH 8.0 150 mM NaCl 50 mM Biotin

2.3.4.1 Affinity Chromatography

Cell lysates were loaded onto Streptactin XT columns equilibrated in Lysis Buffer. In FPLC, the column was washed with further Lysis Buffer until the UV reading reached a stable baseline (≈3-4 column volumes). Elution was performed over 12 column volumes with Elution Buffer and 1ml fractions collected. In gravity flow, columns were washed with 5 column volumes lysis buffer, and then fractions eluted over 3 column volumes of Elution Buffer. Eluted fractions were analysed by SDS-PAGE, and all fractions containing target protein were pooled and concentrated using a Vivaspin-20 10 kDa MWCO concentrator (Sartorius).

2.3.4.2 Size Exclusion Chromatography

Pooled and concentrated fractions from affinity chromatography were injected onto gel filtration columns equilibrated in Lysis Buffer. Columns used were either a Superdex 200 Increase 10/300 GL or a Superose 6 Increase 10/300 GL. Columns were washed at 1 ml/min with Lysis Buffer. As previously, eluted fractions corresponding to the peak UV absorption were analysed by SDS-PAGE, and fractions containing the target protein were pooled and concentrated.

2.3.5 Protein Analysis and Quantification

2.3.5.1 Sodium Dodecyl Sulphate Polyacrylamide Gel Electrophoresis (SDS-PAGE)

Protein samples were denatured at 95 °C for 10 minutes in 1x Bolt™ lithium dodecyl sulphate buffer with Bolt™ reducing agent (Invitrogen). Samples were briefly centrifuged at >13,000xg for a few seconds and mixed by pipetting, before 16 µl was loaded onto Bolt™ 4-12% Bis-Tris Plus gels (Invitrogen). Gels were run at 200V for 23 minutes in MOPS-SDS running buffer. Following electrophoresis, gels were stained with ~10 ml of InstantBlue™ stain (Expedeon) and gently rocked at room temperature for 30 minutes. Stain was washed off and replaced with water before inspection.

2.3.5.2 Native Polyacrylamide Gel Electrophoresis (Native-PAGE)

Protein samples were diluted to 1 µg total protein content in Lysis Buffer and 1x NativePAGE™ sample buffer (Invitrogen). 8 µl of sample was loaded onto NativePAGE™ 3-12% Bis-Tris gels (Invitrogen). Gels were run at 150V for 2 hours, with NativePAGE™ 1X running buffer for the anode chamber and 1X running buffer plus Cathode Buffer Additive for the cathode chamber (Invitrogen). Following electrophoresis, gels were stained with ~10 ml of InstantBlue™ stain (Expedeon) and gently rocked at room temperature for 30 minutes. Stain was washed off and replaced with water before inspection.

2.3.5.3 Western Blot

Solutions used in western blots are outlined in Table 2.8.

Table 2.8: Solutions used in Western Blotting

Solutions were freshly made in distilled water and used immediately.

Solution Name	Components
Blocking Solution	1x Phosphate Buffered Saline (PBS) 0.05 % v/v TWEEN-20 3% w/v Bovine Serum Albumin (BSA)
Wash Solution	1x PBS 0.05 % v/v TWEEN-20
Antibody Solution	1x PBS 0.05 % v/v TWEEN-20 1% w/v skim milk powder 1:10,000 Streptactin/Horseradish Peroxidase Conjugate

Western blots were performed using the iBlot2 dry blotting system (Invitrogen). SDS-PAGE gels were run as previously described but covered with distilled water instead of staining. Gels were then placed inside the blotting cassette and inserted into the iBlot2 transfer device. The transfer scheme (the setting "mixed mW") consisted of 20 V for 1 minute, 23 V for 4 minutes, and finally 25 V for 2 minutes. Following blotting the membrane was removed, and the following incubation steps carried out at room temperature with gentle rocking:

1. 35 ml Blocking Solution for 1 hour
2. Blocking solution was removed and 50 ml Wash Solution added for 5 minutes – this step is repeated three times.
3. 20 ml Antibody Solution for 90 minutes
4. 3x 5-minute washing steps as in step 2.

Following these incubations, the blot was developed by addition of two SIGMAFAST 3,3-Diaminobenzidine (DAB) tablets dissolved in 5 ml of distilled water. This solution was washed off with 50 ml distilled water after 5 minutes.

2.3.5.4 Protein Concentration Measurements
Protein concentration was measured using absorbance at 280 nm. Measurements were taken using a Denovix DS-11 Spectrophotometer, with path length automatically adjusted to 1 cm equivalent. Concentration was calculated using the formula:

$$[Protein] = \frac{A280}{\epsilon} \times mW$$

Where:

[Protein] is measured in mg/ml

A280 is the absorbance at 280 nm as measured with a 1 cm path length equivalent

ε is the extinction coefficient of Eh as estimated by ProtParam [237]

mW is the molecular mass of the protein in g/mol

2.3.5.5 Dynamic Light Scattering (DLS)
Hydrodynamic particle diameters of purified protein samples were measured using dynamic light scattering (DLS) using a Zetasizer Nano ZS (Malvern). Measurements were performed at 0.1 mg/ml protein concentration in 50 mM Tris pH 8.0, 150 mM NaCl, at 25 ˚C. Three repeat measurements were made for each sample with backscatter detection. Measurements were made using the default "Size" protocol in the Measurement Builder wizard in the ZS Xplorer software (Malvern). Instrument parameters were optimised automatically.

2.3.5.6 Transmission Electron Microscopy (TEM)
TEM of negatively stained proteins was performed at the Birkbeck EM and Image Processing Lab by Dr Shu Chen. 3 µl of purified protein (0.1-0.5 mg/ml concentration) in Lysis Buffer was loaded onto 6 mm glow-discharged carbon grids. After 30 seconds, 3 drops of filtered uranyl acetate stain were added. After another 30 seconds, grids were blotted with filter paper and allowed to dry for at least 3 minutes before imaging. Micrographs were taken at 26000x or 52000x magnification on a Tecnai T12 120 keV microscope with a Gatan Ultrascan 4000 camera.

2.4 High-Throughput Methods for Encapsulin Characterisation and Screening

This section details modifications made to the methods in Section 2.3 to adapt them to a high-throughput setting, to allow scalable experimental screening of encapsulin designs. Automated liquid handling protocols were performed using an Opentrons OT-2 liquid handler. All liquid handling protocols and custom labware definitions are publicly available at https://github.com/naailkhan28/opentrons_protocols .

2.4.1 *E. coli* Transformation

The previously described transformation protocol was adapted for use with the Opentrons OT-2 liquid handler. 10 µl of competent *E. coli* cells was aliquoted into each well of a 200 µl 96-well PCR plate (Starlab). The plate was transferred to the OT-2 Temperature Module which was always kept at 4 °C. Cells were incubated for 5 minutes before 15 µl plasmid DNA (diluted to ≈10 ng/µl) or DNA assembly reaction mix was added, and cells incubated for a further 30 minutes at 4 °C. The plate was removed and subjected to a 60 second heat shock at 42 °C in a water bath, before being returned to the Temperature Module for 5 minutes. 125 µl of SOC medium was added to each well, and the plates sealed with breathable plate seals (Sigma) and transferred to a Thermomixer (Thermo Fisher) incubator at 37 °C 750 rpm shaking for 1 hour. Following outgrowth, the plate was returned to the OT-2 where 10 µl of each outgrowth culture was spotted onto a rectangular agar plate (Grenier) in a 96-well pattern. Lastly, 70 µl of each culture was removed and replaced with 70 µl of fresh SOC medium to dilute cultures to double their starting volume. 10 µl of diluted culture was spotted onto the agar plate.

2.4.2 Protein Expression

Colonies from a freshly grown agar plate were picked using a PIXL colony picking robot (Singer Instruments) in the "background subtraction" detection mode, or manually. Each colony was used to inoculate 250 µl of LB medium with antibiotics in a 2.2ml deep 96-well plate (Thermo Fisher). Plates were sealed using breathable seals as previously and incubated in a Thermomixer overnight at 37 °C with 1200 rpm shaking.

The following day, 25 µl of overnight culture was added to 225 µl of TB medium in a fresh deep 96-well plate and plates covered with breathable seals. Expression cultures were grown in a Thermomixer at 37 ˚C, 1200 rpm shaking for 3 hours before induction of protein expression by the addition of 1 mM IPTG as previously described. The temperature was lowered to 25 ˚C and induced cultures grown for 24 hours. Finally, cell pellets were harvested and stored as previously described.

2.4.3 Protein Solubility Screening

Cell pellets in deep-well 96-well plates were resuspended in 25 µl BugBuster protein extraction reagent (Sigma) supplemented with 25 U/ml benzonase. Resuspended cell pellets were incubated at room temperature with 1200 rpm shaking for 30 minutes and clarified by filtration. Lysates were transferred to a 0.22 µm MultiScreen 96-well filter plate (Merck) and centrifuged at 4000xg for 15 minutes at 4 °C. 5 µl of the clarified lysate was applied to a nitrocellulose membrane (Bio-Rad) for dot blotting and the membrane was blocked, washed, incubated with antibody, and detected in the same way as the previously described Western Blots (Section 2.3.5.3). Dot blot images were processed in ImageJ.

Chapter 3: Discovery of Novel Encapsulin Candidates from Metagenomic Databases

Published works statement: Data and figures in this chapter are reproduced from: Naail Kashif-Khan, Renos Savva, Stefanie Frank, Mining metagenomics data for novel bacterial nanocompartments, *NAR Genomics and Bioinformatics*, Volume 6, Issue 1, March 2024, lqae025, **https://doi.org/10.1093/nargab/lqae025** [238] as described in the UCL Research Paper Declaration Form.

3.1 Background

A recent bioinformatics study described a dataset of over 6000 encapsulin sequences [26], however this work deliberately excluded metagenomics data from the search. As described in Section 1.5.3, metagenomics databases contain diverse protein sequences which share little or no identity with known proteins from sequenced, assembled genomes. However no previous work has attempted to search for encapsulins in these databases. In this chapter, it is hypothesised that metagenomics databases are a potential source of novel encapsulin proteins, showing low sequence identity with protein sequences in genomic databases. These novel encapsulins may have interesting new properties, including novel structural features, previously unseen cargo proteins or biological functions, and potentially new assembly architectures beyond the T=1, 3, or 4 encapsulin systems which are currently known.

3.2 Search Pipeline

This work focuses on the MGnify Protein Database, a publicly available metagenomic protein database. This resource contains over 2 billion protein sequences, aggregated from metagenomics studies deposited in MGnify, a universal resource for metagenomics data hosted by the European Bioinformatics Institute. Predicted structures for a large subset of MGnify proteins are also available in the ESM Atlas. An overview of the search pipeline is presented in Figure 3.2.1 (panels a and b).

The search strategy used here is comparable to previous work, with two key differences. The MGnify Protein Database was searched using Pfam functional annotations as in [26], however all 17 Pfam families from the clan CL0373 (“phage coat”) were used as queries. This differs from previous work where only two Pfam families were used as queries, namely

PF04454 (“encapsulating protein for peroxidase”) and PF05065 (“phage capsid family”). Pfam clan CL0373 contains annotations for all HK97-fold proteins, including both phage capsids and encapsulins. Using all Pfam labels in this clan is expected to discover more distantly related encapsulin homologs, or sequences which have been misannotated as phage capsids. However, this strategy is also expected to return many phage capsid proteins, which must be filtered out afterwards. In parallel, the ESM Atlas was searched using a set of encapsulin structures as query, which has not been described in previous work. Protein structure search can recover more distant sequence homologs when compared to sequence-based searches [152], making it well suited for interrogating metagenomic databases like MGnify. However, since encapsulins and phage capsids share the HK97 fold, this search also returned contaminating phage capsid proteins [238].

Pfam family and structure searches returned ≈760,000 and ≈2800 sequence hits respectively (Figure 3.2.1c), however most of these hits were likely phage capsid proteins. To remove viral sequences from the dataset, the genomic context of each candidate encapsulin gene was examined. Metagenomic sequence data consists of contiguous nucleotide sequence regions known as “contigs”. These contigs vary in length and the number of protein-coding genes they contain. Several different database queries against both MGnify and the European Nucleotide Archive (ENA) were required to obtain the contig sequences for each encapsulin candidate (Figure 3.2.1b). Contig accessions were obtained for 83% of initial encapsulin candidates, but protein coding sequences were only available for 53% of these contigs. These protein coding sequences surrounding the putative encapsulins were further investigated to remove potential phage capsid proteins.

Several filtering steps were required to remove potential phage capsid proteins from the dataset. The protein coding genes from each contig were searched against a sequence database containing two phage proteomes, to identify any phage-associated proteins near the candidate encapsulin-coding gene, as in [26]. Any candidates found in contigs containing a hit against a phage protein sequence were removed. Next, candidate proteins associated with a contig under 25 kb in length or containing fewer than 10 protein-coding genes were removed from the dataset. This is to ensure that every putative sequence has a minimum level of genomic context available, to rule out the possibility of that sequence being of viral origin, but also to aid in subsequent functional annotation.

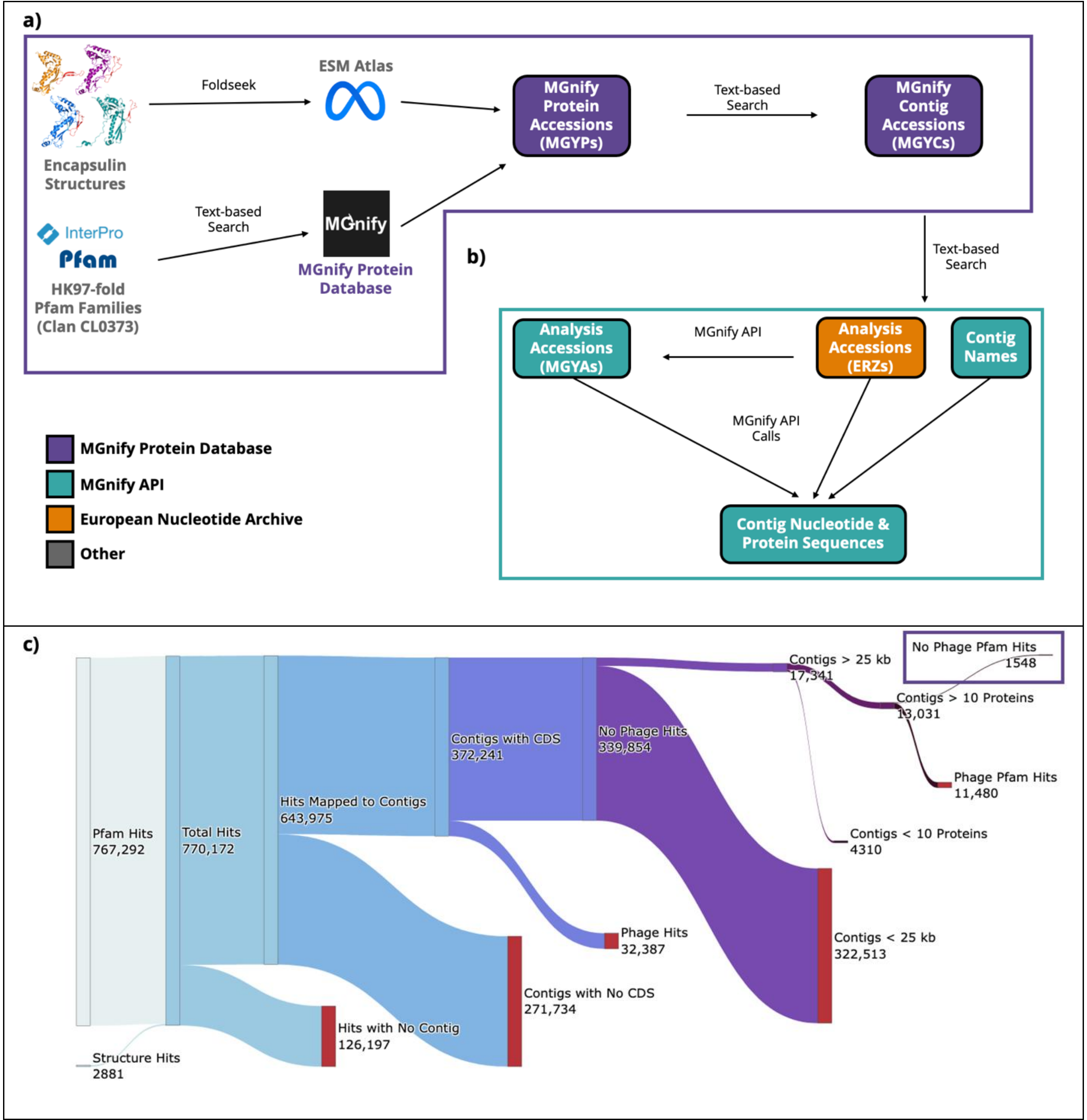


Figure 3.2.1: **Data Search and Retrieval Methodology**
a) Input structures and Pfam families were searched against the ESM Atlas and MGnify Protein Database respectively. The resulting MGYP protein accessions were mapped to MGYC contig accessions from the MGnify Protein Database. **b)** MGYC contig accessions were mapped to ERZ analysis accessions, and MGnify contig names using simple text-based filtering. ERZ accessions were mapped to MGYA accessions using the MGnify API. Repeated API calls were made to retrieve contig sequences from MGYA and ERZ accessions, and contig names. Accessions and data are coloured according to source, with MGnify Protein Database shown in purple, MGnify studies and API in teal, European Nucleotide Archive in orange, and all other data sources in grey. **c)** Sankey diagram showing a summary of the number of putative encapsulin sequences at each step of the filtering and curation pipeline. Red nodes indicate sequences which were removed from the analysis. The purple box in the top right corner indicates the final dataset of putative encapsulin sequences after all curation steps. Figure reproduced from [238].

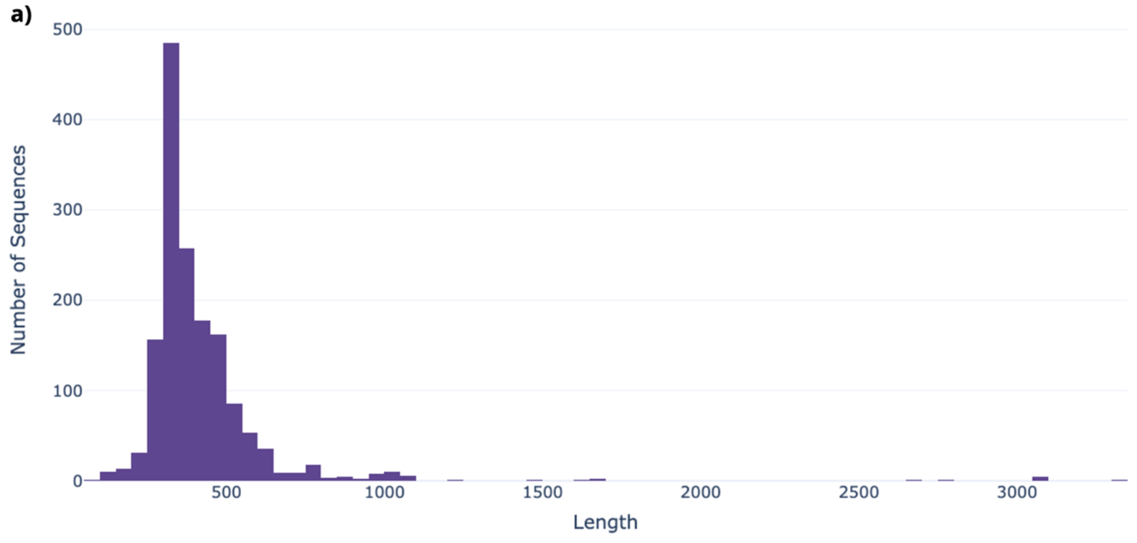
Lastly, the Pfam annotations of all proteins in each contig were examined and compared against a curated set of phage-associated Pfam families. Any candidates associated with contigs containing a putative phage-associated gene were removed from the dataset. Thus, from ≈372,000 encapsulin candidates with available contig proteins, a final curated dataset of 1548 filtered encapsulin proteins was produced [238].

3.3 Length and Biome Distribution of Metagenomic Encapsulin Candidates

As shown in Figure 3.3.1a, the vast majority of encapsulin candidate sequences discovered in this work were under 500 residues in length. This is expected given that all experimentally characterised encapsulins are between 2-300 residues long. The metagenomic candidate sequences were searched against UniRef90 to find the sequence identity with closest hits in a conventional, genomic protein database (Figure 3.3.1b). Whilst 225 candidate sequences shared 95% identity or higher with a UniRef90 sequence, 77 sequences returned no hits at all. The rest of the candidate sequences showed sequence identities ranging from 20-90% against their nearest hit in UniRef90. These data demonstrate that the metagenomic candidate sequences presented here explore a wide range of sequence space, and that some metagenomic sequences do indeed show low identity against sequences in conventional databases [238].

Since the candidate sequences originate from metagenomic contigs, and not fully assembled genomes, tracing the species of origin for these 1548 sequences is non-trivial. However, each of these sequences is associated with a metagenomic study in MGnify, with corresponding metadata on sample collection. These samples are annotated with biome information according to the Environment Ontology, as shown in Figure 3.3.2. Putative encapsulins were found in 2000 different metagenomics samples across diverse environments. Whilst most samples are sourced from aquatic environments, a sizeable proportion (29%) are associated with host-associated biomes, most of which are microbiota of the digestive systems of humans, other mammals, or birds. The presence of encapsulins in host-associated pathogens aligns with previous results [26] and may support the hypothesis that these proteins serve roles in bacterial pathogenicity [239].

a)



b)

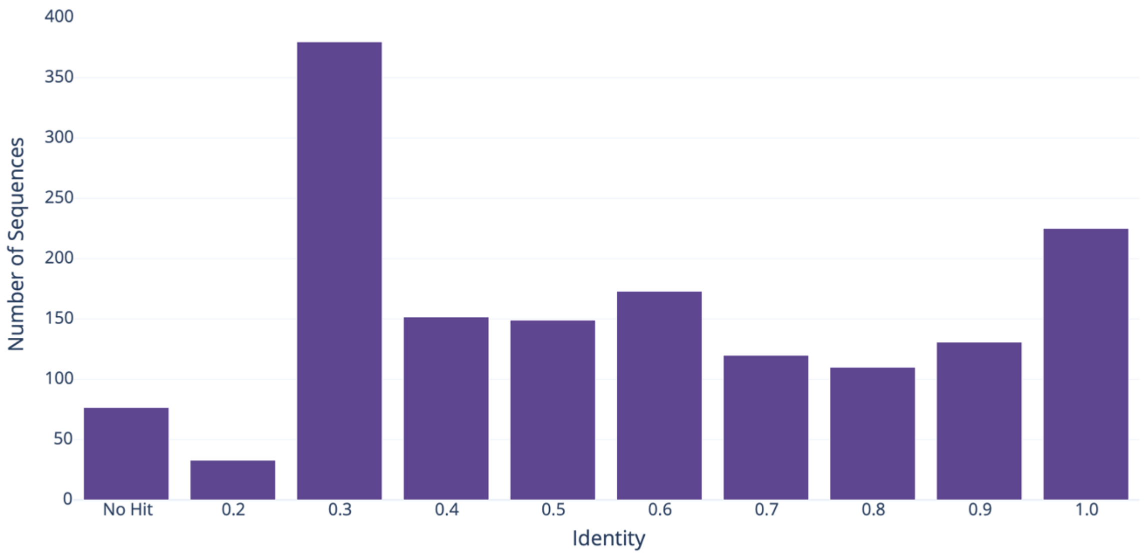


Figure 3.3.1: **Length and Identity Distribution of Encapsulin Candidate Sequences**
Histograms depicting the distribution of **a)** length and **b)** sequence identity of the closest hit in UniRef90 for 1548 encapsulin candidate sequences. X-axis values denote the centre of each bin. 77 sequences had no hits in UniRef90, and 225 sequences showed over 95% identity to a sequence in UniRef90. Most candidate sequences are under 500 residues, with only 38 sequences over 900 residues. Figure reproduced from [238].

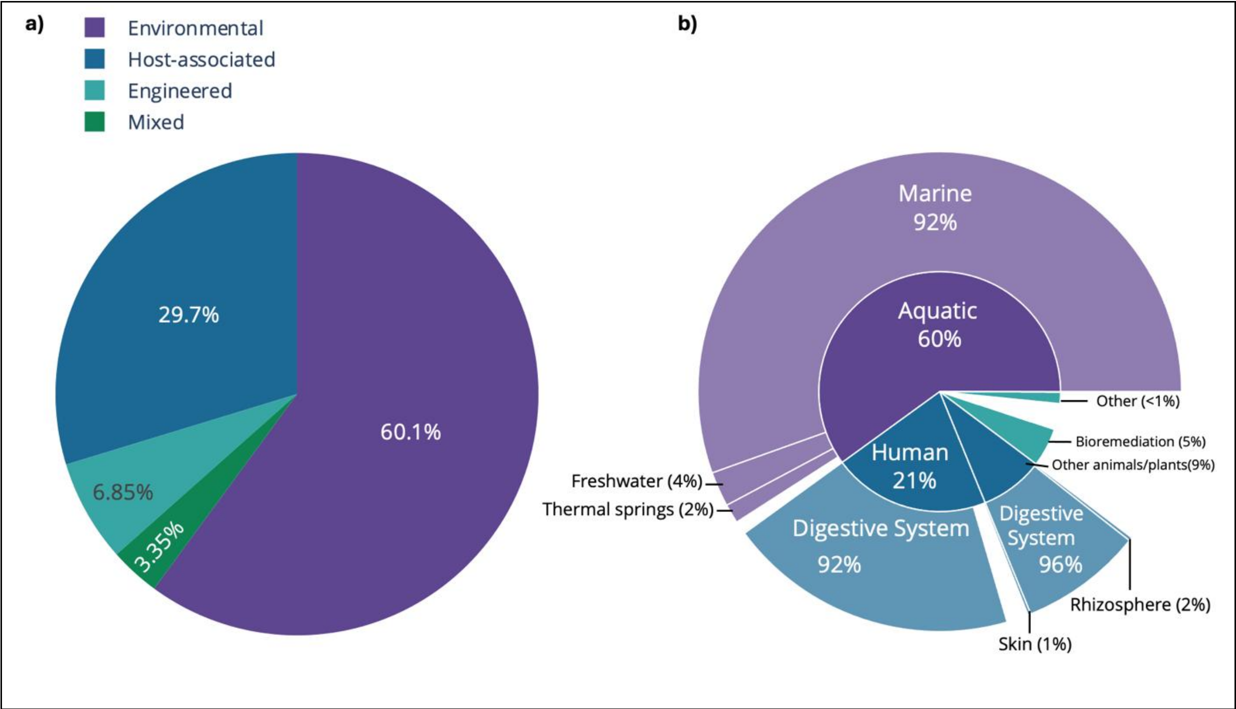


Figure 3.3.2: **Biome Distribution of Metagenomic Encapsulin Candidates**
a) Breakdown of biome data for the 2000 metagenomic samples where putative encapsulin sequences were found. The four categories in the pie chart represent the four top-level categories of the Environment Ontology used in the MGnify database – environmental, host-associated, engineered, and mixed. **b)** Sunburst plot showing a breakdown of the biomes within each category from b). “Other animals” includes mammals and birds. “Other” engineered types include laboratory samples, food production, fermented beverage production, and bioreactors/biogas sites. These categories contain many sparsely populated subcategories which are omitted for clarity. The “Mixed” biome has no subcategories and is thus omitted. Figure reproduced from [238].

3.4 Metagenomic Encapsulin-Associated Cargo Proteins

Several methods were used to annotate metagenomic encapsulin candidates with a cargo protein, to try to establish putative biological function for these encapsulin systems. Pfam annotations for the surrounding proteins from each contig were compared against a curated list of encapsulin cargo Pfam families. Hidden Markov models (HMMs) were constructed for each of the family 2 cargo proteins and searched against contig proteins to identify hits. Contig proteins were searched against the BLAST non-redundant database to potentially identify cargo proteins with similarity to those in genomic databases. Lastly, representative cargo loading peptide (CLP) sequences were searched against these contig proteins, to identify cargo proteins with no annotated function, but containing a potential CLP region targeting proteins for encapsulation [238]. However, even with this involved search strategy, only 177 out of 1548 metagenomic encapsulin candidates were annotated with an associated cargo protein (Figure 3.4.1).

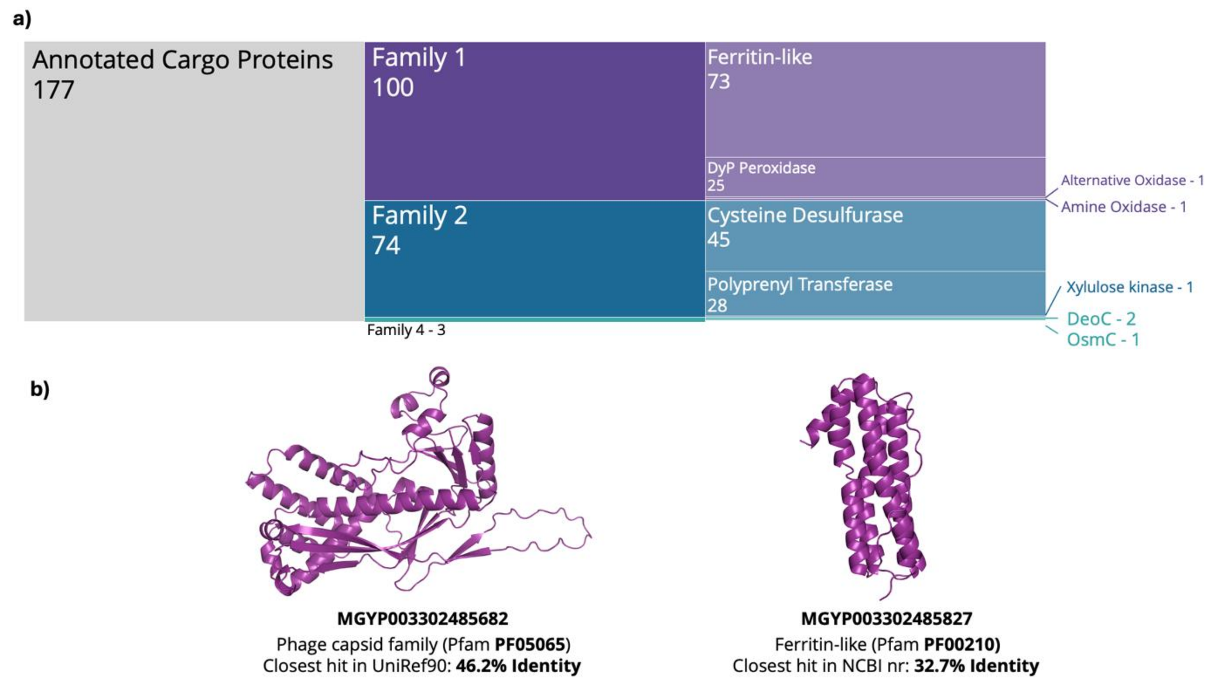


***Figure 3.4.1*: Metagenomic Encapsulin Candidates and Putative Cargo Proteins**
a) Icicle plot showing a breakdown of annotated cargo proteins for metagenomic encapsulin candidates, separated by family. Out of 1548 putative encapsulin hits, 177 were annotated with putative cargo proteins. Most cargo proteins were from families 1 and 2, with 3 proteins from family 4. **b)** ESMFold predicted structures of a putative encapsulin hit and its respective ferritin-like cargo protein. Despite low sequence identity to any sequence in conventional databases, both are identified as encapsulin and ferritin-like proteins by functional annotation and predicted structures respectively. Figure reproduced from [238].

This shows the diverse nature of metagenomic proteins and the well-known difficulties in assigning them biological functions.

Most metagenomic cargo candidates belonged to encapsulin families 1 and 2 and included DyP-type peroxidases, ferritin-like domains, cysteine desulfurases, and polyprenyl transferases. Figure 3.4.1b shows the predicted structure of a representative putative encapsulin and its candidate corresponding ferritin-like cargo protein. Both show low sequence identity to any protein in conventional databases (46.2% and 32.7% for encapsulin and cargo) but are annotated as encapsulin and cargo by Pfam family. Predicted structures of these two proteins also corroborate their sequence-based functional annotations – the encapsulin clearly displays the HK97 fold while the cargo protein gives significant hits against crystal structures of ferritins from *E. coli* when searched against the PDB using Foldseek. Interestingly, this encapsulin was annotated with Pfam PF05065 ("phage capsid family”), a label which was previously assigned to family 3 encapsulins [26] but which is seen here in a putative family 1 encapsulin system.

3.5 Metagenomic Encapsulin-Associated Biosynthetic Gene Clusters

Family 3 encapsulins are found within putative BGCs (see Section 1.4.1). In order to find potential family 3 encapsulins in this metagenomic dataset, each encapsulin-associated contig was run through two BGC prediction tools, DeepBGC [71] and antiSMASH [70]. The former uses a deep learning model to predict the presence of BGCs based on the co-occurrence of Pfam families, while the latter is a more traditional bioinformatics tool which uses manually defined rules to define BGCs based on Pfam families and other HMM searches. Contig nucleotide sequences were used as inputs to both tools. Whilst DeepBGC outputs a simple tabular data file which can be processed in Python, antiSMASH generates a folder of HTML files for each input contig, intended for manual inspection in a web browser but not suitable for automatic processing and analysis. As such, a script was written to automatically scrape the data from these HTML files using the Python package beautifulsoup4 [223].

BGC predictions uncovered a potentially novel encapsulin-associated BGC, the Saccharide BGC (Figure 3.5.1a). These putative BGCs are predicted by DeepBGC to produce antimicrobial or cytotoxic saccharides, and all encode at least one glycosyl transferase enzyme, although most contain multiple such enzymes.

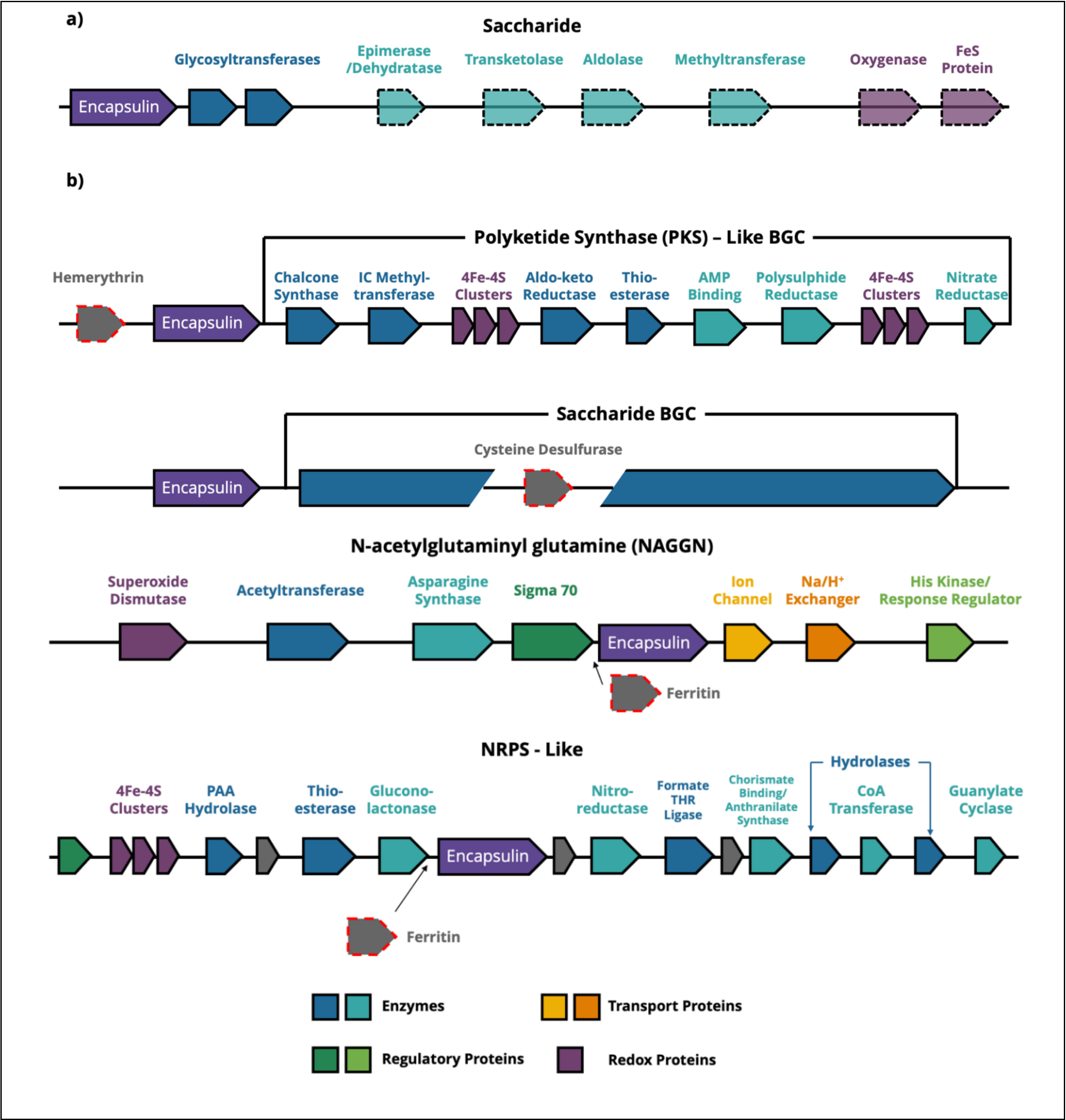


Figure 3.5.1: **Putative Metagenomic Encapsulin-Associated BGCs**
a) 29 putative encapsulins are found in predicted saccharide BGCs, all containing at least one glycosyl transferase. These BGCs may also encode epimerases, aldolases, oxygenases, and redox proteins. b) Some putative encapsulins are found in putative BGCs, and near known cargo proteins with loading peptides or domains (grey arrows with red dashed outline). In one example hemerythrin, a family 1 cargo protein, is seen upstream of the putative encapsulin and a polyketide synthase (PKS)-like BGC. Other encapsulin candidates are found upstream of saccharide BGCs containing cysteine desulfurase, a family 2 cargo. The N-acetylglutaminyl glutamine amide (NAGGN) BGC contains asparagine synthase and acetyltransferase enzymes, whilst non-ribosomal peptide synthetase (NRPS)-like clusters encode phosphate/AMP binding proteins. Both contain putative encapsulins with ferritin cargos. Enzymes not found in all operons are shown in dashed outline. Direction of arrows does not indicate gene orientation and is for schematic purposes only.
PAA = Phenylacetic acid, IC = Isoprenylcysteine. Figure reproduced from [238].

Other enzymes that can be found in such BGCs include carbohydrate epimerases and dehydratases, methyltransferases, and oxygenases. Proteins from several Saccharide BGC systems were used as query for BLAST searches or Foldseek searches against the PDB using ESMFold predictions. However, none of these searches gave any significant matches (E-Value < 10-3 for BLAST, or TM-Score and Probability > 0.5 for Foldseek) to proteins of known structure or function from these databases, indicating that the Saccharide BGC query proteins show a low degree of homology to proteins in these databases.

Several putative encapsulins were found within predicted BGCs with known cargo proteins that contain capsid targeting peptides or domains (Figure 3.5.1b). Several putative Saccharide BGCs also contains cysteine desulfurases, a known family 2 cargo. A putative encapsulin was found downstream of a putative hemerythrin cargo, but upstream of a polyketide synthase-like BGC, which encodes enzymes involved in the synthesis of chalcones. Putative encapsulins with ferritin cargos were also found in N-acetylglutaminyl glutamine (NAGGN) and non-ribosomal peptide synthetase (NRPS)-like clusters, both of whose general function is to synthesise short modified peptides [78, 240].

The encapsulin-associated NAGGN BGC described here encodes the asparagine synthase and acetyltransferase enzymes needed to produce the osmoprotective peptide NAGGN [240]. NRPS-like encapsulin BGCs encode the phosphate/AMP binding proteins usually associated with NRPS BGCs, but are missing the key peptidyl carrier protein (PCP) which is required for non-ribosomal peptide synthesis [241]. However, such systems encode several potentially encapsulated enzymes, including gluconolactonases, thioesterases, aldo-keto reductases, and nitroreductases. Encapsulins have been previously reported as part of NRPS operons, but these partial “NRPS-like” systems lacking a full complement of enzymes have not been seen previously.

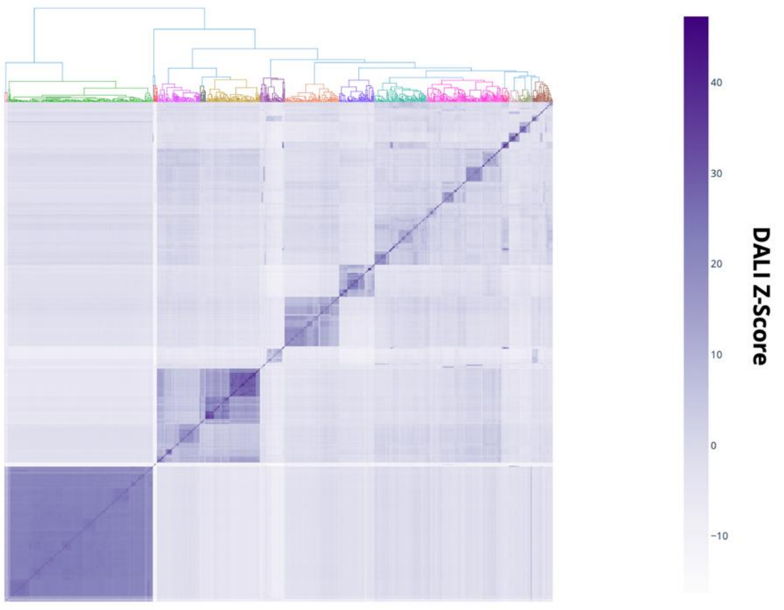
In total, both BGC prediction tools only returned 32 putative BGCs out of 1548 potential candidates. In addition, antiSMASH falsely predicted around 80 “RiPP-like” BGCs (short for ribosomally synthesised and post-translationally modified product). The putative encapsulin genes in these BGCs are assigned the Pfam family PF04454 whose full name is “Encapsulating protein for peroxidase”. However, antiSMASH incorrectly designates this Pfam family with the short name “Linocin:M18” [228]. Since linocin genes are usually found as part of real RiPP-like BGCs, antiSMASH falsely annotated these encapsulin-containing gene clusters as RiPP-like. This false annotation of encapsulin genes has been previously observed in the literature [242] and occurs in the Pfam database itself as well as in programs such as antiSMASH which make use of its functionality.

3.6 Structure Prediction of Metagenomic Encapsulin Candidates

Following cargo annotation and BGC prediction, predicted structures for the encapsulin candidates were inspected, to uncover potentially novel structural features in these diverse metagenomic proteins. Predicted structures for these encapsulin candidates were downloaded from the ESM Atlas where available, however in most cases a predicted structure was not available for download and so ESMFold predicted structures were generated manually. 38 sequences longer than 900 residues were not predicted or included in the analysis due to memory constraints (see length distribution in Figure 3.3.1). Any structures with mean predicted local distance difference test (pLDDT) values below 70 were also excluded from further analysis, to avoid artefacts from low confidence predictions.

Manual inspection of over 1000 predicted structures is impractical, so an automated investigation of these structures was carried out. Pairwise similarity between confident structures was computed in an all-against-all manner using DALI [218]. Four experimentally solved structures were also included in this matrix. As shown in Figure 3.6.1a, the similarity matrix showed clear patterns of clustered structures that share similarity with each other. The resulting similarity matrix was clustered using the scipy package in Python [219], and these clusters correspond to distinct regions of protein feature space (Figure 3.6.1b). Candidate sequences explore a wide range of lengths and charge properties, and in several cases, predicted encapsulin structures in the same cluster show similar length and isoelectric points. Notably, three out of the four experimental structures analysed here fall within the largest cluster of structures. This suggested that the other clusters may contain predicted structures with novel features not observed in these experimental structures.

a)



b)

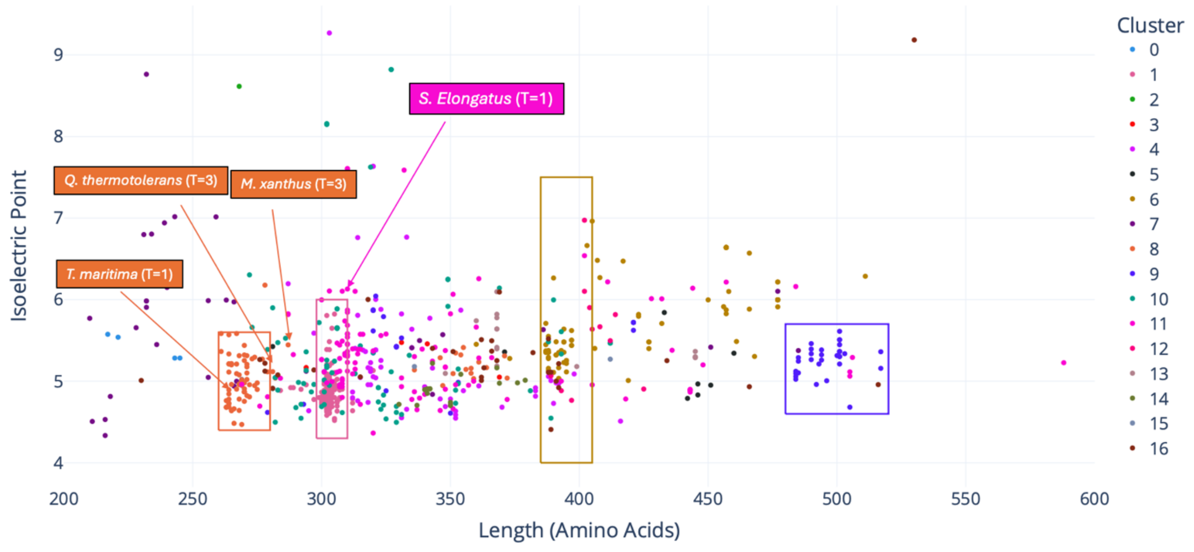


***Figure 3.6.1:* Clustering of Metagenomic Encapsulin Predicted Structures**
a) Clustered heatmap of DALI all-against-all pairwise similarity for high confidence encapsulin predicted structures. Clusters of structures sharing high similarity with each other can be seen visually and assigned using hierarchical clustering. These clusters of similar structures are shown in a coloured dendrogram at the top of the heatmap. **b)** Scatterplot of sequence length versus isoelectric point for predicted encapsulin structures, coloured by cluster. Whilst some clusters are relatively dispersed, there are some local regions where encapsulins of similar length and/or isoelectric point are clustered together (highlighted with coloured boxes). Experimentally solved encapsulin structures are shown, 3/4 of which fall within a single cluster. Figure reproduced from [238].

It is possible that the clustering results are based on artefacts from structure prediction using ESMFold as opposed to the more accurate AlphaFold2. As such, representative sequences from each cluster of ESMFold predicted structures were predicted using AlphaFold2 and the predictions from the two different methods were investigated. Across this set of representatives, the two methods showed good agreement as measured by TM-Score and comparison of pLDDT values (Figure 3.6.2a). In the one case where AlphaFold2 pLDDT is below 70, both structure predictions show good agreement in the HK97 fold and differ only in the conformation of an N terminal extension region (Figure 3.6.2b). TM-Scores between ESMFold and AF2 predictions are always at least 0.8, apart from two cases where, again, N terminal regions differ in conformation but the HK97 fold is preserved (Figure 3.6.2c and d). Most predictions show higher AF2 confidence than ESMFold; this is most likely because of the use of MSAs and templates in AF2 prediction as opposed to just sequence with ESMFold. Overall, these results appear to rule out the possibility that analysis is influenced by artefacts from ESMFold structure prediction.

Next, ESMFold predicted encapsulin structures from several clusters were further investigated to find potentially novel features. Each structure cluster were further clustered at 80% sequence identity to remove redundancy, and all predicted structures from these reduced clusters were visually inspected in PyMOL. As expected, predicted structures from Cluster 8 (which contained 3/4 of the experimental structures analysed) all resembled known encapsulin structures from the literature (Figure 3.6.3a), with E-loops either in the "T=1-like" conformation or in a position resembling the T=3 or T=4 encapsulins. However, encapsulins from other clusters displayed some conformational diversity compared to experimentally solved structures. For example, putative encapsulins from Cluster 1 showed insertions in the A-domain apical loop region, and a longer E-loop. Predicted structures showed favourable pLDDT and predicted aligned error (PAE) values (Figure 3.6.3b).

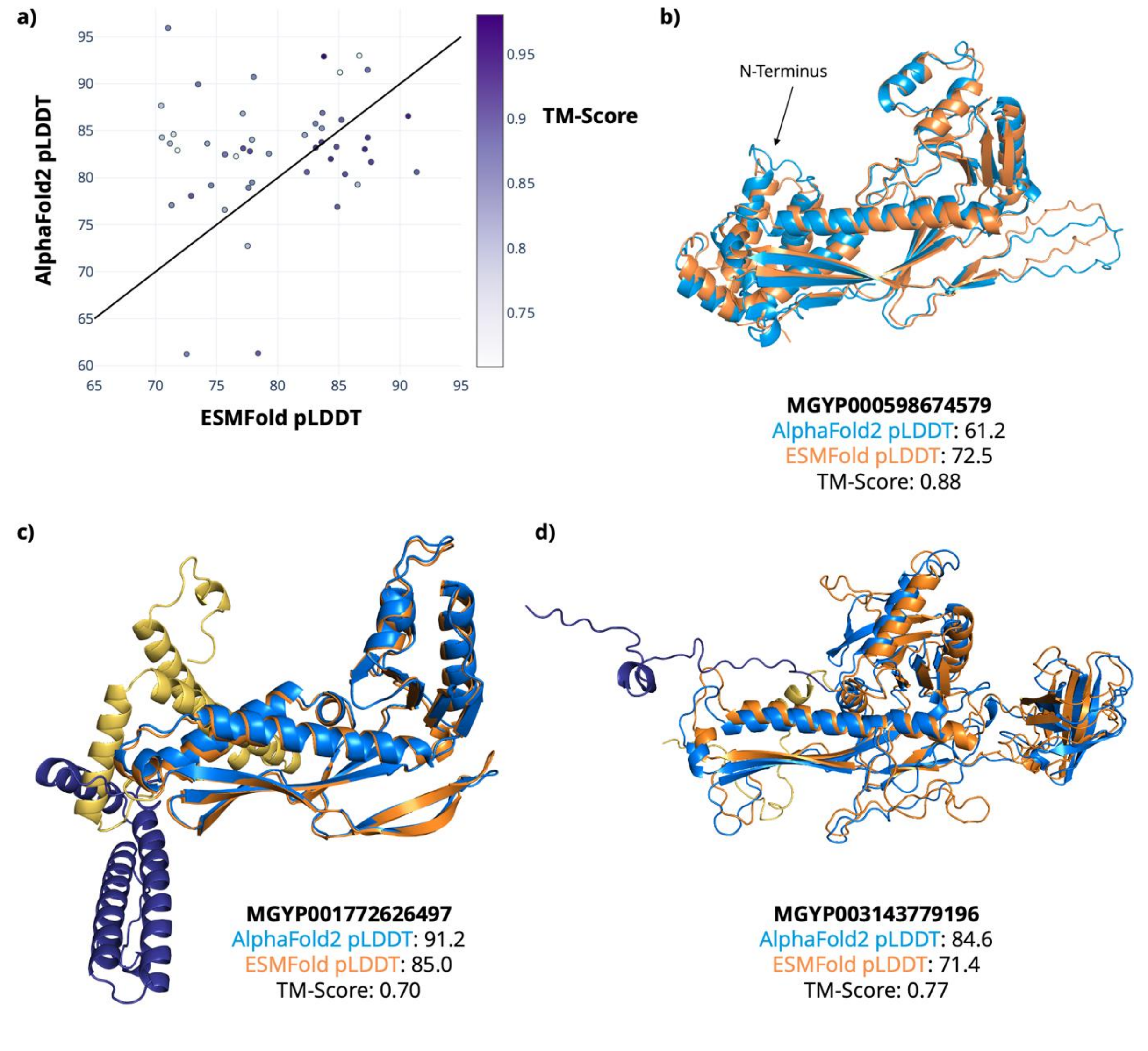


***Figure 3.6.2:* Comparison of Representative AlphaFold2 and ESMFold Predicted Structures**
48 representative sequences were chosen from ESMFold predicted structure clusters and predicted with AlphaFold2.
a) Plot of ESMFold pLDDT against AlphaFold2 pLDDT, showing all but two structures have mean AF2 pLDDT above 70. Similarity between ESMFold and AF2 predictions as measured by TM-Score was always 0.7 or higher, and in all but 5 cases TM-Scores are above 0.8, indicating good agreement between predictions.
b) Visual comparison of ESMFold (orange) and AF2 (blue) predictions where the AF2 prediction shows low confidence. Despite the lower AF2 confidence, the two predictions agree in the HK97 fold and differ only in their prediction of an extended N-terminal region.
c), **d)** Visual comparison of ESMFold and AF2 predictions for two putative encapsulins with TM-Score below 0.8. Despite this, structures agree in both cases and differ only in extended N-terminal regions which appear disordered (shown in darker blue for AF2 and yellow for ESMFold). In **c)** this appears to be a large insertion at the N-terminus. Both methods predict it to be mainly helical, although in differing conformations. Such flexible fusion domains are found in the N-terminus of some Family 1 encapsulins. In **d)** this N-terminal region has an extended loop, comparable to the N-arm of the *S. elongatus* T=1 encapsulin. Aside from these terminus regions indicated, ESMFold and AF2 predicted structures appear to be in good agreement. Figure reproduced from [238].

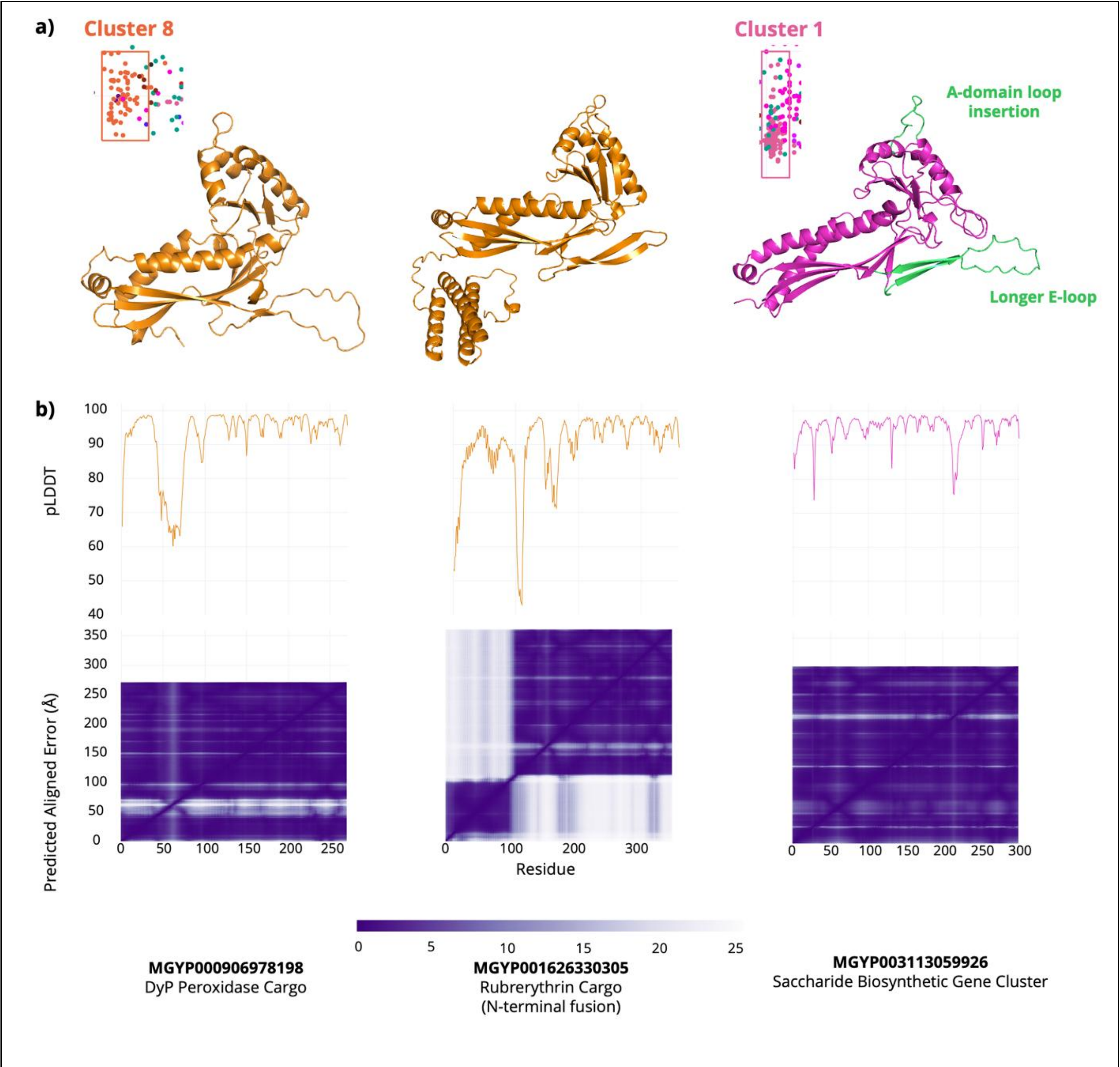


Figure 3.6.3: **ESMFold Predicted Structures of Metagenomic Encapsulin Candidates**
a) Cluster 8 (orange) contains predicted structures closest to experimentally solved encapsulin structures. Some resemble the *T. maritima* T=1 encapsulin (left), while others have an E-loop angle closer to higher T-number encapsulins from M. xanthus and Q. thermotolerans (middle). Cluster 1 (right, pink) contains structures with minor variations, including insertion loops in the A-domain and a longer E-loop.
b) pLDDT (top) and PAE (bottom) plots for the three predicted structures. All structures have a mean pLDDT above 0.7. MGYP000906978198 and MGYP003113059926 showed acceptable mean PAE of 3.7 and 2.9 Å respectively. MGYP001626330305 has mean PAE around 11 Å, however this was due to the presence of two distinct domains structure with poor PAE values against each other, visible in the plot as dark coloured squares separated by regions of brighter colour. These two domains correspond to the main HK97 fold of the encapsulin, and the N-terminal rubrerythrin cargo fusion. The mean PAE within these regions was under 4 Å, indicating two confidently predicted domains whose positioning relative to each other is uncertain. Figure reproduced from [238].

However, the predicted structures from Cluster 6 appeared to be the most interesting. These candidates all displayed several interesting features not seen in known encapsulins (Figure

3.6.4a). This included large insertion domains in the E-loop which could not be identified by sequence or structure searches against existing databases. These predicted structures also showed insertion of a small β-strand in the G-loop region, which is not seen in experimentally solved encapsulin structures. The positioning of these insertions in E- and G-loops indicates that these domains could decorate the outside of the assembled capsid shell (Figure 3.6.4b).

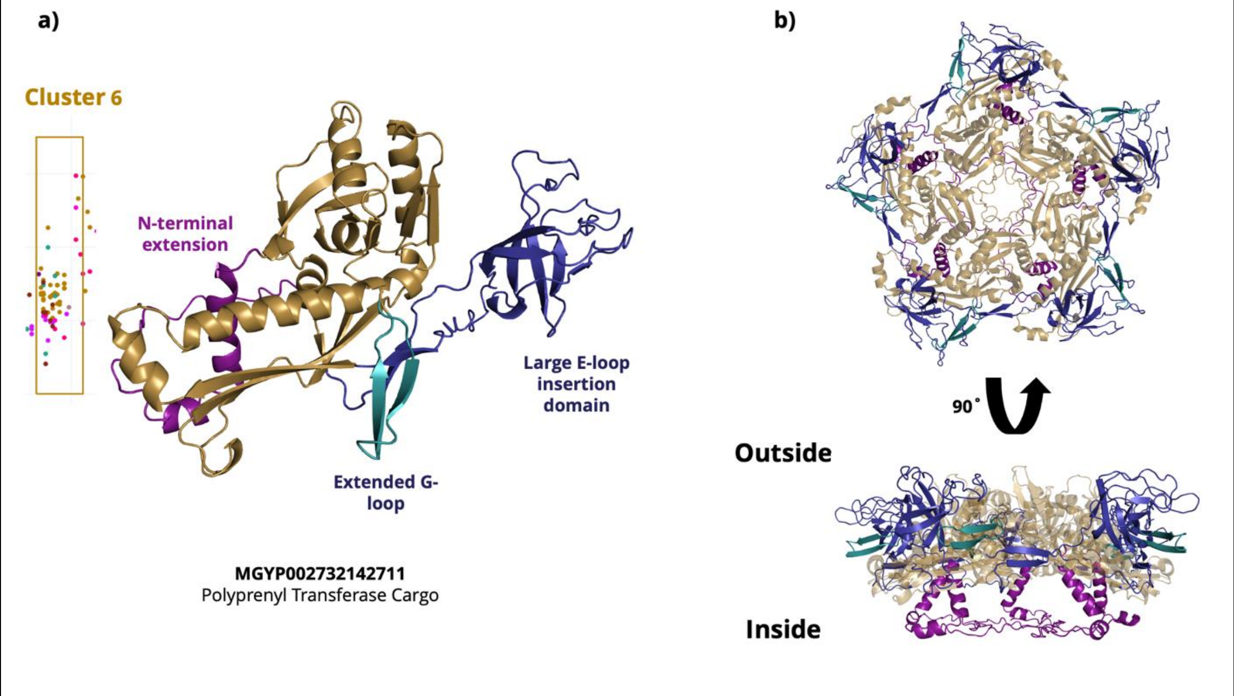


Figure 3.6.4: **A Cluster of Encapsulin Candidates with Novel Structural Features**
a) Cluster 6 contains novel encapsulin predicted structures with a large insertion domain in the E-loop (dark blue), and an extended G-loop (teal) not seen in experimental encapsulin structures. Some predicted structures contain N-terminal extensions or fusion domains (purple). **c)** Alignment with the T. maritima encapsulin pentamer shows E-loop and G-loop extensions are predicted to decorate the outer surface of the capsid. Pentamer fitting is not energy minimised and shown for schematic purposes only. Figure reproduced from [238].

3.7 Experimental Characterisation of Metagenomic Encapsulin Candidates

Next, several metagenomic encapsulin candidates were chosen for experimental characterisation. Experimental efforts were focused on the Cluster 6 candidates since these appeared to show interesting and potentially useful new structural features. Three sequences from Cluster 6 were chosen for experimental characterisation: MGYP000631152961, MGYP002732142711, and MGYP000702098687. These will henceforth be referred to as MGYP-61, MGYP-11 and MGYP-87 for brevity. These Cluster 6 sequences were chosen due to the presence of an annotated family 2 cargo protein on the same contig as the candidate sequence (polyprenyl transferase for MGYP-61 and MGYP-11, and xylulose kinase for MGYP-87). Protein sequences were codon optimised for *E. coli*, ordered as synthetic genes, and cloned into pSB1C3-FB for expression under the *T7* promoter.

Figure 3.7.1a shows the confidence metrics for the three chosen candidate sequences – all have satisfactory mean pLDDT values, sub-5 Å PAE values within the majority of the HK97 fold, and only displaying higher PAE values in E-loop insertion domain region. All three proteins were expressed in *E. coli* in 6 ml cultures, at three different temperatures (Figure 3.7.1b). MGYP-11 appeared to show poor expression and low soluble yield across all temperatures. Whilst MGYP-87 displayed a strong band on SDS-PAGE of the expected size for the insoluble fraction, the same band appeared much fainter in the soluble fraction across all temperatures, indicating that most of the expressed protein was insoluble. However, MGYP-61 showed intense bands of the expected molecular weight for both soluble and insoluble fractions at 18 ˚C and 25 ˚C. As such, given its good expression levels and high soluble yield compared to the other two candidates, only MGYP-61 was carried forward to further experiments.

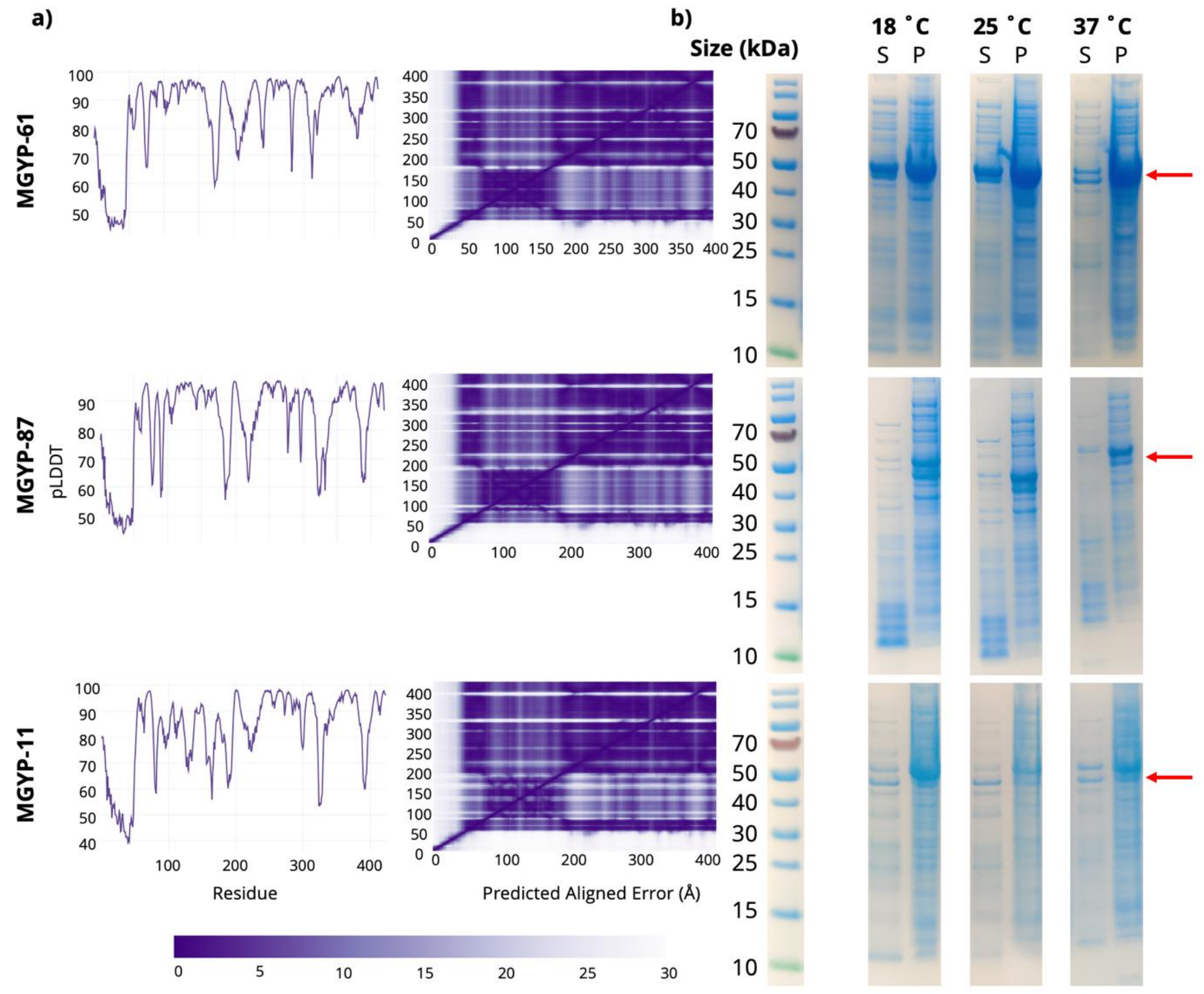


Figure 3.7.1: **Confidence Metrics and Solubility Screening of Metagenomic Encapsulin Candidates**
a) pLDDT (left) and PAE (right) plots for the three chosen experimental metagenomic encapsulin candidate sequences. All sequences have mean pLDDT above 80. Mean PAE for each structure is between 10-11 Å but this drops to 6-7 Å when excluding the low confidence N-termini. This is due to the uncertain alignment of residues 70-180 (the E-loop and insertion domains) with the rest of the structure. **b)** SDS-PAGE gels showing the soluble (S) and insoluble (P) fraction of *E. coli* cell lysate following expression of the three proteins at three different temperatures – 18 ˚C, 25 ˚C, and 37 ˚C. All three proteins have an expected molecular weight of 47 kDa, with the overexpressed protein band indicated with red arrows. MGYP-61 appeared to have the highest level of soluble expression based on the intensity of the band, followed by MGYP-87, with MGYP-11 showing a comparatively poor yield of soluble protein. Note that cropped gel lanes shown here come from three separate SDS-PAGE gel images; one gel was run for the samples for each different protein.

3.8 Further Experimental Characterisation of MGYP-61

Next, MGYP-61 expression in *E. coli* was scaled up to the flask scale, and protein purified using affinity chromatography using Streptactin columns. Purified protein from affinity chromatography was pooled, concentrated, and run on a Native-PAGE gel. This sample was then polished using size exclusion chromatography. Unfortunately, at the time these experiments were carried out, a gel filtration column with the appropriate fractionation range for nanoparticles was not available. Concentrated protein was therefore loaded onto a Superdex 200 column with a 10-600 kDa fractionation range. However, this was deemed unlikely to affect chromatography results, since the hydrodynamic radius of assembled encapsulin particles should be too large to enter the column's gel matrix, and thus, these are expected to elute in the void volume.

As shown in Figure 3.8.1a, pure MGYP-61 was recovered in high yield from large-scale *E. coli* cultures. Native-PAGE analysis of pure MGYP-61 fractions (Figure 3.8.1b) indicated that the protein does not assemble into large nanoparticles as expected from an encapsulin, with no large molecular weight species visible. The MGYP-61 SEC trace (Figure 3.8.1c) showed a broad peak formed of two components – a narrow, earlier component and a broad, later component. This suggests that there could be two distinct populations, one of a larger molecular weight and one smaller. However, both fractions eluted outside the void volume for this column (30 ml). Fractions comprising both parts of this peak were separately pooled and concentrated for further analysis. DLS measurements (Figure 3.8.1d) seemed to indicate that both early and late fractions contained a broad distribution of assembled particle sizes, centred around ≈70 nm. These broad DLS peaks have long tails (truncated on the plot shown in Figure 3.8.1d for clarity) which may indicate the presence of high molecular weight aggregates. TEM images (Figure 3.8.1e) conclusively showed that MGYP-61 does not assemble into nanoparticles, with both early and late samples showing no visible assembled particles.

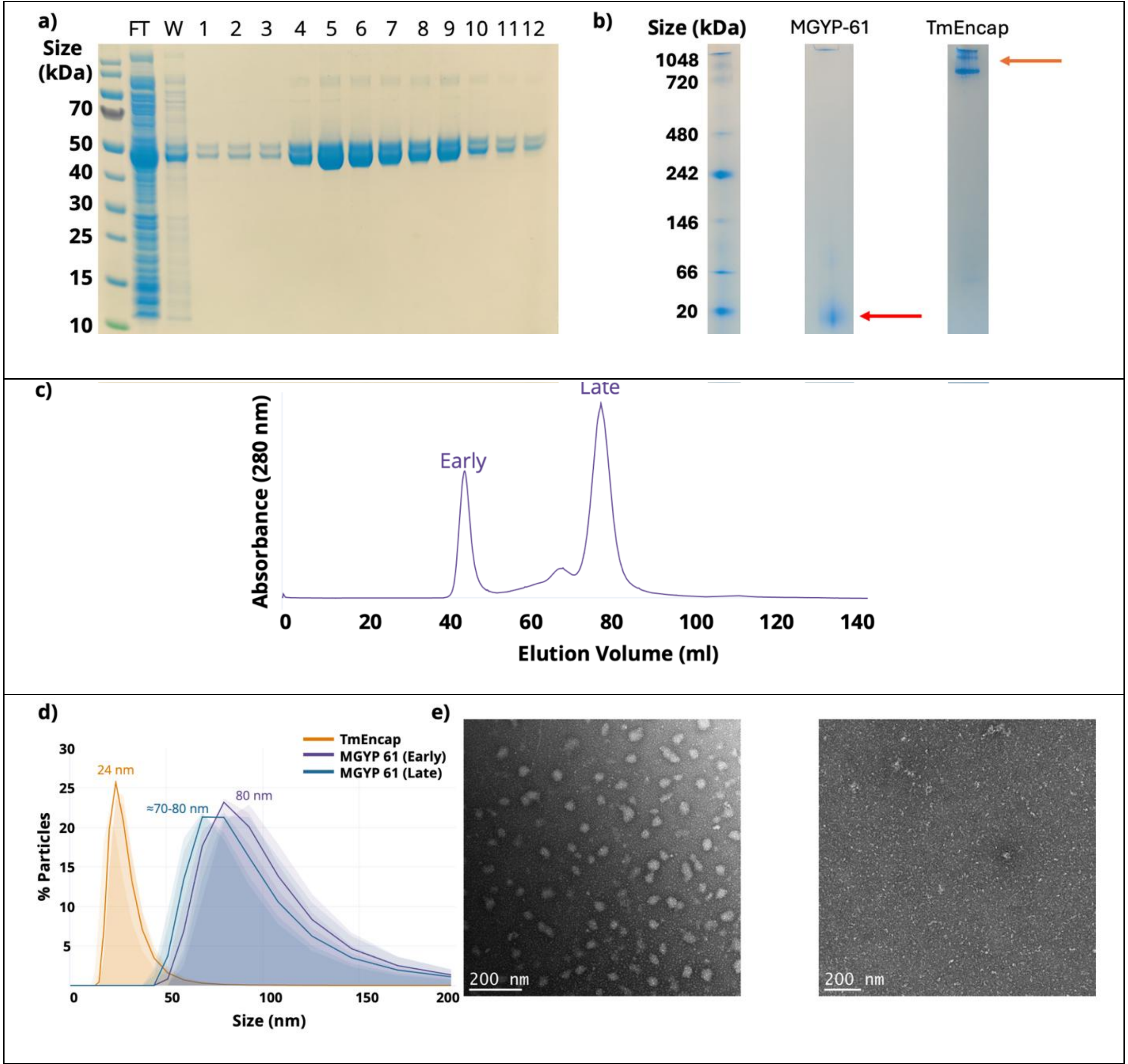


Figure 3.8.1: **Biochemical and Biophysical Characterisation of MGYP-61**
a) SDS-PAGE of MGYP-61 fractions purified using affinity chromatography from large-scale *E. coli* expression cultures. **FT** = flow-through, **W** = wash. Numbered fractions indicate elution fractions from the Streptactin column, showing high yield of pure protein. **b)** Native-PAGE of pooled fractions from MGYP-61 affinity chromatography and a pure TmEncap control. TmEncap shows the higher molecular weight bands associated with capsid assembly (orange arrow), but MGYP-61 does not show any higher molecular weight bands, only a low molecular weight smear (red arrow). **c)** Chromatography trace of Streptactin purified MGYP-61 fractions loaded on a Superdex 75 gel filtration column (10-600 kDa fractionation range). MGYP-61 appears to show a doublet peak composed of early and late components. **d)** DLS intensity data (averaged by number of particles) for the MGYP-61 early and late fractions from SEC (separately pooled and concentrated) and a TmEncap control. Shaded areas represent 3 measurement runs for each sample, and solid lines represent mean values of these measurements. TmEncap shows the expected narrow distribution around 24 nm, where both MGYP-61 fractions show broad distributions centred around 70-80 nm sizes. **e)** TEM images showing MGYP-61 early (left) and late (right) fractions. Neither sample appears to contain assembled particles. Bright spots seen in the MGYP-61 early fraction image (left) are staining artefacts.

3.9 Discussion

3.9.1 Metagenomic Encapsulin Discovery

In this chapter, a dataset of novel putative encapsulin sequences is presented, leveraging the rapid growth in metagenomic databases, and the wealth of new sequence diversity contained within them. This diversity presents many opportunities for discovery of novel proteins; however, it also brings to light several challenges which were encountered in this work. Analysing metagenomic encapsulin hits by functional prediction is a non-trivial task, as seen in the relatively slim proportion of candidate encapsulin sequences that could be annotated with a feasible cargo type. This could be because of low sequence identity (often sub-30%) of putative cargo proteins with any protein of known function, or it could also indicate that putative encapsulins in this dataset are associated with novel cargo proteins whose function hasn't previously been observed in known encapsulin systems. The scarcity of genomic context surrounding metagenomic encapsulin hits also makes removing phage proteins difficult, requiring a much more involved search and filtering strategy compared to previous work. Many initial candidate sequences had to be removed due to small contig sizes, and contigs could not be retrieved for many candidate sequences due to missing metadata in the MGnify Protein Database (Figure 3.2.1c).

It must be noted that the Pfam annotations used in the initial search stage of this work were generated using ProtENN, a previously described deep learning model which has been demonstrated to be more accurate than conventional HMM-based approaches [138], particularly on sequences with remote homology to any existing annotated sequence. The authors of ProtENN suggest combining deep learning predictions with traditional HMM approaches for optimal performance and coverage. Unfortunately, re-annotating all 2.4 billion protein sequences using HMMs is far beyond the scope of this study, especially when ProtENN annotations are already provided, and are likely to be more accurate for the novel sequences sought after in this work. Aside from its use in the MGnify Protein Database, there are no examples in the literature of ProtENN being applied to metagenomics data. This work is the first study focusing on genome mining of the MGnify Protein Database, and as such the first detailed interrogation of ProtENN annotations applied to metagenomics data.

Despite the rigorous filtering strategy employed in this study, there is still a chance that some candidate encapsulins presented here are phage proteins and not encapsulin proteins of cellular origin. Encapsulins and phage capsid proteins from conventional databases can show as little as 20% sequence identity despite similarity in tertiary and quaternary structure. The degree of uncertainty is compounded by the fact that the metagenomic candidate sequences presented here show low identity with sequences in conventional biological databases, for which species and functional annotations are available. This twofold issue of low sequence identity is then also complicated by the unclear evolutionary history of the HK97 fold: as described in Section 0 it is unknown whether this family of proteins originated in viruses or cellular organisms. Due to this obscure evolutionary relationship, putative encapsulin hits may resemble phage capsid proteins in sequence or structure, however it is impossible to rule out the scenario that they are very primitive cellular proteins close to the HK97 fold's common ancestor. Genomic context can give clues as to a protein's origin, however in the metagenomic case this information is limited, and functional annotation of neighbouring genes is troublesome. Indeed, an ancient phage-like encapsulin sequence could be neighboured by primitive genes appearing viral in character, and with limited sequence identity and annotations it would be very difficult to decide whether these genes are cellular or viral in origin. These thought experiments serve to demonstrate the difficulty in distinguishing encapsulins from phage capsid proteins, and more broadly to discriminate between viral and cellular proteins. Notwithstanding the presence of such thought-provoking examples, this work strives to use all available information to rule out the presence of phage capsids where possible.

3.9.2 Putative Encapsulin-Associated BGCs

Biosynthetic gene cluster prediction revealed a potentially novel class of encapsulin-associated BGC, the Saccharide BGC. This may be an interesting new class of encapsulin system involved in producing cytotoxic or antimicrobial saccharides, as predicted by deep learning tools. The precise substrates and products of these saccharide pathways are not known. However, given the presence of putative encapsulins in these systems, and that the predicted product classes of these systems are antimicrobial/cytotoxic, it is assumed that these saccharide pathways produce toxic products or intermediates, hence the requirement for enzyme encapsulation. There are many known glycosylated cytotoxic natural products in bacteria, for example the substituted aminoglycoside pactamycin [243]. However, it is important not to draw too strong a conclusion from the BGC prediction data; BGC prediction algorithms are notoriously error-prone and are known to produce many false positives [244]. Indeed, 80 false positive "RiPP-like" BGCs were predicted by antiSMASH in this work, emphasizing the care needed in interpreting BGC prediction data.

Putative cargo proteins from several Saccharide BGC examples returned no informative hits when searched using BLAST, or when predicted structures were searched against the PDB using Foldseek. The few significant (E-Value < 10-3) sequence hits from BLAST were all hypothetical proteins with no annotated function. Structure hits only showed insignificant structural similarity over small regions (TM-Scores and probabilities below 0.5). The number and accuracy of BGC predictions is limited in this case by the genomic context available in the contigs surrounding each candidate encapsulin, which explains the relatively few BGC predictions observed in this study. Such tools are usually intended to be run on full genome sequences. BGC prediction tools also make use of Pfam and other functional annotations, which have their own limitations with metagenomic proteins as previously mentioned. Given the limitations in the underlying data and the tools used, Saccharide BGCs remain a hypothetical new biological function for encapsulins until experimental characterization can be done.

The same limited conclusions can be drawn from the observation of known family 1 or 2 cargos within other types of BGC, including Saccharide BGCs. The limitations of the data presented here indicate that this is simply a coincidence, however an encapsulin and its associated cargo forming part of a larger cluster of metabolic genes is an interesting possibility. It is speculated that if this were to be observed in a more significant number of genomes or metagenomic contigs, this could have implications for encapsulated ferritin or cysteine desulfurase function as part of a larger cluster of genes involved in secondary product metabolism. It is noted that BGC prediction in the context of encapsulins has not previously been carried out on as large a scale as in this work, and such approaches could be applied to the existing encapsulin datasets to potentially give new insights into biological function.

3.9.3 Experimental Characterisation of Metagenomic Encapsulins

Predicted structures of putative encapsulin hits revealed some interesting new structural features. Whilst many of these predicted structures show similar topology to experimentally resolved encapsulin structures, the predicted structures from Cluster 6 display a set of novel structural features compared to known encapsulins. Initially, it seemed that insertions in the E-loop and A-domain could decorate the vertices of pentameric units in the capsid shell, and it was speculated that these insertion domains may lead to architectural differences in these putative capsids. Three of these candidates were chosen for experimental characterisation, and two of them showed good expression levels in *E. coli*. The best expressing candidate protein was purified and investigated using biophysical techniques, however it appears that this candidate (MGYP-61) does not form assembled particles under the conditions used in this study.

3.9.4 MGYP-61 Experimental Characterisation

There are several possible explanations as to why MGYP-61 was not observed to form assembled particles following expression in *E. coli* and purification. It is possible that the experimental conditions used in this study were not conducive to capsid formation. Only the capsid protein was overexpressed, but co-expression of the putative cargo protein may be necessary for capsid formation. Indeed, it has been observed in some encapsulin systems that co-expression of the cargo protein can have a "scaffolding" effect and direct assembly of the shell. In the *M. xanthus* encapsulin, expression of just the shell protein alone leads to T=1 sized particles, whereas co-expression of the cargo leads to the formation of a larger T=3 shell [22]. Whilst cargo proteins have never been implicated as essential for encapsulin assembly, this could be the case for MGYP-61 and its polyprenyl transferase cargo. The pH and salt concentration used (150 mM NaCl, pH 8.0) could be potentially suboptimal for capsid formation. However, encapsulins are generally stable across wide pH ranges, typically pH 3-12 [245]. As such, it is surprising that no assembled particles at all were observed under these conditions. Similarly, whilst assembly could be disrupted by inclusion of a Strep tag for affinity purification, or by strong overexpression using 1 mM IPTG under the *T7* promoter (as opposed to more gentle overexpression techniques like autoinduction, or lower inducer concentration), neither of these is observed to affect assembly in known encapsulins, so this is unlikely.

Alternatively, it is possible that MGYP-61 is not an encapsulin at all, but in fact a phage capsid protein, or an otherwise inactive/non-functional protein that resembles an encapsulin or a phage capsid protein. As noted previously, it is possible that the metagenomic encapsulin dataset presented here contains some phage capsid proteins. Furthermore, whilst MGYP-61 does have an annotated polyprenyl transferase cargo protein, this gene is found several kilobases upstream of MGYP-61, in the opposite orientation. In almost all experimentally investigated encapsulin/cargo systems, the shell protein and the cargo protein are found adjacent to each other in the genome. However, it has been observed that the *M. xanthus* encapsulin encapsulates several different iron-binding cargo proteins, including two proteins located several kilobases upstream and downstream of the shell protein [22, 27, 246].

Native-PAGE, SEC, and TEM experiments conclusively demonstrated that MGYP-61 does not assemble into particles under the conditions tested here, however DLS measurements showed broad peaks resembling an assembled particle population. It is unknown why DLS produces these false positive results. It is possible to speculate that this is due to the minute presence of protein aggregates in pure samples, or the presence of other contaminating species such as lipids or nucleic acids. Alternatively, it could be due to the overfitting of the DLS data by the analysis packages used. DLS experiments measure the intensity fluctuations in laser light scattered by particles in the sample. These intensity fluctuations are used to calculate an autocorrelation function, which is then used to fit various other parameters, including a diffusion coefficient which is related to the particle diameter [247].

Despite this high degree of model fitting and processing of the raw data, the proprietary Malvern software for the instrument used in this study only outputs the intensity-weighted distribution as raw data, as well as other processed forms of this data. As such, it is possible that overfitting has occurred somewhere "behind the scenes" between data acquisition and data output for plotting. DLS measurements overall are prone to high variability; in Frank Lab unpublished data, variation in DLS diameters has been observed across different wild-type encapsulin samples, even in the same sample on subsequent experimental runs. As such, the DLS results here are false positives arising from experimental or data processing artefacts and are conclusively contradicted by other biochemical and biophysical measurements.

3.9.5 Future Work

In this chapter, three encapsulin candidates were experimentally tested, from a potential dataset of over 1000 new sequences. As such, the natural next steps are to experimentally characterize more of these candidates. Candidates can be screened for expression and solubility before purification and biophysical characterization as presented in this work. Since around 120 candidates have been annotated with a putative cargo protein, these sequences could also be co-expressed with the cargo protein to assay function and assembly. The Saccharide BGC candidates could even be expressed with the entire putative operon, to investigate potential secondary product synthesis in these systems. Indeed, future work should focus on these interesting BGC candidates, establishing their biosynthetic function(if any) and how the putative encapsulin protein is involved in this function.

3.9.6 Conclusion

To conclude, this study presents exploratory work towards discovering new encapsulin sequences in metagenomic databases, and the workflows required to filter and analyse these sequences. These new data may be useful in understanding encapsulin biology and/or in developing new engineering applications for encapsulins. One of these encapsulin candidates gave unsatisfactory results when investigated experimentally, highlighting the strong need for experimental screening following the computational discovery process. The data presented here may provide a platform for the discovery of more novel encapsulin systems in future.

Chapter 4: Developing a High Throughput Design Pipeline for Encapsulin Proteins

4.1 Background

One of the aims of this work was to generate novel encapsulin variants with increased solubility and expression yield. This requires high-throughput, scalable methods for computational protein design and screening, as well as experimental characterisation in the laboratory. Both computational and laboratory methods must be scalable to large sets of candidate proteins – on the order of thousands or hundreds of thousands *in silico*, and hundreds in the laboratory using 96-well plates.

The following section will outline the development of this pipeline, starting with experiments using *in silico* protein design tools, and methods for computationally validating designed sequences. Finally, the establishment of a high-throughput workflow for cloning, expression, and solubility screening of encapsulin candidates is presented. This experimental workflow can be applied to *de novo* designed encapsulin proteins, or natural candidates from bioinformatics databases (such as those presented in the previous chapter).

4.2 Encapsulin Design and *In Silico* Screening Experiments

4.2.1 Structure Prediction Benchmarking

A key principle of protein design in the deep learning era is the use of structure prediction. Typically, candidate protein sequences are used as the input to structure prediction tools such as AlphaFold2 [134] and the resulting structure predictions compared to experimentally solved structures of the design template. The quality of designed sequences is inferred from the confidence of their predicted structure, as measured by the average predicted Local Distance Difference Test (pLDDT) value over all residues. Design quality can also be estimated from the structural similarity between the predicted structure and the template; there are several measures of structural similarity between two protein structures, but the metric used in this study is the Template Modelling score (TM-Score), since it is generally robust to variations in the length of structures [225].

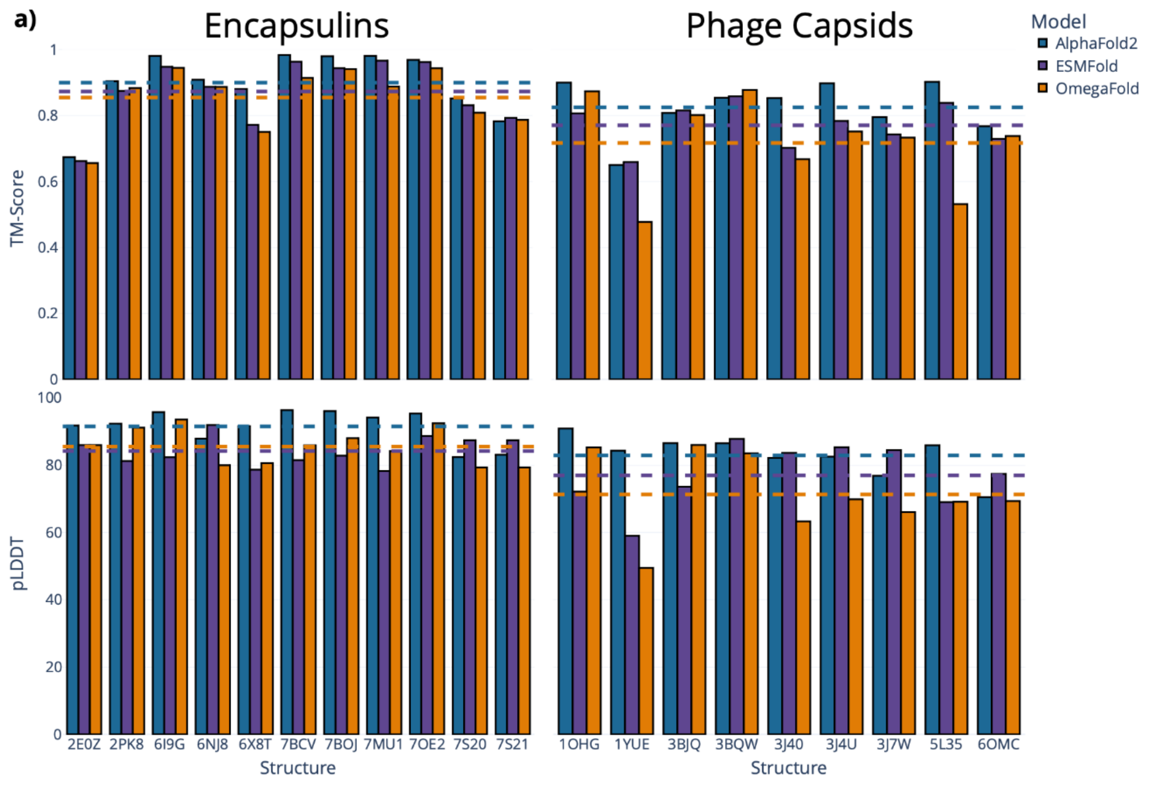
This process is sometimes called "refolding" [172] and is often used in the validation of protein design tools [162, 163]. AlphaFold2 is the *de facto* standard in protein structure prediction, however it can take several minutes or longer to generate a prediction, owing to the time-consuming step of searching genetic databases to generate a multiple sequence alignment (MSA). This can be a significant issue if the scalable screening of thousands of candidate protein sequences is required, as in this work. However, the advent of MSA-free structure prediction tools relying on protein language models, such as ESMFold [137] and OmegaFold [224] may provide a potential solution to this problem of scaling.

To investigate this, the prediction performance of AlphaFold2, ESMFold, and OmegaFold was tested on a representative set of experimentally solved, HK97-fold protein structures (Table 4.1) taken from [26]. These structures encompass the structural diversity of the HK97-fold across phage capsid proteins and encapsulins, with diverse T-numbers, architectures, and biological functions.Figure 4.2.1 shows the results of this benchmark experiment.

As expected, AlphaFold2 shows the best performance across both encapsulins and phage capsid proteins but is orders of magnitude slower than MSA-free methods. ESMFold shows better performance than OmegaFold across both encapsulins and phage capsid proteins and generates outputs an order of magnitude faster than OmegaFold and two orders of magnitude faster than AlphaFold2. ESMFold showed acceptable performance on these representative structures, with all TM-Scores above 0.6, and all pLDDT scores above 60 apart from the phage capsid protein 1YUE. ESMFold was thus chosen for all structure prediction experiments in this study (unless otherwise stated), due to this combination of adequate performance and quickly obtainable results.

Table 4.1: **Protein Structures used for Structure Prediction Benchmarking**
FLP = Ferritin-like protein **DYP** = Dye-decolourizing-type Peroxidase
IMEF = Iron-Mineralizing Encapsulin-Associated Firmicute Protein

PDB ID	Species	Type	Pfam	T-Number
7MU1	*Thermotoga maritima*	Encapsulin (FLP Cargo)	PF04454	1
7BOJ	*Mycobacterium smegmatis*	Encapsulin (DYP Cargo)	PF04454	1
6I9G	*Mycobacterium hassiacum*	Encapsulin (DYP Cargo)	PF04454	1
7OE2	*Haliangium ochraceum*	Encapsulin (FLP Cargo)	PF04454	1
7BCV	*Brevibacterium linens*	Encapsulin (DYP Cargo)	PF04454	1
7S20	*Myxococcus xanthus*	Encapsulin (FLP Cargo)	PF04454	3
7S21	*Myxococcus xanthus*	Encapsulin (FLP Cargo)	PF04454	1
2E0Z	*Pyrococcus furiosus*	Encapsulin (FLP Cargo)	PF04454	3
6NJ8	*Quasibacillus thermotolerans*	Encapsulin (IMEF Cargo)	PF04454	4
2PK8	*Pyrococcus furiosus*	DUF2184	PF08967	Unknown
1OHG	*Escherichia* virus HK97	Phage capsid family	PF05065	7
6OMC	*Escherichia* virus T5	Phage capsid family	PF05065	13
3BJQ	*Bordetella bronchiseptica*	Phage major capsid protein E	PF03864	Unknown
3BQW	*Escherichia coli* CFT073	Phage major capsid protein E	PF03864	Unknown
1YUE	*Escherichia* virus T4	Major capsid protein Gp23	PF07068	13/20
5VF3	*Escherichia* virus T4	Major capsid protein Gp23	PF07068	13
5L35	*Shigella flexneri* bacteriophage Sf6	P22 coat protein Gp5	PF11651	7
3J7W	*Escherichia* phage T7	P22 coat protein Gp5	PF11651	7
3J40	*Salmonella* phage Epsilon15	DUF2184	PF08967	7
3J4U	*Bordetella* phage BPP-1	DUF2184	PF08967	7



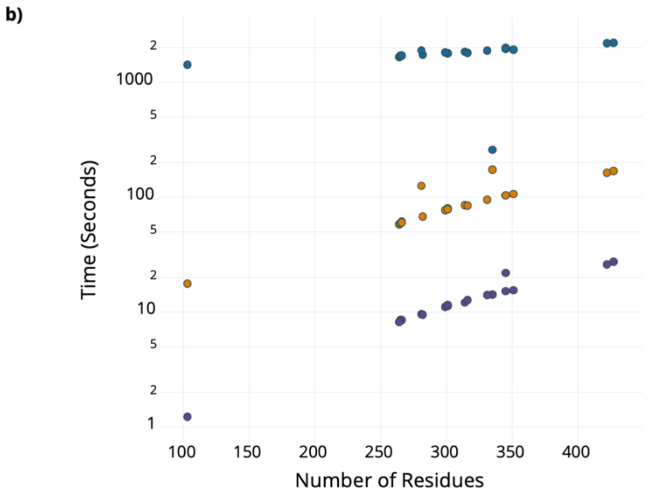


***Figure 4.2.1:* Benchmarking Structure Prediction Methods on HK97-Fold Proteins**
a) TM-Scores and **b)** pLDDT confidence values for encapsulins (left) and phage capsid proteins (right) when predicted using AlphaFold2 (blue), OmegaFold (orange) and ESMFold (purple). Mean values for each model shown with dashed horizontal lines. AlphaFold2 shows the best mean performance across both encapsulins and phage capsid proteins. ESMfold gave higher mean TM-Score than OmegaFold for both encapsulins and phage capsids, and higher pLDDT values for phage capsids. OmegaFold shows higher mean pLDDT for encapsulins than ESMFold despite worse performance as measured by TM-Score.
b) Time taken for each model to generate structure predictions as a function of protein length in amino acids. Time is shown on a log scale. ESMFold is by far the fastest method, with times an order of magnitude lower than OmegaFold and two orders of magnitude lower than AlphaFold2.

4.2.2 Inverse Folding Models for Encapsulin Design

Next, the performance of fixed-backbone protein design models was investigated. These models are sometimes called “inverse folding” models, because they generate protein sequences from an input protein structure and will henceforth be referred to as such. Two inverse folding models, ESM-IF [163] and ProteinMPNN [162] were tested with three different encapsulin monomer structures as input; the *T. maritima* T=1 encapsulin (PDB: 7MU1), the *Q. thermotolerans* T=4 encapsulin (PDB: 6NJ8) and the *S. elongatus* T=1 encapsulin (PDB: 6X8M). These three cases were chosen since they represent the diversity of solved encapsulin structures – a simple T=1 system, a larger T=4 system, and a T=1 system with an exposed N-terminal arm as in HK97 phage capsids. In addition to the design template, the effect of a design parameter known as the “sampling temperature” was also investigated in this initial experiment. Sampling temperature is a common parameter in machine learning models which generate an output probability distribution; in this case ProteinMPNN and ESM-IF both generate a distribution over all 20 amino acids, at each position in the protein backbone. It is typically expected that a lower sampling temperature produces less diverse but more accurate results, whereas a higher temperature allows more diversity to be sampled at the cost of producing lower quality sequences. This is sometimes referred to as the “exploration-exploitation trade-off” [248].

As shown in Figure 4.2.2, both ESM-IF and ProteinMPNN show lower performance at higher temperature settings, as measured by the identity of the designed sequence against the original structural template (henceforth referred to as sequence recovery). Across all temperature values and templates, ESM-IF shows a higher median sequence recovery than ProteinMPNN, apart from when temperature > 1 for the *Q. thermotolerans* encapsulin. However, as discussed in Section 1.5.2.3, sequence identity against a template can be a poor predictor of experimental success and favourable biochemical properties. Two proteins can show low sequence identity but have identical folds and similar physical properties, whereas a single point mutation can be enough to destabilise protein folding and destroy activity. Therefore, to further investigate quality of the inverse folding designs beyond sequence identity alone, protein structures were predicted for all designs using ESMFold. Design quality was measured was measured by structural similarity against the template (TM-Score) and confidence (average pLDDT).

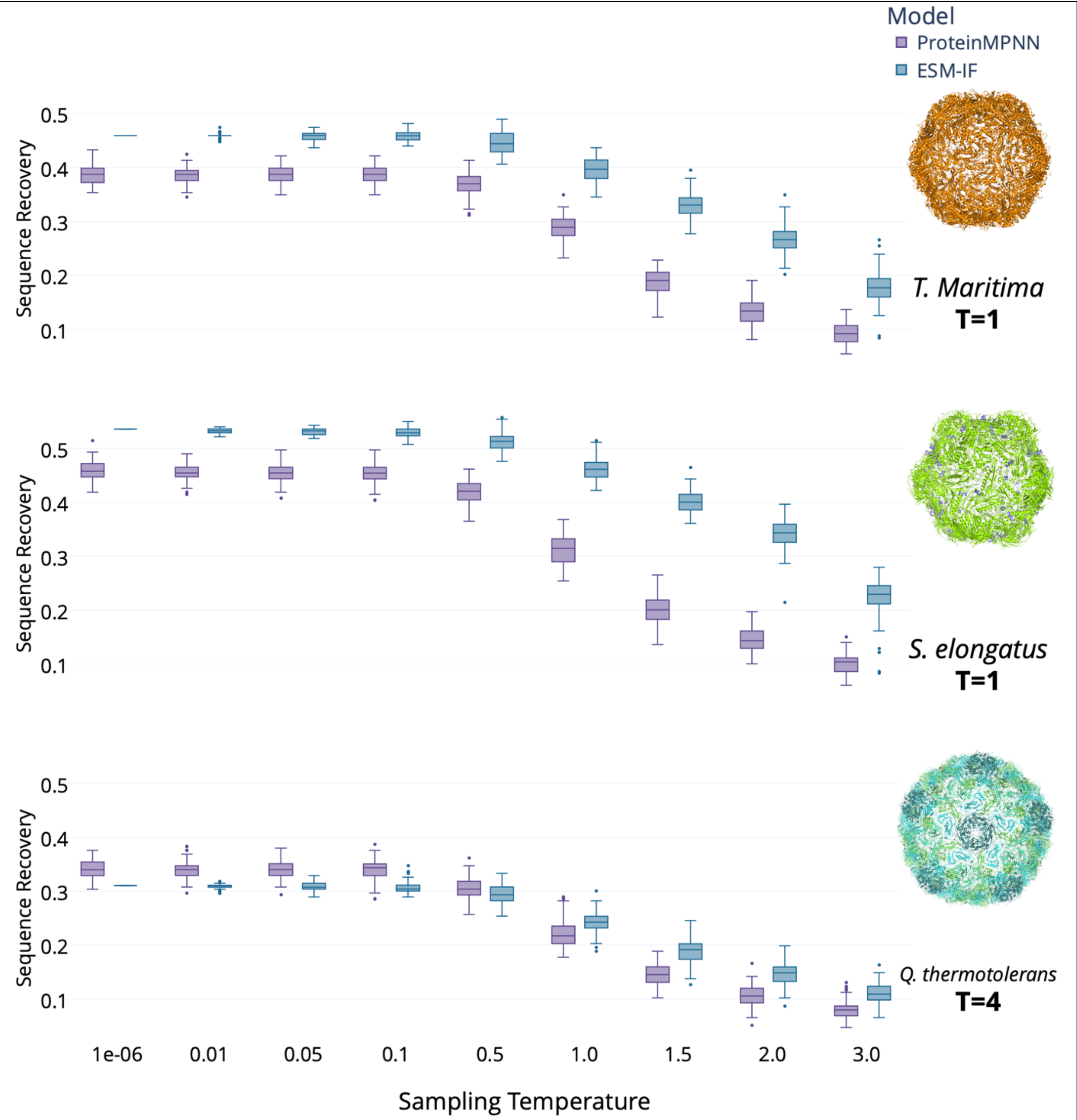


Figure 4.2.2: **Inverse Folding Performance Varies with Sampling Temperature**
Boxplots (n=100), showing distribution of sequence recovery (percent identity against the template sequence) of proteins designed against three different encapsulin templates (bottom, middle, and top). ProteinMPNN data shown in purple and ESM-IF data in blue. Sequence recovery decreases as temperature increases, across all three templates. ESM-IF shows higher median sequence recovery in all cases, except from low temperature (less than 1.0) designs against 6NJ8, where ProteinMPNN shows a higher median recovery. Note that sequences were designed using monomer structures as input – encapsulin shell structures are shown for illustration only.

Figure 4.2.3 shows that the low-temperature designs generated by ESM-IF and ProteinMPNN also show good refolding accuracy, as measured by TM-Score and pLDDT. This indicates that not only does lowering temperature improve sequence recovery, but it may also improve structural accuracy of generated sequences. Across all templates, both inverse folding models show a dense cluster of low-temperature sequences which display high TM-Scores and pLDDT (above 0.7 and 70 respectively). However, both models are also capable of generating sequences with acceptable TM-Scores above 0.7, but with low confidence (pLDDT below 50). To better compare the two models across experiments, sequences were designated as "satisfactory" if they showed both TM-Score of at least 0.7 and pLDDT of at least 70. The choice of TM-Score and pLDDT thresholds is somewhat arbitrary, and many different values are deemed "acceptable" in the literature. However, for this work, the pLDDT cut-off of 70 was chosen to match the AlphaFold database guidance for what constitutes a well-modelled protein structure [144]. The TM-Score cut-off of 0.7 was chosen based on a previously published analysis of TM-Score distributions among solved structures of natural; this analysis demonstrated that two protein structures with a TM-Score of 0.7 or higher have a 90% chance of belonging to the same fold [249].

Figure 4.2.4 shows the fraction of "satisfactory" sequences from both models, across all three templates and all temperature settings, and allows detailed comparison of model behaviour with varying temperature across the three templates. For the basic test case of the *T. maritima* T=1 encapsulin, almost every designed sequence is satisfactory when temperature is set to 0.5 or less, for both models. Above this temperature value, the fraction of satisfactory designs starts to markedly decrease, with ProteinMPNN failing to generate a single satisfactory design when temperature is set to 1.5 or 2.0.

For the *S. elongatus* encapsulin designs, ProteinMPNN displays weak performance compared to ESM-IF, with the former model never showing a success rate higher than 20% for any temperature value. ESM-IF, on the other hand, shows similar performance to the *T. maritima* test case (and generating more satisfactory sequences when temperature is set to 1.5, for example). Both models show excellent performance on the *Q. thermotolerans* encapsulin template, when temperature is 0.1 or lower. Above this temperature, ProteinMPNN shows a higher fraction of satisfactory sequences than ESM-IF, however performance for both models rapidly decreases. Neither model shows more than 10% satisfactory sequences when temperature is 1.5 or 2.0. On a final note, none of the designs generated at temperature of 3.0 were satisfactory for either model, across all three templates.

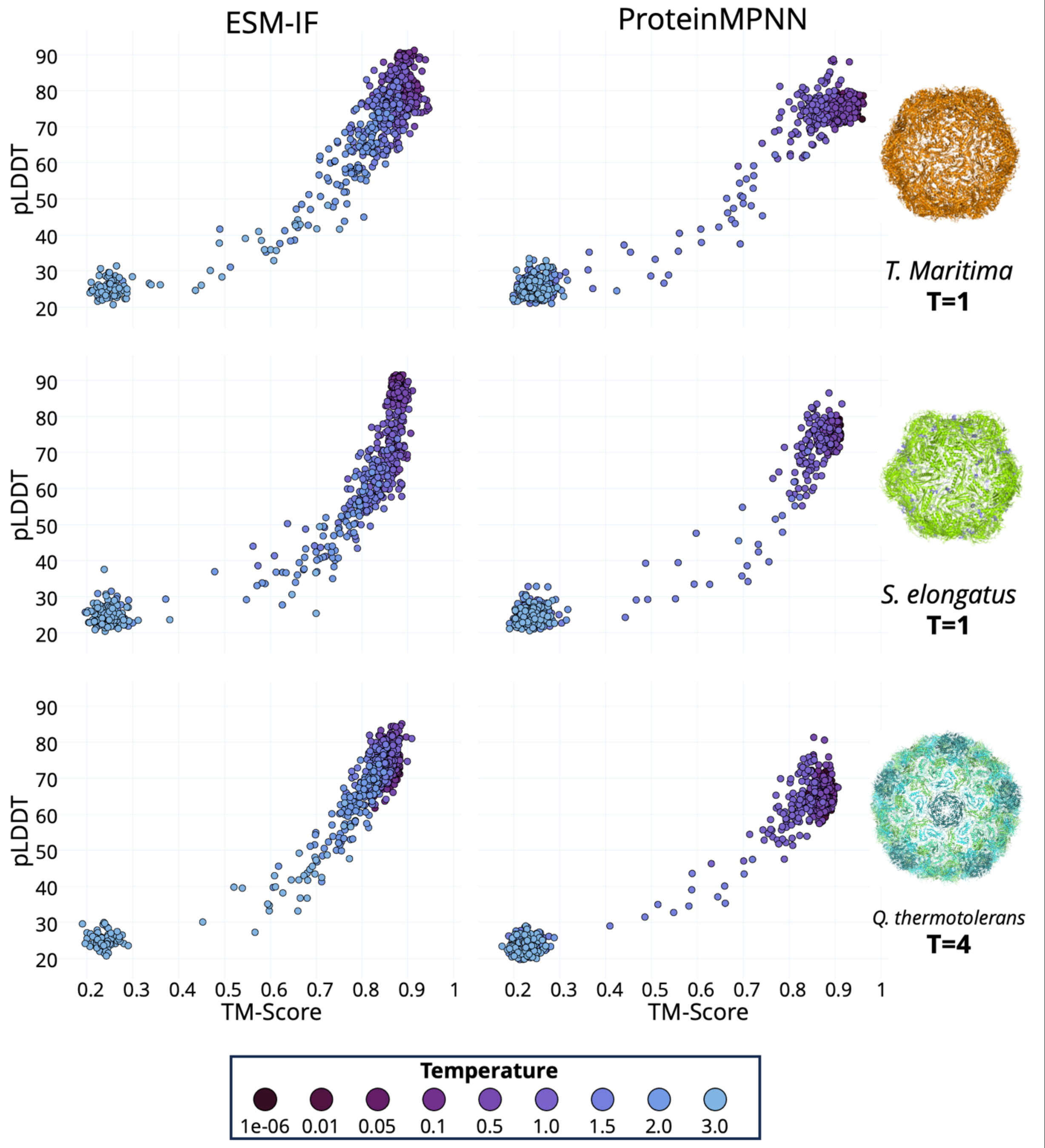


***Figure 4.2.3*: Refolding Performance of Inverse Folding Models**
Plots showing structural similarity to the template (TM-Score) and average confidence (pLDDT) of refolded protein designs, generated by ESM-IF (left) and ProteinMPNN (right), for the three different design templates. Designs are coloured by sampling temperature. Low temperature designs generally show high TM-Score (above 0.7) and high pLDDT (above 70). However, some designs show TM-Scores above 0.7 but low confidence (pLDDT of 50 or less).

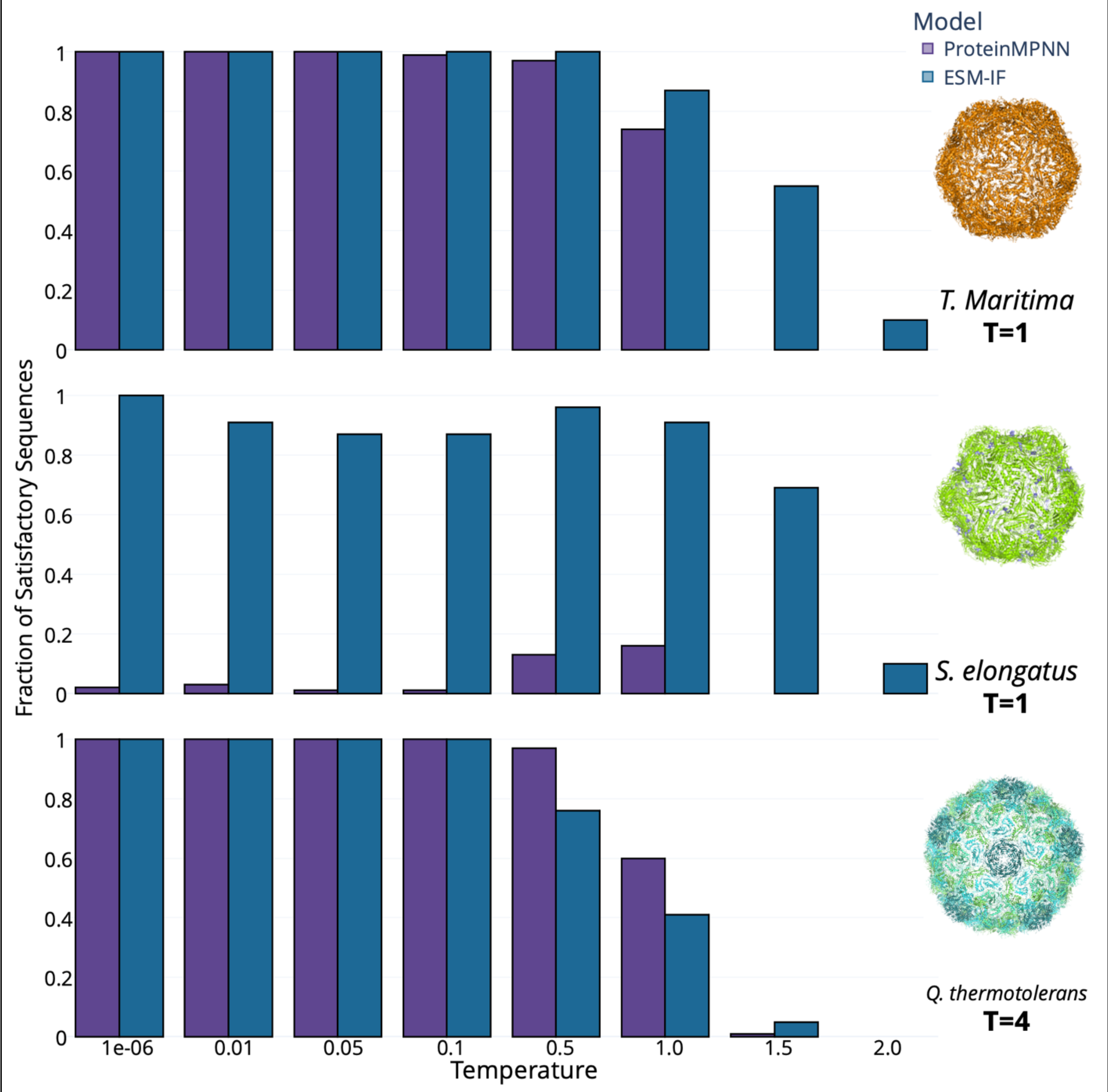


Figure 4.2.4: **Inverse Folding Models Generate Feasible Protein Sequences**
The fraction of "satisfactory" protein sequences, defined as the fraction of designs with TM-Score at least 0.7 and pLDDT at least 70, shown for both models across all three templates and all temperature values. For the *T. maritima* encapsulin (top), almost every designed protein sequence is satisfactory when temperature is below 1.0, for both models. When temperature is above 1.0 in this case, the fraction of satisfactory sequences drops, with ESM-IF showing a higher proportion of satisfactory designs than ProteinMPNN. For the *S. elongatus* encapsulin, the fraction of acceptable sequences is never more than 20% across all temperatures for ProteinMPNN, where ESM-IF in contrast shows similar performance to the *T. maritima* encapsulin designs. For the *Q. thermotolerans* encapsulin designs, ProteinMPNN now shows a higher proportion of satisfactory designs than ESM-IF across all temperatures. ESM-IF's fraction of satisfactory sequences falls off more drastically with increasing temperature for this encapsulin compared to the other two, with less than 80% satisfactory sequences even when temperature is set to 0.5.

To summarise, the data presented here demonstrate that both ESM-IF and ProteinMPNN can generate feasible, satisfactory encapsulin sequence designs, as measured by sequence recovery, TM-Score, and pLDDT, for three different template structures. Investigation of both models' behaviour reveals certain "blind spots" with specific template structures or temperature settings.

4.2.3 Markov-Chain Monte Carlo (MCMC) Methods for Encapsulin Design

Next, the performance of two Markov-Chain Monte Carlo (MCMC) protein design methods was investigated for the design of encapsulin sequences. MCMC methods for protein design (Figure 4.2.5) involve iteratively mutating a protein sequence, and then calculating a score for that sequence based on a protein structure. This score is usually a weighted sum of multiple terms, which can be physics-based terms calculated using force fields, or statistical terms based on protein structures and sequences in the PDB [250]. However, with the advent of deep learning models trained on protein sequences and structures, there has been considerable interest in augmenting these scoring functions with probabilities calculated using machine learning tools.

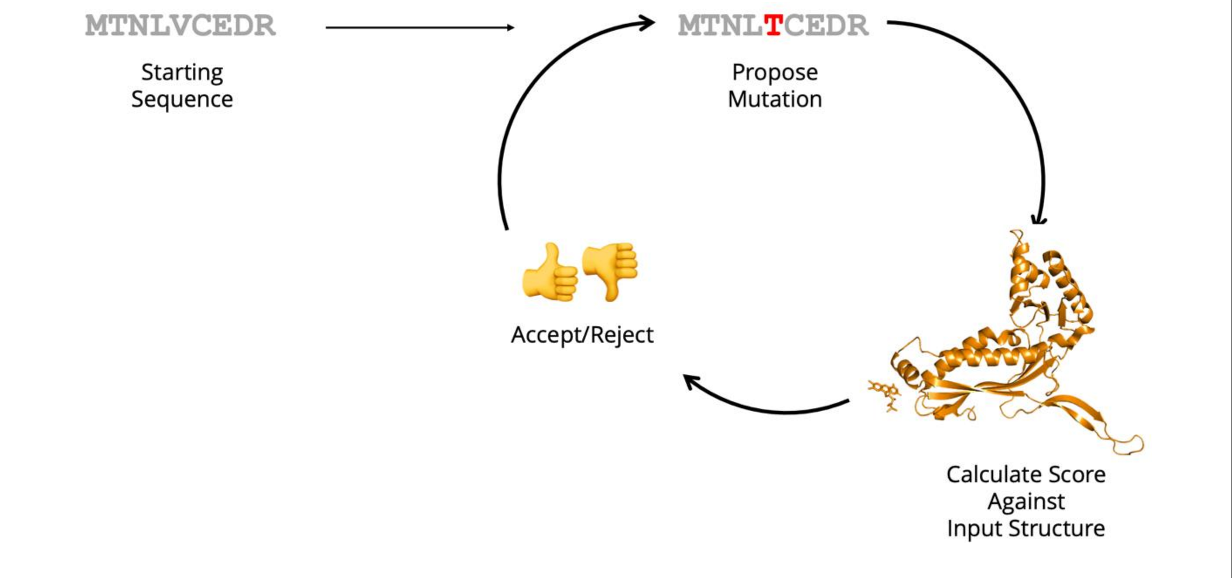


***Figure 4.2.5:* Overview of Markov-Chain Monte Carlo (MCMC) Protein Design**
In MCMC, an initial protein sequence is randomly mutated, and the score of the input sequence evaluated against the design template (a protein structure in this case). This score function can be a physics-based force field, but more recent methods use deep learning models for scoring (see text). Depending on the score of the mutated sequence, the mutation can either be accepted or rejected. The criteria for accepting or rejecting a mutation is often lenient to begin with, before becoming more and more strict over later iterations – this is known as simulated annealing ***[251]*** and helps avoid local minima.

In this work, the performance of two deep learning-based MCMC methods will be investigated; these methods will be referred to as ProteinLM [171] and Protein Programming Language [226]. These two methods follow the regime shown in Figure 4.2.5, but use either the ESM-2 protein language model or ESMFold respectively to calculate the score of the proposed sequence against the input structure.

Figure 4.2.6 shows the result of a successful MCMC protein design experiment. Sequences were designed with ProteinLM for the *T. maritima* T=1 encapsulin monomer structure (PDB **7MU1**) with varying number of iterations. Sequences generated with 80,000 iterations or fewer show low sequence recovery and poor TM-Score pLDDT values, indicating poor design quality and bad structural accuracy compared to the design template. These sequences are also assigned a relatively high score by the model (where lower scores are better), indicating poor quality. However, around 100,000 iterations, a phase transition occurs, and sequences begin to show acceptable TM-Score and pLDDT values. This is accompanied by low model score, and higher sequence recovery against the template. This behaviour makes sense, given that the authors of ProteinLM use 170,000 iterations over ≈10 hours to generate protein designs [171]. Overall, these findings indicate that ProteinLM can generate high quality sequences against the *T. maritima* encapsulin monomer, given enough MCMC iterations.

In contrast, Figure 4.2.7 shows the result of an unsuccessful design experiment using Protein Programming Language. In this experiment, sequences never show TM-Score above 0.3 or pLDDT above 30 for any number of MCMC iterations, and sequence recovery never rises above 5%. These experiments were run with a maximum of 30,000 iterations, which is the setting used by the authors themselves in their protein design experiment. It should be noted that protein design with 30,000 MCMC iterations took over 72 hours, and so it is impractical to experiment with more iteration steps. These findings suggest that the Protein Programming Language method is incapable of generating plausible protein sequences for encapsulin monomer structures.

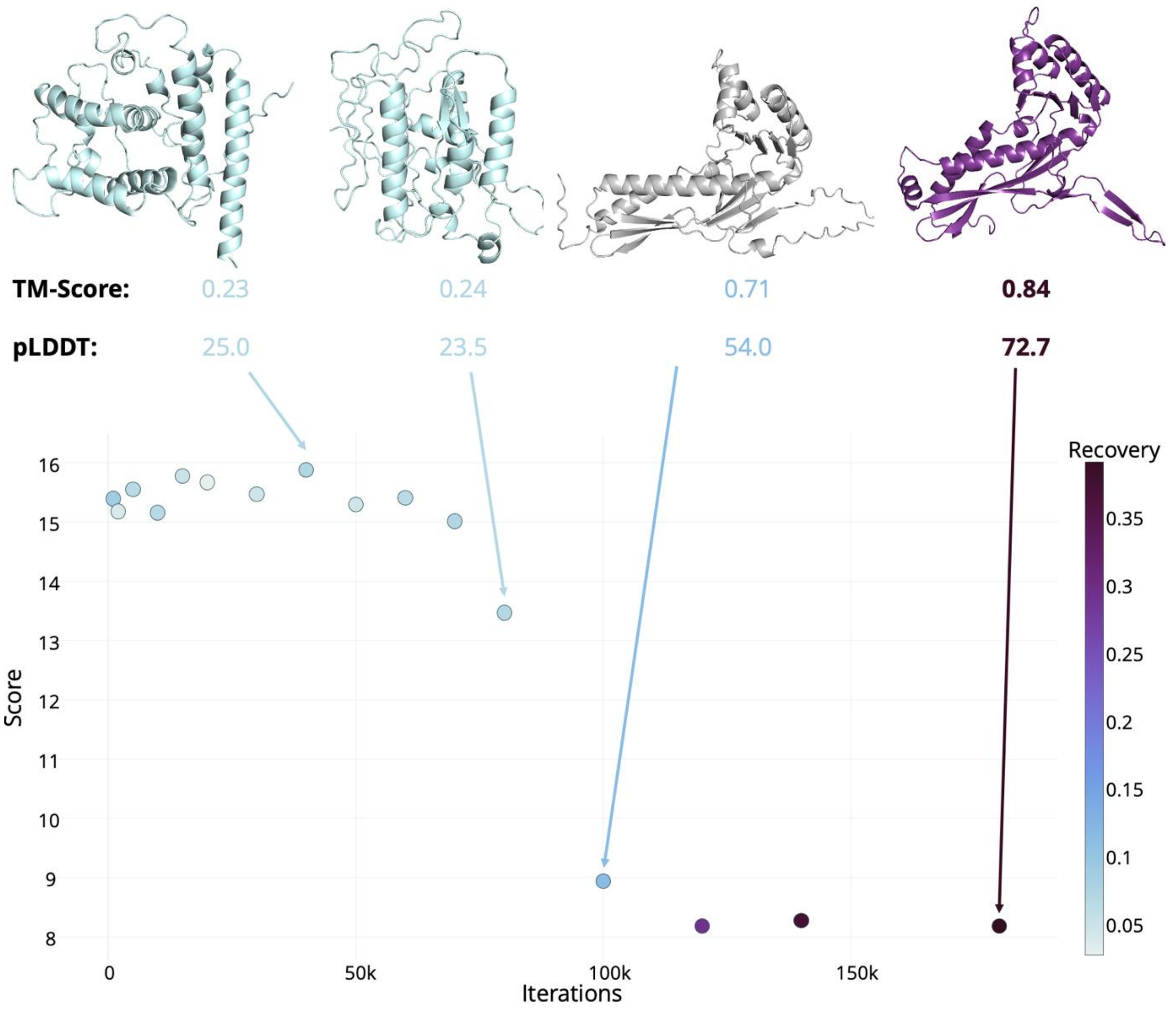


Figure 4.2.6: **MCMC Protein Design with ProteinLM**
Sequences were designed using the ProteinLM method against the *T. maritima* encapsulin monomer structure. Plot showing score assigned to each sequence (y-axis) with varying number of MCMC iterations (x-axis). For selected designs, ESMFold predicted structures are shown with TM-Scores and pLDDT values indicated. Data points coloured by sequence identity to the template (sequence recovery). Below 100,000 iterations, sequences show low recovery (below 10%), and correspondingly low TM-Scores and pLDDT. Around 100,000 iterations a sharp phase transitions occurs, where the model's score of the sequences drops drastically. This is associated with a large increase in sequence recovery (now around 25% or higher), TM-Score, and pLDDT.

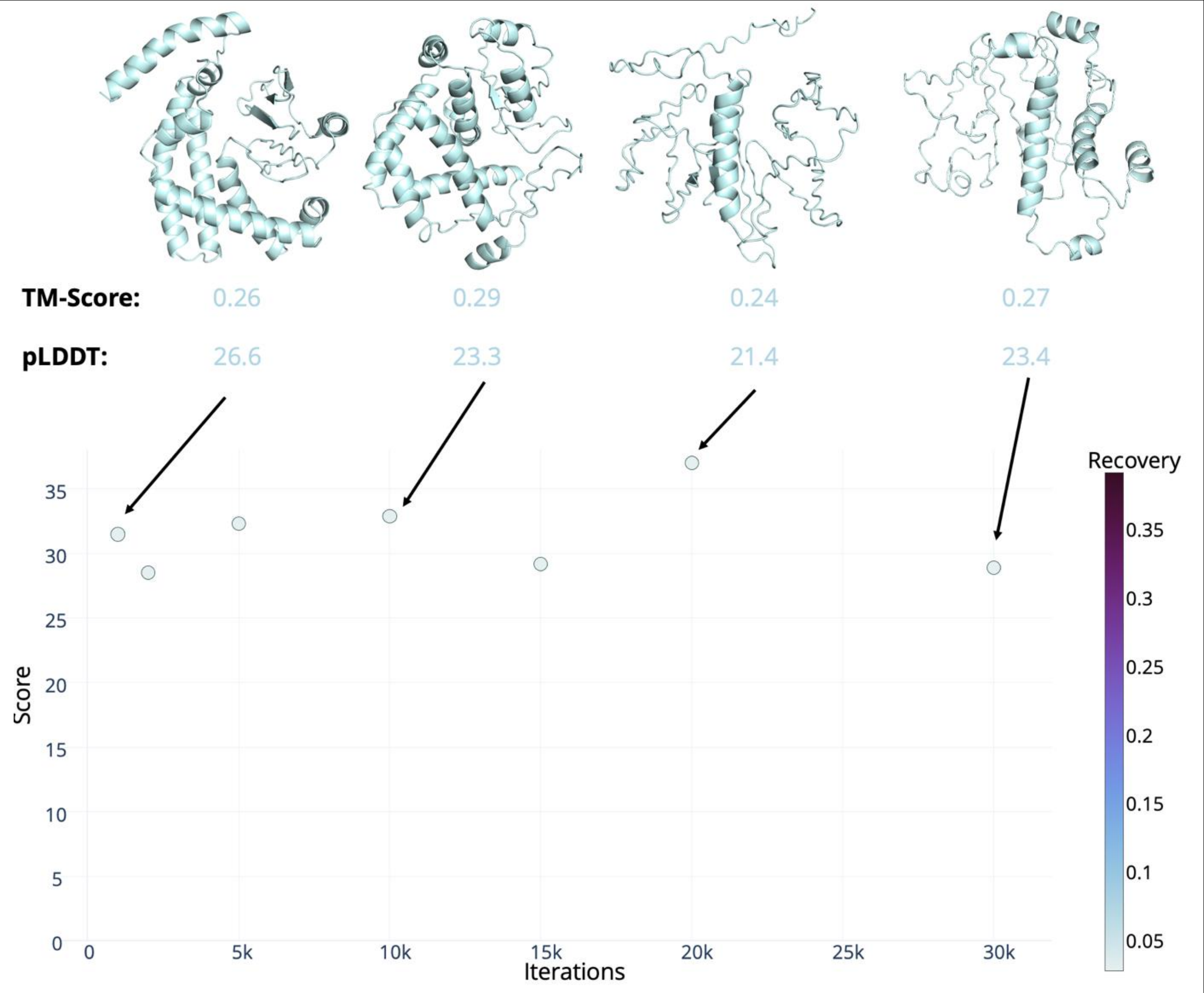


***Figure 4.2.7*: Unsuccessful MCMC Protein Design with Protein Programming Language**
Sequences were designed using the Protein Programming Language method against the *T. maritima* encapsulin monomer structure. Plot showing score assigned to each sequence (y-axis) with varying number of MCMC iterations (x-axis). For selected designs, ESMFold predicted structures are shown with TM-Scores and pLDDT values indicated. Data points coloured by sequence identity to the template (sequence recovery). Protein sequences seemingly never converge on an accurate design, with low recovery, TM-Scores, and pLDDT values even at the 30,000 iterations recommended by the authors.

4.2.4 Protein Language Models (PLMs) for Encapsulin Design

4.2.4.1.1 Background on Generative Language Models

As described in Section 1.5.2.2, there is a large body of work describing the application of natural language models to protein sequences, essentially treating amino acids and proteins as letters, words, and sentences. These protein language models (PLMs) show promising results in various bioinformatics tasks, including sequence generation and protein design as is the focus of this work. Large language models for text or for proteins typically make use of the Transformer architecture. The initial implementation of the Transformer describes a model with two parts, an encoder and a decoder [95]. So-called "generative" language models sometimes have an encoder and a decoder, as in the T5 language model [252] or the ProstT5 PLM [253]. Generative models may also only use a decoder, as in the ubiquitous GPT-3 [97] or the ProGen PLM [164]. In contrast, there is a class of encoder-only language models, which are not usually intended for generative use, but are designed for classification and other understanding tasks. The archetypal encoder-only language model is the original BERT model [254], upon which the first ESM family of PLMs was based. This family of models was the first notable example of Transformer language modelling applied to proteins, and is perhaps the most widespread PLM, chiefly in its application to the ESMFold structure prediction tool [137].

Whilst previous work in the natural language field has focused on the use of BERT as a generative model [255], the capabilities of ESM models for protein design have not been well explored. One previous study investigated two of the first-generation ESM models and their use in generating variants of a chorismate mutase enzyme [256], however this study suffers from a number of limitations. Generated sequences were only subjected to a limited set of validation steps, and structure prediction was not used for validation. Limited optimisation of parameters used in generation was done, and most notably, no experimental validation of these sequences was performed. Additionally, this study looked at ESM-1b models, which have been since superseded by the ESM-2 family. Another previous study [232] looked at generation of 23 different proteins using MSA Transformer, an ESM-like model trained on sequence alignments instead of single sequences [141]. However, generation with MSA Transformer requires an alignment instead of a single sequence, which may be undesirable for design of proteins with few or no homologs.

In this section, the use of ESM-2 family PLMs for protein sequence generation will be systematically explored. Different model sizes and generation methods will be investigated, as well as the fine-tuning of ESM-2 models on specific protein families for enhanced generation.

4.2.4.1.2 Protein Sequence Generation with ESM-2

Protein sequence generation in this work adopts a similar scheme to that used in [232], as described in Figure 4.2.8a. Here, a starting protein sequence has a proportion of its residues randomly replaced with mask tokens. Then, at a given masked position, the language model is used to obtain a probability distribution over all 20 amino acids. This probability distribution can be sampled from, and an amino acid chosen to replace the mask token. This process is repeated until no mask tokens remain and a newly generated sequence is obtained. As such, this is not strictly *de novo* design since a starting sequence is required. This process of masking sequences and estimating probabilities is essentially the same as that used in pre-training the PLM. During pre-training the mask probability is fixed, but for the generation of new sequences it can be freely set and may differ from the training value.

To investigate the optimal masking proportion, the sequence generation performance of the ESM-2 3B parameter model was investigated at varying masking percentages (Figure 4.2.8b). 300 sequences were generated for each masking percentage, using the scheme described in Figure 4.2.8a, with the *T. maritima* T=1 encapsulin as a starting sequence. This sequence was used for all future language modelling experiments unless otherwise stated. As a baseline, sequences were also generated in the same manner, but instead of using a PLM to estimate probabilities and sample amino acids, a random choice of residue was used as a control. Performance was measured by TM-Score of the designed sequences as previously.

The results in Figure 4.2.8b show that when masking percentage is low, ESM-2 3B only gives a modest improvement over a random baseline. This is expected, since changing 5-10% of a protein’s sequence is unlikely to cause large perturbations in the predicted structure, whether these changes are random or estimated using a PLM. Furthermore, at very high masking percentages, the performance of ESM-2 3B is worse than the random baseline. This is again expected, as during pre-training the model is only exposed to protein sequences with up to 15% masking. At higher masking levels, the model clearly lacks sufficient sequence context to accurately predict missing residues. However, at 25% or 50% sequence masking, ESM-2 3B shows a clear performance improvement over a random control. It appears that this “sweet spot” masks too much of the protein sequence for a random choice of residues to rescue the predicted structure, but not too much for the PLM to generate feasible protein sequences. As such, a masking frequency of 0.5 was chosen for subsequent PLM experiments.

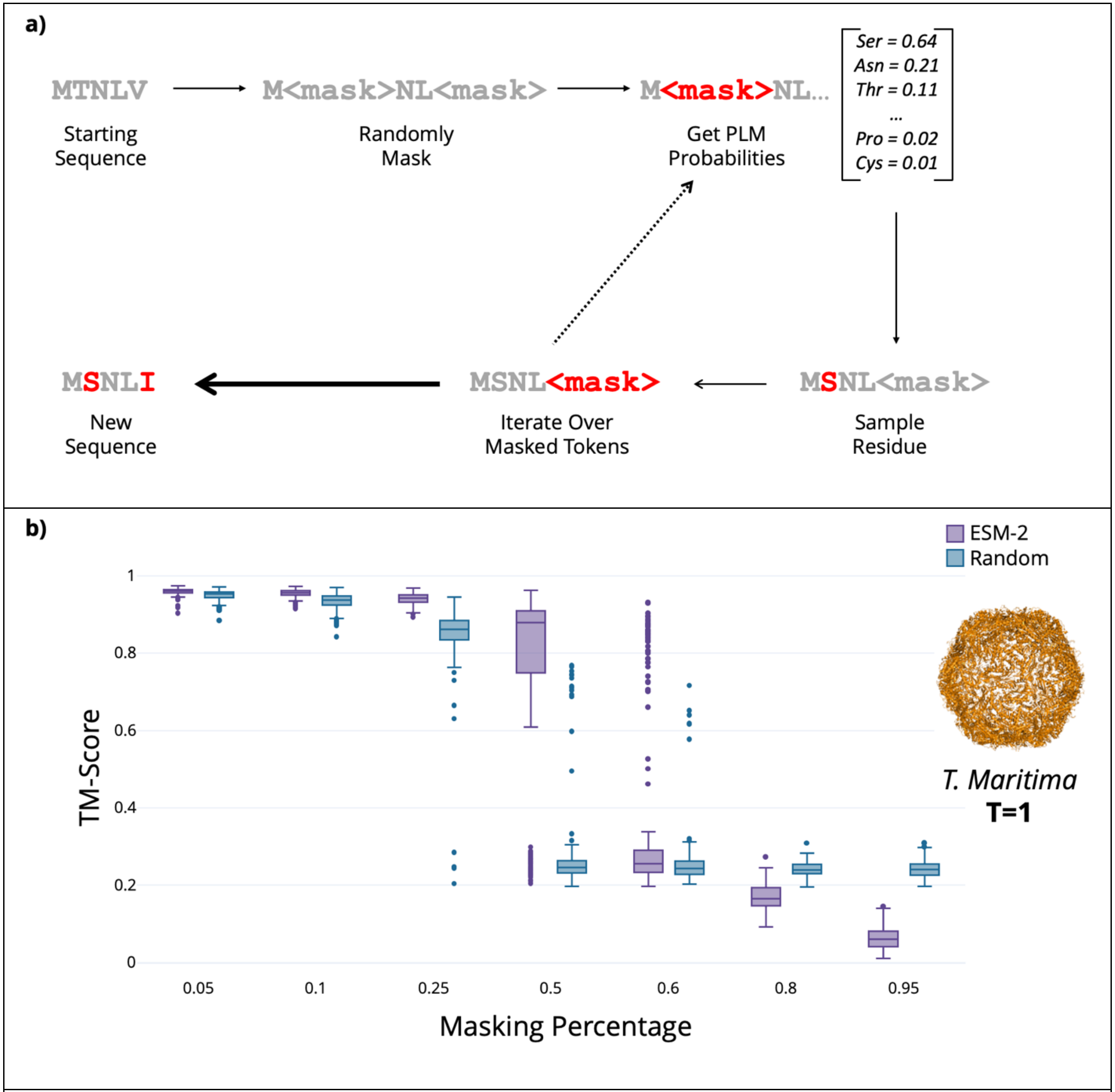


Figure 4.2.8: **Protein Sequence Generation Using Encoder-Only PLMs**
a) Diagram describing the process of sequence generation using encoder-only PLMs. An initial protein sequence has fixed proportion of residues randomly replaced with mask tokens. For each of these mask tokens, a PLM is used to obtain probabilities for each amino acid. An amino acid is sampled from this probability distribution and replaces the mask token. This process iterates over all mask tokens until a newly generated sequence is obtained. **b)** Boxplot (n=200) showing the TM-Scores of *T. maritima* encapsulin sequences generated using the ESM-2 3B model (or a random choice of amino acids) with varying masking percentages. At low masking levels (0.05 and 0.1) ESM-2 shows similar performance to a random baseline, and at high masking levels (0.8 and above) the PLM shows worse performance than random. However, at intermediate masking (0.25 and 0.5) the language model shows greatly improved performance compared to the random baseline.

These initial masking experiments used the ESM-2 3B parameter model with standard temperature-based sampling at T=1.0 (this sampling technique is the same as that used in the ESM-IF and ProteinMPNN models described in Section 2.2.2). The 3B parameter model is the largest that fits in the GPU memory available (with the cloud resources used in this work), however it is possible that models with a smaller memory footprint provide comparable performance. Similarly, changing the sampling method may also improve generation performance over standard temperature-based sampling with T=1.0. As such, the next set of ESM-2 experiments focused on varying either model size or sampling method (Figure 4.2.9a and b respectively).

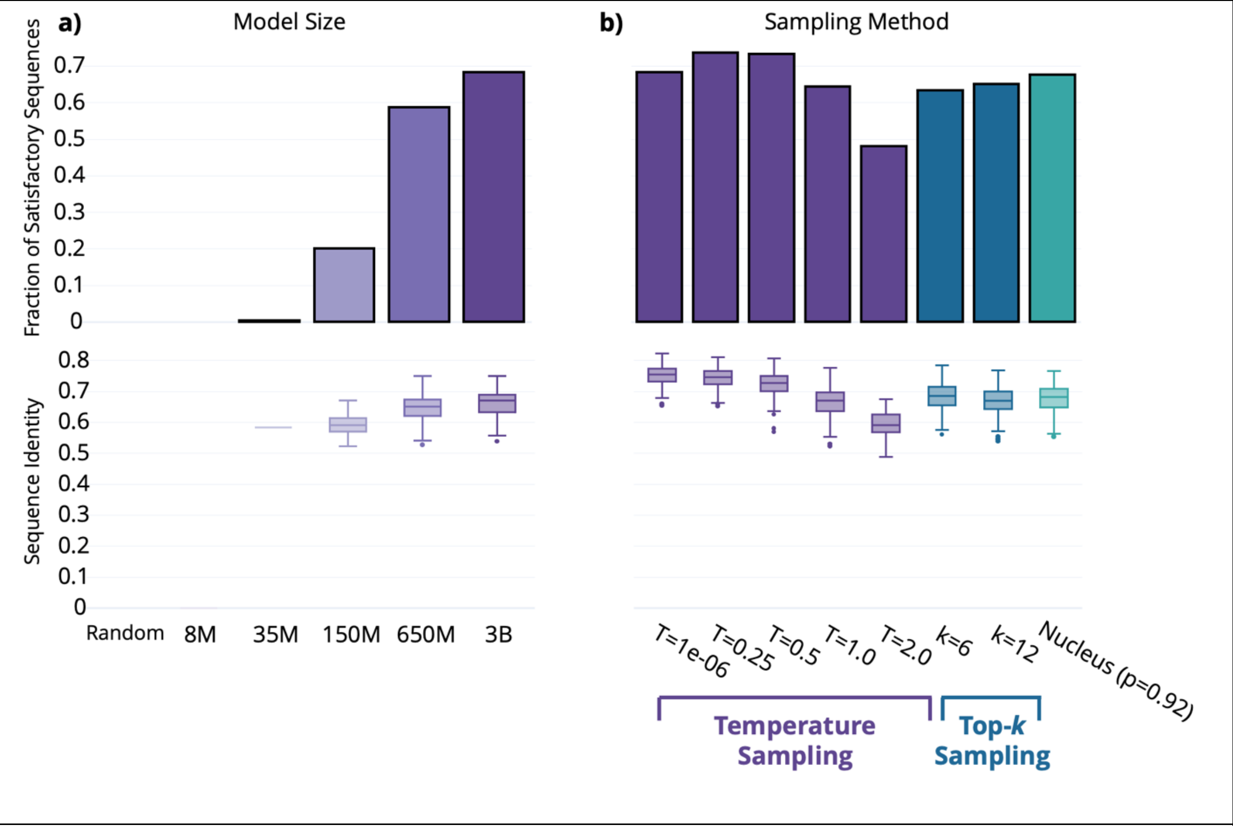


***Figure 4.2.9:* ESM-2 Model Size and Sampling Experiments**
Bar charts showing the fraction of satisfactory sequences generated as in Figure 4.2.4 (top) and boxplots showing the sequence identity distribution against the *T. maritima* template for satisfactory sequences (bottom). All experiments were done using 0.5 masking fraction.
a) Experiments with ESM-2 models of varying number of parameters (n=300). The smallest 8M parameter model failed to generate any feasible sequences, and the 35M parameter model only produced a tiny fraction of viable sequences. However, there was a large jump in performance between 35M and 150M parameters, and an even larger jump from 150M to 650M, as measured both by fraction of satisfactory sequences and by median sequence identity. The 3B parameter model gave a further, smaller increase in performance.
b) Experiments with different sampling techniques using the 3B parameter model. In temperature-based sampling, decreasing the T value below 1.0 gave more viable sequences and a higher median sequence identity, whilst increasing the T value to 2.0 gave the opposite effect. Top-*k* sampling and nucleus sampling (see text for details) appeared to give comparable results to temperature-based sampling with T=1.0, both in the fraction of satisfactory sequences, and in the sequence identity distribution of these satisfactory sequences.

300 sequences were generated with each ESM-2 model size using 0.5 masking frequency, and the fraction of satisfactory sequences reported as in Figure 4.2.4, as well as the

sequence identity distribution of all satisfactory sequences against the *T. maritima* template. 8M and 25M parameter models generate few (if any) feasible protein sequences, but the 150M parameter model shows a sizeable increase in generation performance as measured by feasible sequence fraction, or by median sequence identity. An even bigger increase is seen between 150M and 650M parameters, with a more modest increase seen between 650M and 3B parameters (Figure 4.2.9a). This is broadly in line with the performance profile on UniRef reported in the ESM-2 paper [137] and supports the phenomenon of “emergent properties” in language model understanding as the number of parameters and compute resources used in pre-training increases. It is particularly noteworthy that the 3B model only shows a 10-percentage point increase in feasible sequence generation, and a 2-percentage point increase in median identity over the 650M model, despite using almost 5 times the parameters (and thus 5 times the memory at inference time).

Next, the impact of using different sampling methods was investigated (Figure 4.2.9b). All experiments so far have used temperature-based sampling, but with T set to 1.0, meaning that in effect no temperature was used. In this scheme, amino acids are sampled directly from the probability distribution provided by the PLM with no processing or scaling. As explained in Section 2.2.2, setting T to a value lower or higher than 1.0 can change the shape of this probability distribution and prioritize either diversity or accuracy of generated sequences. The experiments in Figure 4.2.9b bear out this hypothesis, where lower temperature values increase the fraction of feasible sequences but produce sequences with a higher identity against the template. The reverse is seen when T=2.0, with fewer feasible sequences at a lower sequence identity.

Two alternative sampling techniques were also investigated, namely Top-*k* sampling and Nucleus sampling. In these techniques, instead of sampling from the probability distribution of all 20 amino acids, a subset of amino acids are first chosen and then sampled from; in Top-*k* this is the top *k* most likely residues, and in Nucleus sampling this is the top *n* amino acids where the sum of the probabilities of all *n* amino acids adds up to a fixed value (here set as *p=0.92*). Top-*k* sampling was used with two different *k* values, however neither this technique nor Nucleus sampling gave a big improvement in performance over T=1.0 temperature sampling: 64% of sequences generated were feasible with T=1.0

compared to 63%, 65%, and 67% for Top-*k* with k=6 or 12 and Nucleus respectively. Similarly, median sequence identity with T=1.0 was 67%, versus 71%, 67%, and 68% for the three alternative sampling methods. Overall, this standard temperature sampling with T=1.0 was used for all further experiments, since it seemingly provides a good balance between generating feasible sequences, while still providing some diversity in these sequences compared to the starting template.

Fine-tuning is a common technique used in deep learning, whereby a large deep learning model is pre-trained on a large corpus of data, before being further trained on smaller, more specific datasets to provide better performance in a particular task or problem domain. ESM-2 models are already pre-trained on large protein sequence datasets, but fine-tuning on smaller sequence sets has not been well explored in the literature. Here, a dataset of HK97-fold sequences was constructed and used to fine-tune the ESM-2 650M model (see Methods for details on datasets, training, and evaluation) to investigate whether this improves performance on this family of proteins. The model was trained for 10 epochs, however overfitting was seen after 4 epochs, so this checkpoint was used for further experiments (Figure 4.2.10a).

The fine-tuned model was evaluated on an unseen test set comprised of HK97 fold sequences, including sequences of experimentally solved encapsulin and phage capsid protein structures. Perplexity is one of the chief metrics used to evaluate language models, including in the ESM-2 paper, and can be intuitively thought of as the number of different amino acids the model is "choosing" from when replacing a masked residue in a protein sequence; hence a lower perplexity value is better, and an ideal perplexity is 1. The fine-tuned ESM-2 650M model shows a better perplexity on the unseen HK97 test set than the non-fine-tuned model with the same number of parameters but doesn't perform as well as the larger 3B parameter model (Figure 4.2.10b, left). The model was also evaluated on contact prediction on a set of encapsulin and HK97 protein structures and sequences but failed to show improvement on the non-fine-tuned model (Figure 4.2.10b, right).

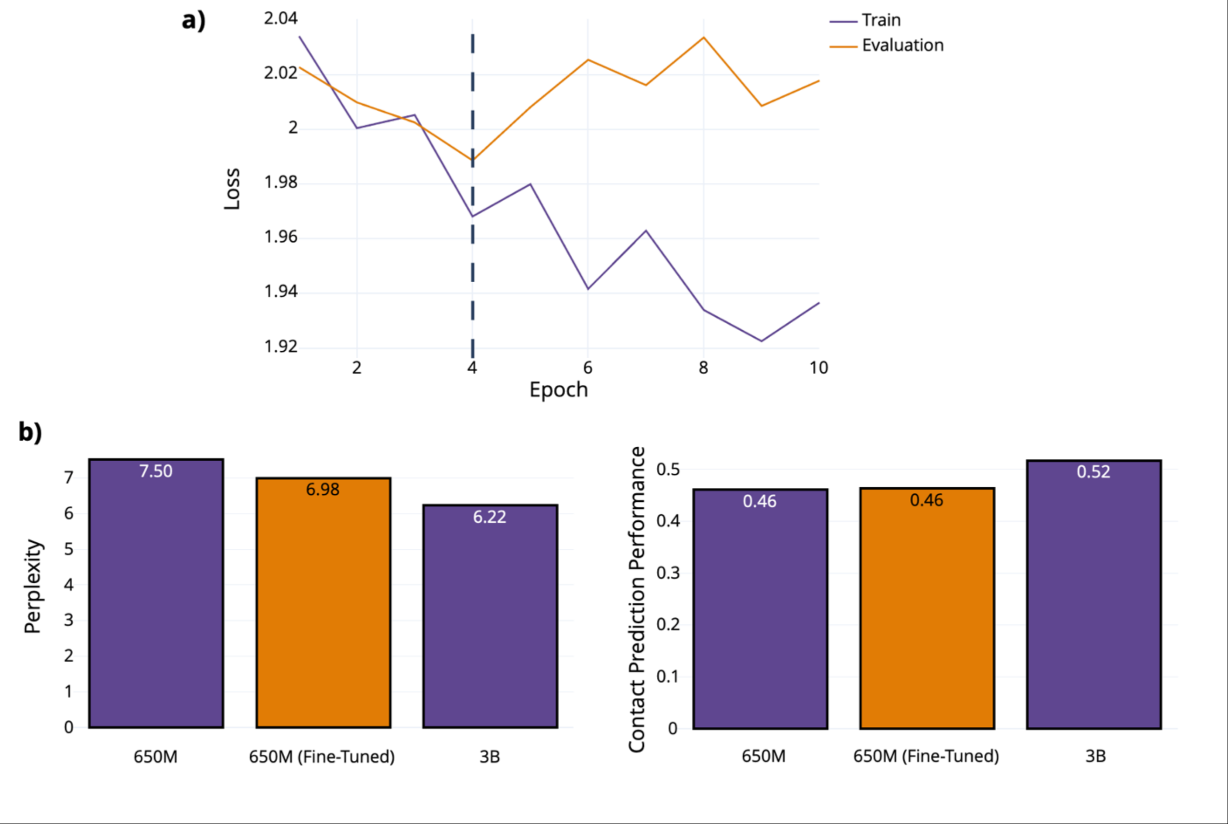


Figure 4.2.10: **ESM-2 650M Fine-Tuning Experiments and Evaluation**
a) Fine-tuning loss as a function of training epoch for the ESM-2 650M model on train (purple) and validation (orange) sets. After epoch 4 (indicated by dashed black line) the model appears to start overfitting as shown by evaluation loss increasing. The fine-tuned model from epoch 4 was used in further experiments. **b)** Perplexity (left) and contact prediction performance (right) for the fine-tuned model compared to the non-fine-tuned 650M and 3B parameter models. The fine-tuned model improves on perplexity compared to the non-fine-tuned model but doesn't perform as well as the 3B model (lower perplexity is better). The fine-tuned model shows the same contact prediction performance as the non-fine-tuned model.

Next, the performance of the fine-tuned model on sequence generation tasks was investigated. As shown in Figure 4.2.11a, the fine-tuned model outperforms not only the non-fine-tuned 650M model, but also the larger 3B parameter model. This is an interesting finding since this larger model has almost 5 times the parameters, as previously mentioned. The performance of this fine-tuned model was also investigated across different sequence templates (Figure 4.2.11b). Performance was comparable for the *T. maritima* and *M. xanthus* encapsulins, and the best performance was seen for the *Escherichia* phage HK97 major capsid protein. Interestingly, poor performance was observed for the *S. elongatus* T=1 encapsulin, with only 6% of generated sequences deemed as satisfactory. This appears to agree with the poor performance on this template seen in the inverse folding models in Section 2.2.2. Overall, these findings seem to support the claim that fine-tuning ESM-2 PLMs can provide performance benefits in sequence generation.

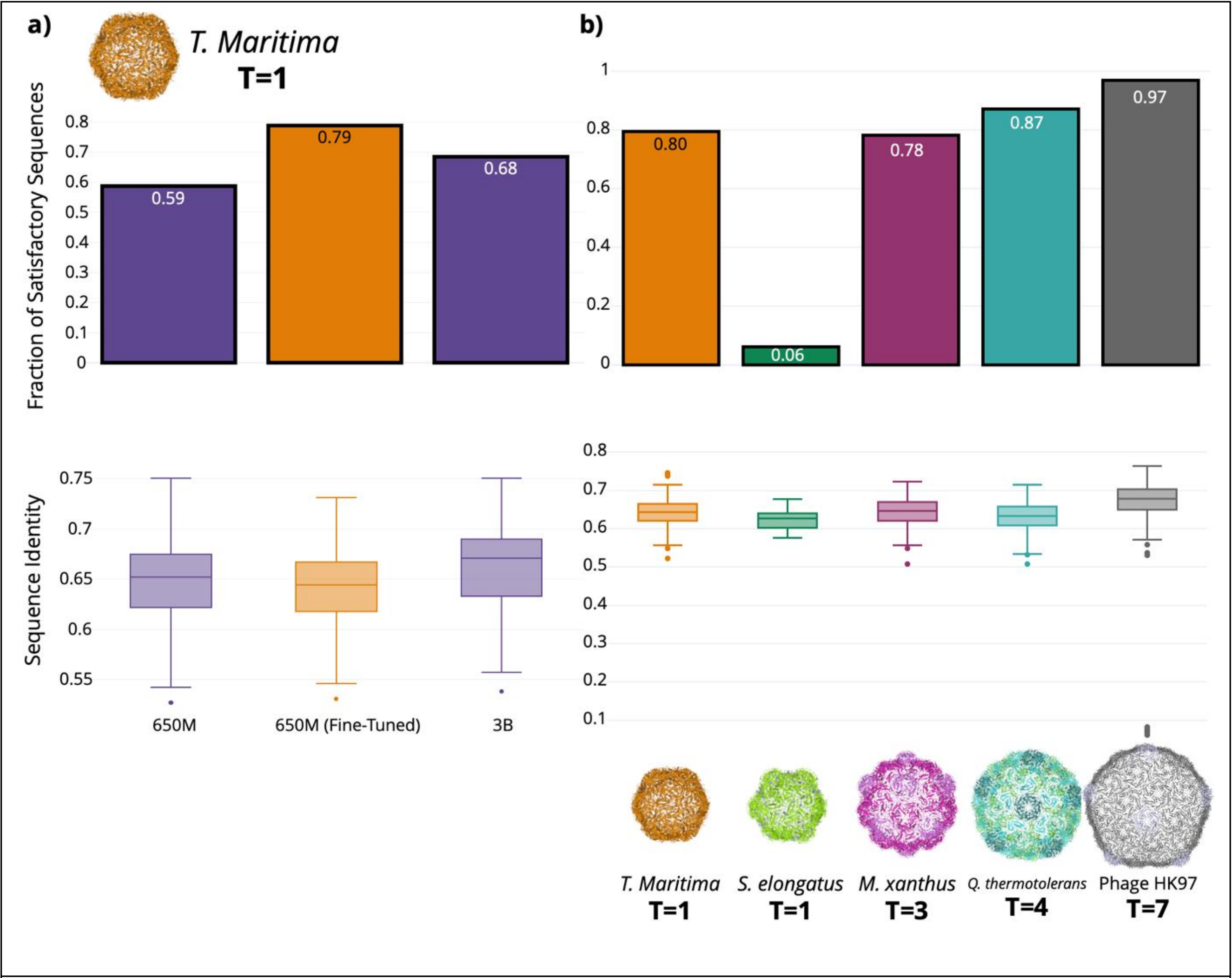


Figure 4.2.11: **Sequence Generation with the fine-tuned ESM-2 650M Model**
Bar plots showing fraction of satisfactory sequences (top) and boxplots showing identity distribution of satisfactory designed sequences (bottom). All experiments are n=300 with 0.5 masking. **a)** Comparison of the fine-tuned 650M model with non-fine-tuned 650M and 3B parameter models on generation for the *T. maritima* T=1 encapsulin. The fine-tuned model shows the best performance as measured by fraction of satisfactory sequences. **b)** Sequence generation performance of the fine-tuned model across various encapsulin and phage capsid templates. The model shows optimal performance when designing sequences for the HK97 phage capsid, but performance is acceptable across all other templates, except the *S. elongatus* T=1 encapsulin, which shows poor generation performance.

4.2.4.1.3 Protein Sequence Generation with Ankh

As described above, ESM-2 is an encoder-only PLM, however there are other model architectures that are commonly used. To this end, the Ankh family of PLMs [257] was investigated, as an example of an encoder-decoder PLM. These models are often used in natural language generation for so-called “sequence to sequence” tasks, such as translating text from one language to another [252]. As with the ESM-2 models, an initial experiment was performed to investigate sequence generation using the *T. maritima* T=1 encapsulin as a template, using the two pre-trained Ankh models – a 450M parameter “Base” model and the larger 1.15B parameter “Large” model.

Ankh can be used to generate sequences using the iterative mask-and-replace scheme as described above with ESM-2 (this will hereafter be referred to as “MLM” generation). However, as an encoder-decoder model, it can also be used to generate sequences “autoregressively”. This is where, instead of a randomly masked protein sequence, a contiguous protein sequence is provided as a prompt, and the model generates a new sequence by adding residues to the C-terminus of this sequence. Both methods were used in the Ankh paper [257], however the performance of these two methods was not compared, and the level of masking used was not systematically explored. As such, both Ankh models were used to generate sequences using either autoregressive or MLM generation, with a variety of masking levels ranging from 5% to 95% of the input sequence. To reiterate, in MLM generation this masking is random and spread throughout the input sequence, whereas in autoregressive modelling residues are masked starting from the end of the input sequence (i.e. the C-terminus).

Figure 4.2.12a compares generation performance in autoregressive vs MLM generation, using the fraction of satisfactory sequences metric defined previously. In MLM generation, neither Ankh model greatly outperforms a random baseline (by more than 5 percentage points). In fact, the random baseline sometimes performs better than both models, as in the experiment with 25% masking. In autoregressive modelling, both models fail to generate more than 15-20 satisfactory sequences with any masking level above 0.05. These data indicate that the pre-trained Ankh models show poor generation performance for encapsulins with the parameters used in this experiment.

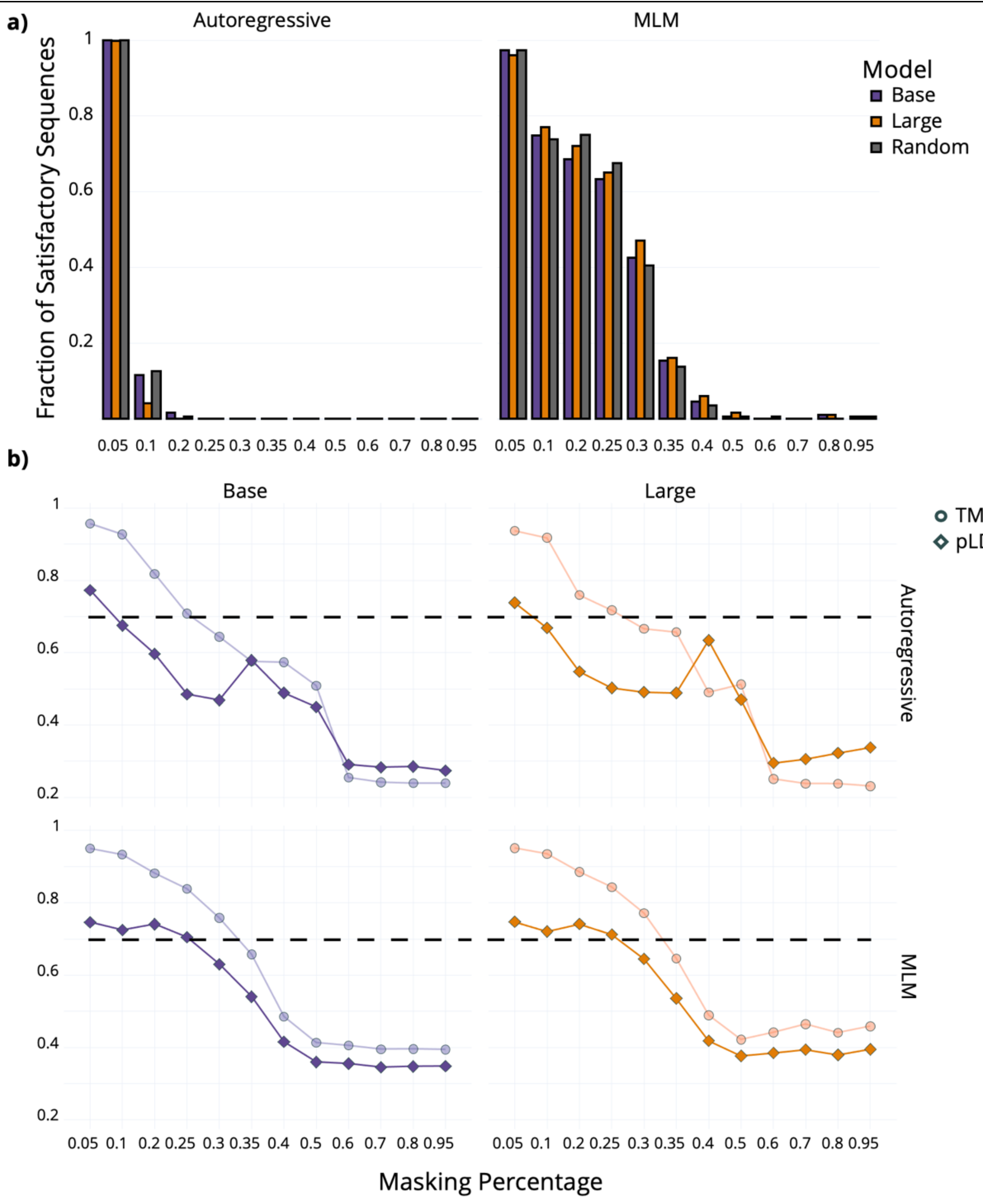


Figure 4.2.12: **Sequence Generation with Ankh PLMs**
a) Bar plot showing fraction of satisfactory sequences (n=200) in either autoregressive (left) or MLM generation, with Ankh Base and Large models and a random baseline, at varying levels of sequence masking. Autoregressive generation shows poor performance above 0.05 masking, where less than 15% of sequences generated are viable. MLM performance appears better, however neither model outperforms the random baseline by more than 5 percentage points at any masking level (apart from 0.3 masking where Ankh Large performs 7 percentage points better). **b)** Line plots showing mean pLDDT (diamonds) and TM-Score (circles) as a function of masking, for both Base and Large models (left and right respectively), in both autoregressive and MLM generation (top and bottom respectively). pLDDT is divided by 100 for presentation on the same axis as TM-Score. Dashed lines indicate the 0.7 cutoff for both TM-Score and pLDDT used to determine "satisfactory" protein sequences. In both generation methods, mean TM-Score starts higher than mean pLDDT, and TM-Score stays above the cutoff at higher masking level than pLDDT. Across both models, pLDDT stays above the 0.7 threshold at a higher masking level for MLM generation than autoregressive generation.

As described above, a “satisfactory” sequence is defined here as a sequence whose ESMFold predicted structure shows pLDDT above 70 and TM-Score above 0.7. To better understand the poor generation performance of Ankh PLMs, the mean TM-Score and pLDDT for each generation experiment was plotted as a function of masking level (Figure 4.2.12b). This provides several insights into the model’s performance. Across both models and both generation types, mean TM-Score is high at low masking levels, and remains higher than pLDDT across all masking experiments. In all four plots, the TM-Score line crosses the threshold at a higher masking level than pLDDT. This indicates that overall, generated sequences tend to fail the satisfactory sequence constraint because of low predicted structure confidence, not low predicted structural similarity to the template fold. Notably, in both models, mean pLDDT appears to fall off much faster in the autoregressive generation experiments (the top pair of plots in Figure 4.2.12b) than in MLM generation (the bottom pair of plots). This indicates that poor confidence of generated sequences is a particular issue in autoregressive generation. Fine-tuning experiments also failed to provide any tangible performance improvements (see Figure 7.1.3, Appendix). As such, Ankh was ruled out as a feasible encapsulin design tool, in favour of the ESM-2 PLMs.

4.3 Establishing an Experimental Encapsulin Characterisation Pipeline

4.3.1 High-Throughput Cloning of Encapsulin Proteins

In a typical protein expression experiment, a DNA sequence encoding a protein of interest is assembled with a bacterial plasmid sequence, containing an origin of replication, antibiotic selection marker, and other regulatory elements required for stable replication and expression of the protein. Classical methods for this DNA assembly step (commonly referred to as “cloning” for historical reasons) involve laborious enzyme digestion, DNA purification, and ligation steps and often require non-trivial troubleshooting and optimisation experiments. Clearly, such classical methods are not suitable to a high throughput setting and are not amenable to automation. This section will describe the use of modern DNA assembly methods and automation to enable high throughput cloning of encapsulin candidate DNA sequences.

The Golden Gate/MoClo DNA assembly protocol based on Type IIS restriction enzymes [258] was used in this work. As shown in Figure 4.3.1a, this method relies on restriction enzymes which cut outside of their recognition sequence, which allows the user to define the nucleotide sequences left at overhang regions. Type IIS DNA assembly provides a myriad of advantages over conventional methods (for a detailed review see [259]) but the main advantage in this use case is convenience. In contrast to other methods, Type IIS assembly only requires the mixing of DNA and enzymes, thermocycling for ≈ 5 hours, and then transformation of the assembly mixture into *E. coli*. As such it is perfectly suited to the parallel cloning of hundreds of different DNA sequences.

Like other DNA assembly methods, Type IIS assembly is often used in conjunction with so-called "blue-white" screening (Figure 4.3.1b). In this system, a plasmid contains the beta-galactosidase alpha fragment gene (LacZ), flanked by Type IIS sites in the backbone. Upon successful cloning, this fragment is replaced by the gene of interest. The DNA assembly mix is transformed into *E. coli* plated on agar plates containing IPTG and the dye molecule X-Gal [260]. If assembly is successful, colonies will appear white in colour after transformation, however if the LacZ fragment is still present (indicating unsuccessful insertion of the DNA fragment), then beta galactosidase will be expressed under the T7 promoter and will process the X-Gal substrate to produce blue colonies.

Cloning and screening a protein of interest requires a suitable expression vector for the host organism (in this case *E. coli*). In this work, the pSB1C3-FB vector (iGEM registry part BBa_K2842666) was used for all cloning and protein expression experiments. This vector is based on the well-characterised iGEM vector pSB1C3, which has been used extensively in the Frank group for overexpression of encapsulin proteins in *E. coli*. pSB1C3-FB is based on pSB1C3 but adds the necessary Type IIS restriction sites and LacZ fragment for cloning. A plasmid map of pSB1C3-FB is shown in Figure 7.1.1 (Appendix).

Physical DNA stocks for pSB1C3-FB vector could not be found, meaning the vector had to be re-synthesised. Forward and reverse primers (binding to the BioBrick suffix and prefix sequences respectively) were used to amplify the pSB1C3 plasmid backbone using inverse PCR. A linear DNA fragment was ordered, containing the LacZ alpha fragment insert, Type-IIS sites, and 20 bp overhangs matching the amplified backbone region. These two linear DNA fragments were assembled using Gibson Assembly, transformed into *E. coli* DH5α, and purified plasmid DNA isolated. The purified pSB1C3-FB vector stocks were verified using restriction digestion and Sanger sequencing, and testing with a candidate encapsulin

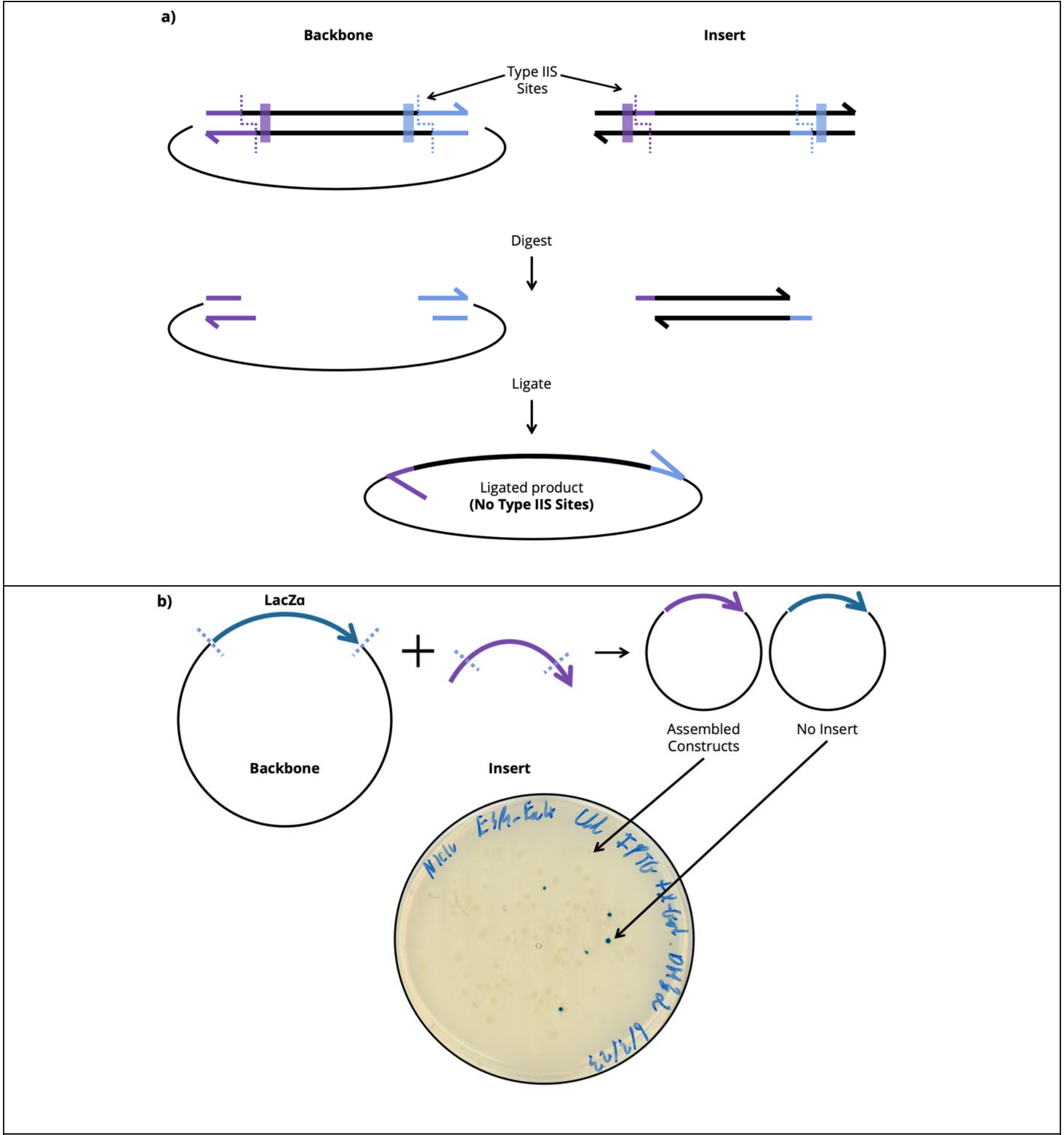


Figure 4.3.1: **DNA Assembly using Type IIS Restriction Enzymes**
a) Type IIS restriction enzymes cut (sites shown with purple/blue dashed lines) outside their recognition sequences (shown as rectangles). This allows assembly of vector backbones (left) and inserts (right) containing outward- or inward-facing Type IIS sites respectively, so long as the overhang sequences are designed appropriately. Following digestion and ligation (which occur in cycles in the same reaction), the assembled product has no remaining Type IIS sites and so cannot be cut and re-ligated, which reduces the number of empty background vector DNA in the reaction over time. **b)** "Blue-white" screening is a commonly used selection method used with DNA assembly methods, including Type IIS. In this system, a beta-galactosidase (LacZ) fragment is included between the restriction sites in the backbone. Successful assembly of an insert will cause this fragment to be excised, while empty background colonies will still harbour the LacZ gene. The presence or absence of this gene (and therefore assembly) can be screened by plating transformants on agar plates containing IPTG and the dye molecule X-Gal.

protein sequence confirmed that Type IIS assembly yielded plasmids with the correct sequence.

Next, an automated Type-IIs protocol was established and optimised using the Opentron OT-2 robot. Type IIS assembly was tested using eight control sequences – designed encapsulin sequences against the *T. maritima* encapsulin (PDB 7MU1) using ProteinMPNN and ESM-IF (four sequences each). These protein sequences were not subjected to rigorous computational screening or selection but were chosen simply as test sequences for DNA assembly. 6/8 of these control sequences were successfully manually cloned using Type IIS assembly and verified by restriction digest (Figure 7.1.2, Appendix).

Figure 4.3.2a shows an overview of the experimental steps required in Type IIS assembly. The OT-2 robot has the capability of performing all these steps automatically, however attempting to automate the entire process failed to give any colonies for any of the controls (a detailed log of all OT-2 optimization experiments is shown in Table 7.4, Appendix). This indicated that the issue lies in thermocycling, transformation, or plating. To investigate the latter, a Type IIS reaction was set up manually and transformed into *E. coli*, and varying amounts and dilutions were manually plated onto the same agar plates used in the OT-2 (Figure 4.3.2b). Several dilutions and amounts yielded colonies, indicating that plating using the OT-2 is possible.

As such, it was hypothesised that the issue lies in the OT-2's thermocycler, or the temperature module used for heat shock and outgrowth. To investigate this, a "semi-automated" procedure was used. Here, the robot was used to mix up Type IIS reactions and add a layer of silicone oil to prevent evaporation. Plates were then manually sealed with foil seals and placed in a normal benchtop thermocycler for assembly. Following this, plates were returned to the OT-2 and reactions mixed with *E. coli* competent cells. Cells were heat shocked in a water bath and outgrowth performed in a benchtop incubator, as in conventional manual cloning. Here, the OT-2 was only used to add outgrowth medium, and to plate transformation cultures onto agar plates. This protocol yielded a single successful colony when tested with the control inserts (Figure 4.3.2c).

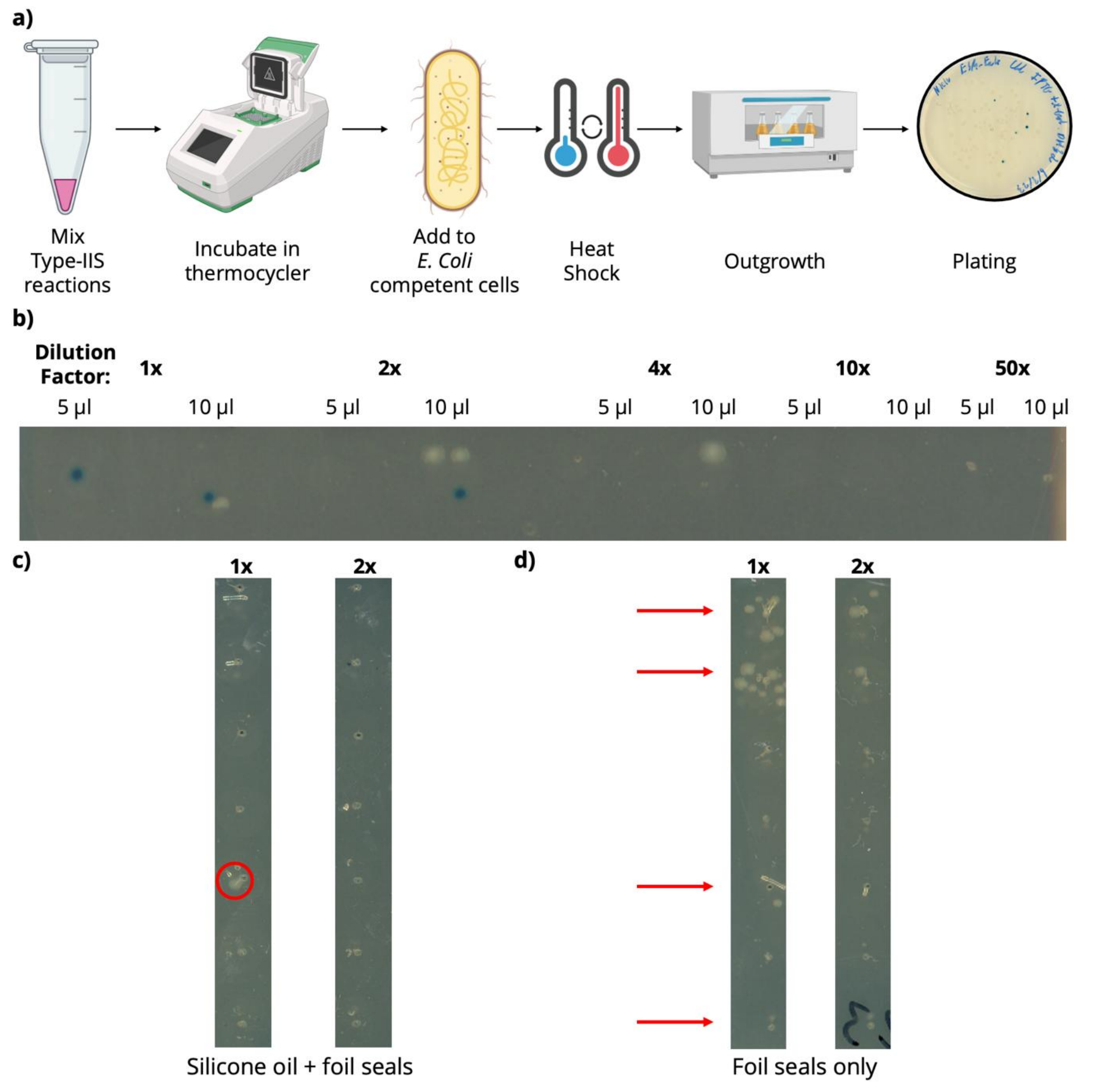


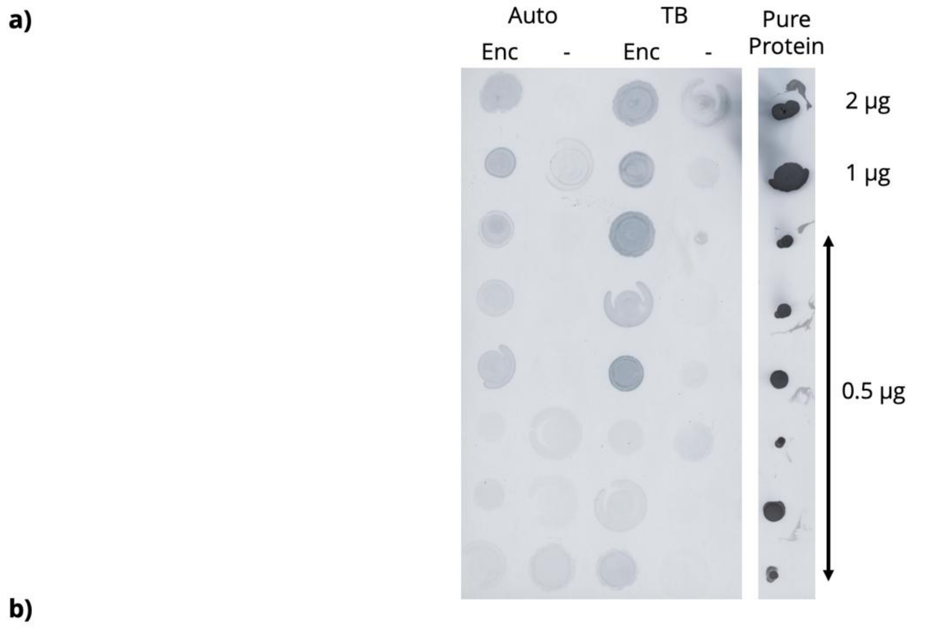
***Figure 4.3.2:* Optimizing Type IIS Assembly with the Opentron OT-2**
a) Overview of the Type IIS protocol. Reactions mixtures are set up and incubated in a thermocycler. Assembly mixtures are added to competent *E. coli* cells and heat shocked, before medium is added and cells are grown at 37 ˚C with shaking. Finally, cell mixtures are plated on agar plates. No colonies were seen when attempting to automate this entire protocol using the OT-2 robot. **b)** Testing Type-IIs assembly manually with a known control insert yields colonies at varying plating densities. From 200 µl of outgrowth culture, varying amounts and dilution factors were tested. 10 µl of culture at 1x (undiluted) and 2x dilutions was used in subsequent experiments. **c)** Type-IIs assembly of 8 different insert sequences under a "semi-automated" scheme, where the robot only carries out key protocol steps (see text for details). Reaction plates are covered with silicone oil and sealed with foil seals before thermocycling. This protocol only yielded a single colony for one assembly reaction. **d)** Removing the silicone oil step and only sealing plates with foil seals gives multiple colonies with the same assemblies shown in **c)**.

However, the addition of silicone oil in this protocol added significant complexity to the OT-2 protocol, requiring custom tuning of pipetting rates and movement speeds, and despite this tuning it was observed that adding and mixing oil with the reaction mixtures introduced air bubbles and caused spillages. Since plates were manually sealed with foil seals afterwards, it was hypothesised that the silicone oil could be omitted entirely, and Figure 4.3.2d shows that with this step removed, 4 out of 8 trial assemblies gave colonies. Further test runs using purified assembled plasmid DNA also yielded satisfactory numbers of colonies following optimization (see Table 7.4, Appendix). As such, this optimised protocol was used for encapsulin design cloning in subsequent experiments.

4.3.2 Solubility Screening of Encapsulin Proteins

Next, a scalable method for assessing the soluble yield of encapsulin proteins was developed. In a low throughput setting, protein soluble yield is assessed following expression in *E. coli*, from cultures in tube or flask scale. Expression is usually verified using SDS-PAGE or western blots, and solubility can be interrogated by comparing the soluble and insoluble fractions of the *E. coli* cell lysate. This approach presents several problems when screening many proteins in parallel. First, tubes or flasks are difficult to handle in large quantities (especially when biological replicates are required), and SDS-PAGE is laborious when screening many samples in parallel. Furthermore, the insoluble fraction of *E. coli* cell lysates can be viscous and difficult to handle, especially when running SDS-PAGE gels.

To streamline solubility screening of encapsulin proteins, a protocol was developed based on *E. coli* expression in 96-well plates, followed by solubility screening using a dot blot against the Strep Tag II. Across many experiments, several different variables relating to expression or immunodetection were optimised; these include cell culture volume and medium type, shaking speed, membrane blocking, and experiments on denaturation or clarification of cell lysates. Figure 4.3.3a shows the results of a successful dot blot with the final optimised protocol. Here, 250 µl *E. coli* cultures are grown with high shaking, harvested and resuspended in 25 µl of BugBuster cell lysis reagent. Cell lysates are clarified by filtration before 5 µl of lysate is pipetted onto the membrane for immunodetection (see Methods for full details). In this experiment, two different induction methods and media were tested, either autoinducing medium or Terrific Broth with manual IPTG induction after 3 hours. Both methods appeared to give good signal for the encapsulin cultures, and lower signal for the negative controls.



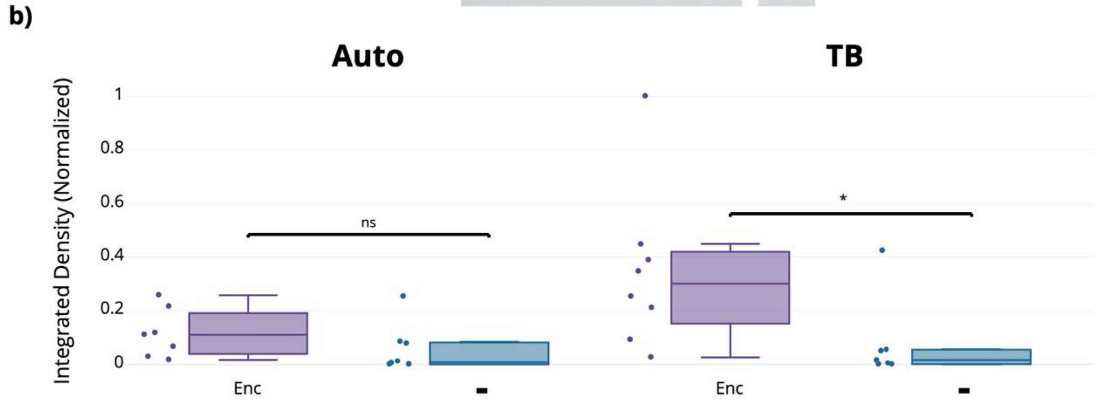


***Figure 4.3.3*: An Optimised Dot Blot Protocol for Encapsulin Solubility Screening**
a) Scanned image of a nitrocellulose membrane showing a dot blot against Strep Tag II, for either the wild-type *T. maritima* encapsulin (Enc) or an empty plasmid as negative control (-). Two different induction methods were tested: autoinduction medium (Auto) or Terrific Broth with manual induction using IPTG (TB). Purified TmEncap protein was also loaded onto the membrane as a loading control. Each spot is a different biological replicate. **b)** Boxplots showing the quantified integrated density of the dot blot spots after image processing in ImageJ. Density values are background subtracted and normalised by the OD_{600} of each sample. Distributions of the density values of TmEncap and empty plasmid populations were compared using Welch's *t*-test. There was no significant difference between Enc and **–** densities with autoinduction medium, but a significant difference was observed for TB samples ($p < 0.05$).

However, following image processing and quantification of the normalised density of the dots in ImageJ, it appears that there is no significant difference between the densities for the encapsulin protein and the negative control using autoinduction medium. The TB samples induced by IPTG showed a higher mean dot density, which was statistically significant. As such, this experiment produced a final optimised dot blot protocol using TB medium with manual induction by IPTG.

4.4 Discussion

4.4.1 Deep Learning Tools for Encapsulin Design

4.4.1.1.1 Structure Prediction Benchmarking

In this chapter, several different deep learning tools for protein design were investigated, and their performance characterised on encapsulin proteins as a use case. As a foundation for this work, the performance of three different protein structure prediction tools was benchmarked on a representative set of HK97-fold protein structures. AlphaFold2 shows the best performance on this dataset, at the cost of prohibitively long run times. ESMFold shows more than acceptable performance with runtimes an order of magnitude shorter than AlphaFold2, and so this model was used in subsequent experiments.

4.4.1.1.2 Inverse Folding Models

The first set of experiments focused on ProteinMPNN and ESM-IF, two so-called "inverse folding" models which generated protein sequences conditioned on a structural backbone provided as input. Both models appeared to generate feasible protein sequences (as measured by TM-Score and pLDDT of predicted structures) across a selection of encapsulin structure templates. Interestingly, ProteinMPNN provides comparable performance to ESM-IF, despite having ≈100 times fewer parameters and being trained on a vastly smaller dataset (≈25,000 clustered PDB structures for ProteinMPNN, versus 12 million AlphaFold2 predicted structures plus PDB structures for ESM-IF). However, it is unclear whether these differences will result in any real-world performance differences until sequence designs are experimentally characterised.

4.4.1.1.3 Monte-Carlo Methods

Next, two Monte Carlo-based protein design methods were tested, namely ProteinLM and the Protein Programming Language. ProteinLM generated feasible protein sequences after a sufficient number of Monte Carlo iterations, over the course of several hours. However, the Programming Language model failed to generate any feasible sequences even after hours of Monte Carlo iterations. There are several potential causes for this poor performance on encapsulin proteins. The examples presented in the Programming Language preprint are all small, single domain proteins, which are a much easier design challenge conceptually than an HK97-fold protein monomer. The HK97 fold is formed from three different domains with flexible regions, and notably several beta sheets, which are known to be more difficult to design than helical regions due to their more complex hydrogen bonding patterns and longer-range interactions.

Alternatively, the parameters used for the energy function weightings were left on their default values, and so could have been configured incorrectly. This could be a potential avenue to explore in future work, however it may not be necessary when ProteinLM provides acceptable performance in Monte Carlo-based design. Recall that both models use Markov Chain Monte Carlo to optimise a random starting protein sequence, but ProteinLM uses scoring based on ESM-2 PLMs, where Programming Language uses ESMFold. Given that sequence designs are subsequently screened using ESMFold in this work regardless of design method, it appears that the Programming Language method is not worth investigating any further.

4.4.1.1.4 Protein Language Models

In this chapter, the performance of two different PLMs in encapsulin design was investigated. ESM-2 encoder only PLMs were shown to give acceptable performance in encapsulin design across different templates, masking percentages, and model sizes. Different sampling methods didn't appear to have a great impact on generation performance. Initial experiments suggested that masking 50% of the input sequence provided a "sweet spot" where the model had enough context to generate plausible protein sequences, but allowing for some variation to be generated relative to the starting sequence. It should be noted that masking in the generation algorithm used here is stochastic and based on a binomial distribution (see Methods), and so not every generated sequence has the same number of residues designed by the PLM.

Next, it was demonstrated that fine-tuning an ESM-2 PLM on a curated dataset of HK97-fold protein sequences can provide performance benefits for sequence generation. Fine-tuning did not provide any performance benefits for contact generation, but did boost sequence generation performance, to the point where a fine-tuned 650M parameter model outperformed a much larger 3B parameter model. This fine-tuned model was also able to generate satisfactory sequences across a range of encapsulin templates, apart from the *S. elongatus* T=1 encapsulin (PDB 6X8M). It is interesting to note that slightly inferior performance on this template was observed with ProteinMPNN and ESM-IF compared to other encapsulin templates. There are two possible explanations for this behaviour. On the one hand, this template might be a difficult challenge for deep learning-based protein design tools. Alternatively, given that all designs were validated using ESMFold, it is also possible that this protein and its variants are a "blind spot" for the structure prediction tool used to validate designs, rather than the design tools themselves.

Next, the performance of the two Ankh PLMs was investigated. Overall, these models showed poor performance in encapsulin generation, both using autoregressive or MLM generation. Autoregressive modelling, sometimes called causal language modelling, is often used in natural language models (for example the GPT family of LLMs). In autoregressive modelling, sequences of words or characters are generated from "left to right" given a starting prompt. This makes intuitive sense for languages like English where sentences are constructed and read in this way. However, this modelling scheme makes less sense for proteins, where the order in which domains or secondary structure elements appear in the sequence can sometimes have little effect on structure or function; consider for example circularly permuted proteins, where the order of domains can be swapped with no impact on function. Nevertheless, Ankh models showed poor performance in sequence generation in either the autoregressive or MLM-based schemes, even with fine-tuning.

Lastly, it may be worth considering whether the "fraction of satisfactory sequences" metric used in this work is a good measure of generation performance. It could be argued that the number of feasible protein sequences a model can generate is irrelevant, as long as it can generate at least one attractive candidate. However, if a model generates very few feasible sequences and many unsuitable sequences, then this requires lots and lots of generation experiments and computational screening, which is not efficient or sustainable with regards to energy and compute resources. Furthermore, the "satisfactory" sequences in this work are delineated as such by acceptable TM-Score and pLDDT. This is intended to be a starting point for further in-depth screening specific to a certain use case. Aside from TM-Score and pLDDT, there are many other metrics and screening criteria that can be considered, for example the presence of specific structural or sequence motifs, predicted solubility or thermal stability, or the suitability of DNA sequences.

4.4.2 Experimental Screening Methods for Encapsulin Design

Next, a set of automated screening methods were developed for the cloning, expression, and screening of encapsulin proteins. First, a type IIs DNA assembly protocol using an Opentron

OT-2 robot was established and optimised. Whilst the robot has the capability to carry out all steps of the protocol from start to finish, in testing it was observed that some steps had to be carried out using conventional lab equipment. For example, when transforming DNA assembly mix into *E. coli*, the heat shock step was carried out in a water bath instead of the OT-2 Temperature Module. It is likely that the Temperature Module cannot heat up and cool down fast enough to provide a heat shock to *E. coli* cells, compared to incubating the plate on ice and transferring to and from a 42 ˚C water bath. This Module also has no shaking capability, and so 37 ˚C outgrowth was performed in a standard benchtop incubator. In future, an Opentron Heater-Shaker Module could potentially be used to automate both steps, towards a fully automated protocol for cloning and transformation.

Finally, a pipeline was developed for expression and solubility screening of encapsulin proteins in *E. coli* in 96-well plates. Many variables were investigated and optimised, including cell culture volumes, shaking speeds, temperatures, and downstream processing methods - for example, using clarified lysates versus crude whole lysates, adding a denaturing step, and resuspending frozen cell pellets in different volumes. Eventually, an optimal protocol was established based on growth of 250 µl *E. coli* in TB medium, and manually inducing protein expression with 1 mM IPTG after 3 hours. It should be noted that the final, optimised protocol only investigates encapsulin soluble yield under a single experimental condition (comprising temperature, *E. coli* strain, medium type, induction etc). However, the aim of this screen is to detect candidate proteins with increased soluble yield compared to a wild-type protein, and since it is already known that the wild-type shows good yield under this condition, this single condition should be sufficient to screen for these candidates.

After this work was completed, a preprint was made available detailing an automated protocol for *E. coli* transformation and protein expression using the Opentron OT-2 robot [261]. However, this method does not deal with DNA assembly steps in any way and starts from purified plasmid stocks. Crucially, this method does not allow for any kind of solubility screening of expressed proteins and relies on low-throughput SDS-PAGE readouts, or plate reader fluorescence measurements for fluorescent proteins (which is not applicable here). As such, the automated method detailed in this work still has some key differences with this concurrently released work.

The expression and dot blot screening protocol does have some limitations, namely low sensitivity, and readouts only being semi-quantitative. There is also variability between samples in the dot blot assay, potentially arising from the manual pipetting of samples onto a membrane. This could be alleviated in future using a vacuum dot blot apparatus. Manual induction after 3 hours may also introduce some variability in protein expression since cultures may be at different stages of growth. However, with adequate biological replicates it is likely that this variability can be averaged out somewhat. Overall, despite these limitations, the dot blot protocol does appear to give satisfactory results when comparing wild-type encapsulin with a control, and so it should work well enough to spot promising candidates with higher soluble yield.

4.4.3 Future Work

The computational design and experimental screening methods described in this chapter lead to one obvious next step, which is their application to the design of new encapsulin proteins. This will be described in the following chapter. However, there are several other interesting avenues which could be investigated in follow up work. On the computational side, there are many protein design methods which were not tested here, the most exciting of which are diffusion models such as RFDiffusion [165] and Chroma [172]. These models could be useful in designing new encapsulins in future and merit further investigation. Furthermore, all computational design presented here took place in a "monomeric" context, that is, designing new sequences given a structure of template sequence of an encapsulin monomer. However, in a true biological context encapsulin proteins are large icosahedral complexes, and so *in silico* design methods should take this into account if possible. This could be done using multichain design and scoring methods (possible with ProteinMPNN and ESM-IF) or using physics-based docking and scoring methods such as those available in the Rosetta software package.

On the experimental side, the "semi-automated" cloning and solubility screening protocol shown here works well to parallelize and automate time-consuming steps during characterization. However, the method is not fully automated and still requires manual intervention and supervision. Future work could attempt to fully automate the Type IIS assembly thermocycling and bacterial transformation/growth steps in the protocol, which would be possible on an upgraded Opentron robot with more available modules.

Chapter 5: *In silico* Design and Experimental Validation of Novel Encapsulin Proteins

5.1 Background

The previous chapter described preliminary work on computational and experimental design and screening of encapsulin proteins. In this chapter, learnings from these design and screening experiments will be applied to the design of novel encapsulins. The primary aim of this chapter is to produce novel variants of the *T. maritima* T=1 encapsulin protein, with increased soluble yield. Deep learning tools for protein design will be used to generate thousands of candidates, which will be computationally screened to produce a small final set of candidate proteins. These candidates will be experimentally validated using the cloning and expression pipeline outlined in the previous chapter.

Alongside this work, the assembly of encapsulin proteins will also be investigated. Previous literature has hypothesised that the E-loop region is a major determinant of encapsulin T-number and assembly dynamics, but this has not been experimentally demonstrated. To this end, a pair of chimeric encapsulin variants with modified E-loops will be designed and included in the experimental pipeline with the solubility designs. One chimeric encapsulin variant with altered assembly characteristics will be characterised in more detail.

5.2 Design of Novel Encapsulin Variants with Improved Soluble Yield

5.2.1 Computational Encapsulin Design and Screening

When changing the sequence of a protein to make it more soluble, care must be taken not to disturb function by changing important residues. Indeed, as described in [211], redesigning wild-type proteins can be viewed as a trade-off between preserving natural function and increasing solubility. As such, starting from the *T. maritima* T=1 encapsulin sequence or structure, several residue positions were fixed during design. Residues involved in flavin binding, capsid assembly, and cargo loading were manually chosen based on crystal structures and experimental data (both in the literature and from previous Frank Lab experiments). Additionally, a set of residues were chosen automatically based on an MSA as in [211]. The *T. maritima* T=1 encapsulin sequence (UniProt accession Q9WZP2) was used to search UniRef90 using mmseqs2. All 430 hits returned showed E-values below 10^{-20} and fewer than 3 gap openings and were used to build an MSA using MUSCLE. At each position in the alignment, the frequency of each amino acid was calculated, and positions in the sequence were ranked by the frequency of the most common amino acid. The top 30% of these most conserved positions were chosen as fixed residues for the design process, making a total of 96 fixed amino acid positions (out of a total of 263 residues in the encapsulin sequence).

Supplementary Tables

Table 7.1 (Appendix) shows a detailed breakdown of fixed residue positions. Figure 5.2.1a shows the full design pipeline. Following fixed residue selection, four different models were chosen for encapsulin design, based on the previous chapter's findings. Two inverse folding models (ProteinMPNN and ESM-IF), one protein language model (ESM-2, both "vanilla" and fine-tuned variants) and one Monte-Carlo based method (ProteinLM) were used. For the inverse folding models, sequences were generated with six different temperature values for sampling: 10^{-6}, 0.1, 0.2, 0.3, 0.4, and 0.5, with 200 sequences per temperature value. 40 sequences were generated with ProteinLM using the default "fixedbb" protocol and 120,000 iterations as in previous experiments. For both ESM-2 models (vanilla and fine-tuned), sequences were generated with 9 different masking amounts based on previous experiments: 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, and 200 sequences per masking amount.

Following generation, designs were first subjected to a set of simple sequence-based filters. Here, any sequences containing illegal amino acid residues (such as B, X, O, or Z) or repeats of more than two residues were removed. Duplicates were also removed. This step removed over 2000 initial designs (Figure 5.2.1b). Next, NetSolP was used to predict the solubility of all designs. Any designs with predicted solubility or usability scores lower than the *T. maritima* template sequence were removed. Finally, a multichain score was calculated for each design. Both ESM-IF and ProteinMPNN can calculate a sequence likelihood score – the probability of a sequence given a protein structure. This can also be applied to complexes, where the model computes a likelihood of a sequence given a single chain in a multichain complex. Computing this likelihood for a sequence given an entire capsid of 60 subunits is not feasible given the GPU memory requirements of modelling all ≈47,000 atoms, and so to calculate the probability of sequences assembling into a full capsid, likelihoods were calculated against a single chain in either an encapsulin pentamer, or in a "capsid fragment" containing a single monomer and the 6 neighbouring subunits which make contacts with it.

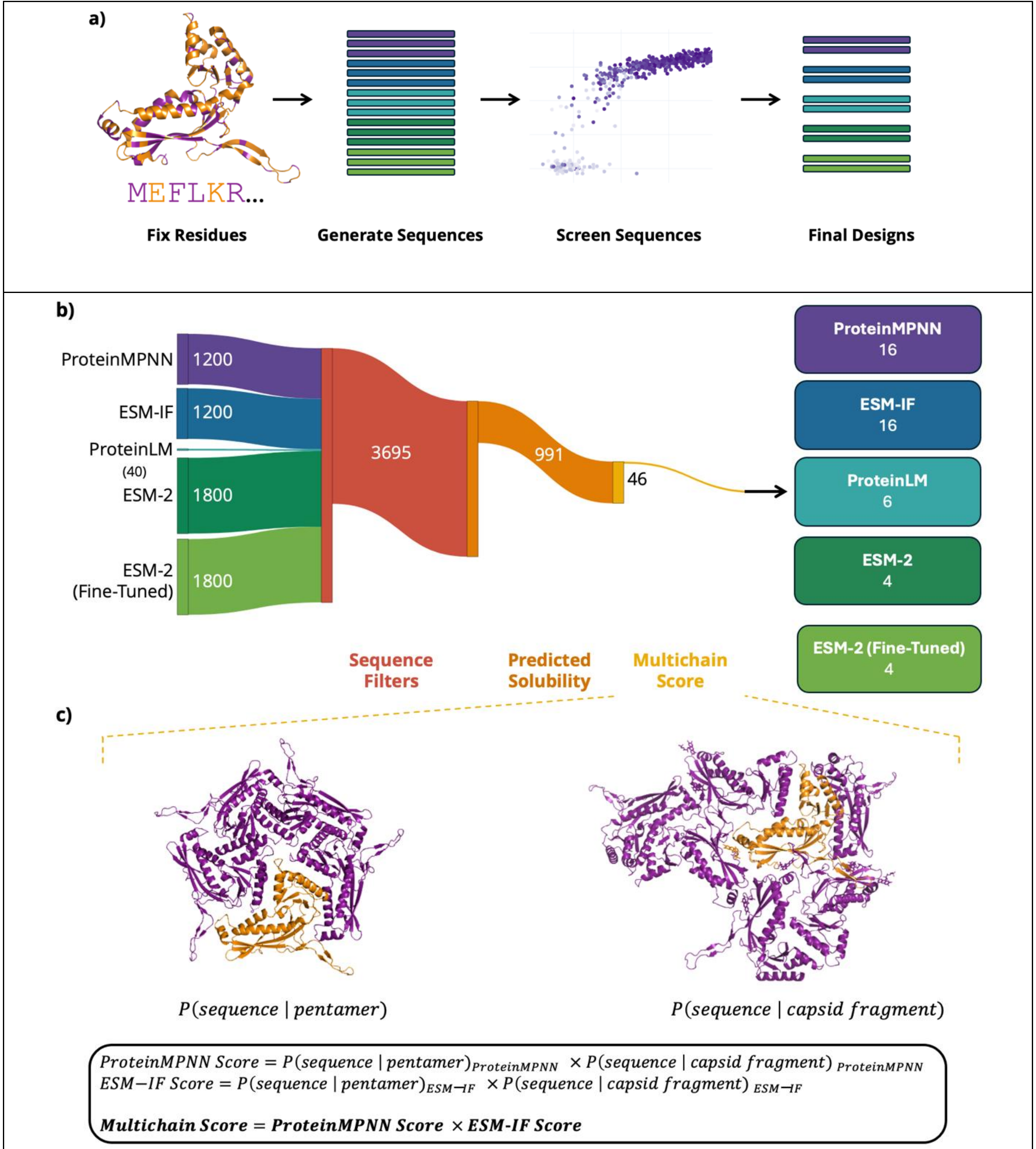


Figure 5.2.1: **Computational Design of Encapsulin Variants**
a) Illustration of the computational design pipeline. Starting from a structure or sequence, residue positions are fixed, and design models used to generate candidate sequences, which are screened to produce a final set of sequences for experimental validation. **b)** Sankey diagram showing the number of candidates screened. Designs are first screened using a set of basic sequence filters, before solubility is predicted using NetSolP, and any sequences with predicted solubility lower than the template are removed. Finally, sequences from each model are ranked using an aggregate multichain likelihood score, and 46 sequences total are chosen across the four models (see text for details on filters). **c)** Overview of multichain scoring. ProteinMPNN and ESM-IF are used to compute a probability for each sequence, against a single chain in a complex of either an encapsulin pentamer, or a fragment of the capsid containing the 6 neighbouring subunits which contact a single monomer.

Multichain scoring is explained schematically in Figure 5.2.1c. Following selection, structures were predicted for each chosen design using AlphaFold2 in single-sequence mode as in [211]. All designs showed high TM-Score and pLDDT when predicted with AlphaFold2 (Figure 5.2.2a) as well as ESMFold (not shown). Designs also spanned a wide range of sequence identity against the original *T. maritima* template sequence, between 40% and 95% (Figure 5.2.2b).

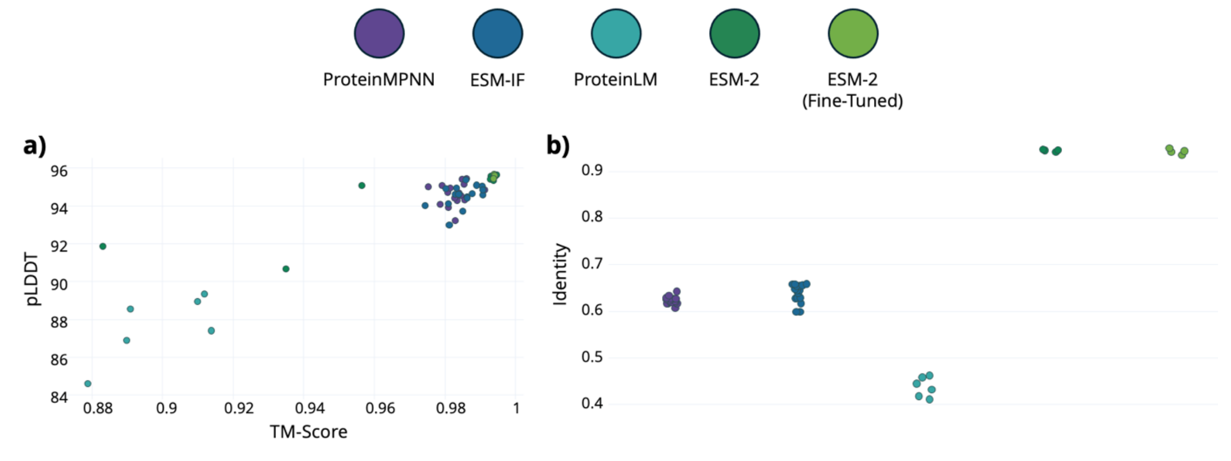


Figure 5.2.2: **Quality Metrics for Encapsulin Designs**
Plots showing **a)** TM-Score and pLDDT of single-sequence AlphaFold2 predictions and **b)** sequence identity against the template sequence for all 46 encapsulin designs. All designs show acceptable TM-Scores and pLDDTs, above 0.8 and 80 respectively. Sequence identities of the designs span a wide range - as low as 40% for the ProteinLM designs, around 60-70% for the inverse folding designs, and above 90% for the ESM-2 designs. ESM-2 vanilla and fine-tuned designs also appear to show the highest TM-Scores and pLDDTs.

5.2.2 Experimental Encapsulin Screening

Synthetic DNA fragments encoding each of the 46 encapsulin designs was ordered. Fragments contained DNA encoding the encapsulin protein (codon optimised for *E. coli* K12 using the IDT website), along with the *T7* promoter and RBS upstream and *rrnB T1* and *T7* terminators downstream of the gene, and BsaI sites on either end of the fragment for assembly into pSB1C3-FB with Type IIs assembly. Protein coding genes were ordered without Strep Tag, which was to be added once fragments were cloned into pSB1C3-FB.

Figure 5.2.3a shows the results of the Type-IIs assembly process. All 46 assembly reactions gave colonies, and 32 were verified as correct by sequencing. Of the remaining 14 samples, only 1 was verified as incorrect, with another sample verified as correct by sequencing but with a single residue mutation. 12 samples gave inconclusive Sanger sequencing data. All 46 designs were carried forward to the next stage of cloning regardless of sequencing data.

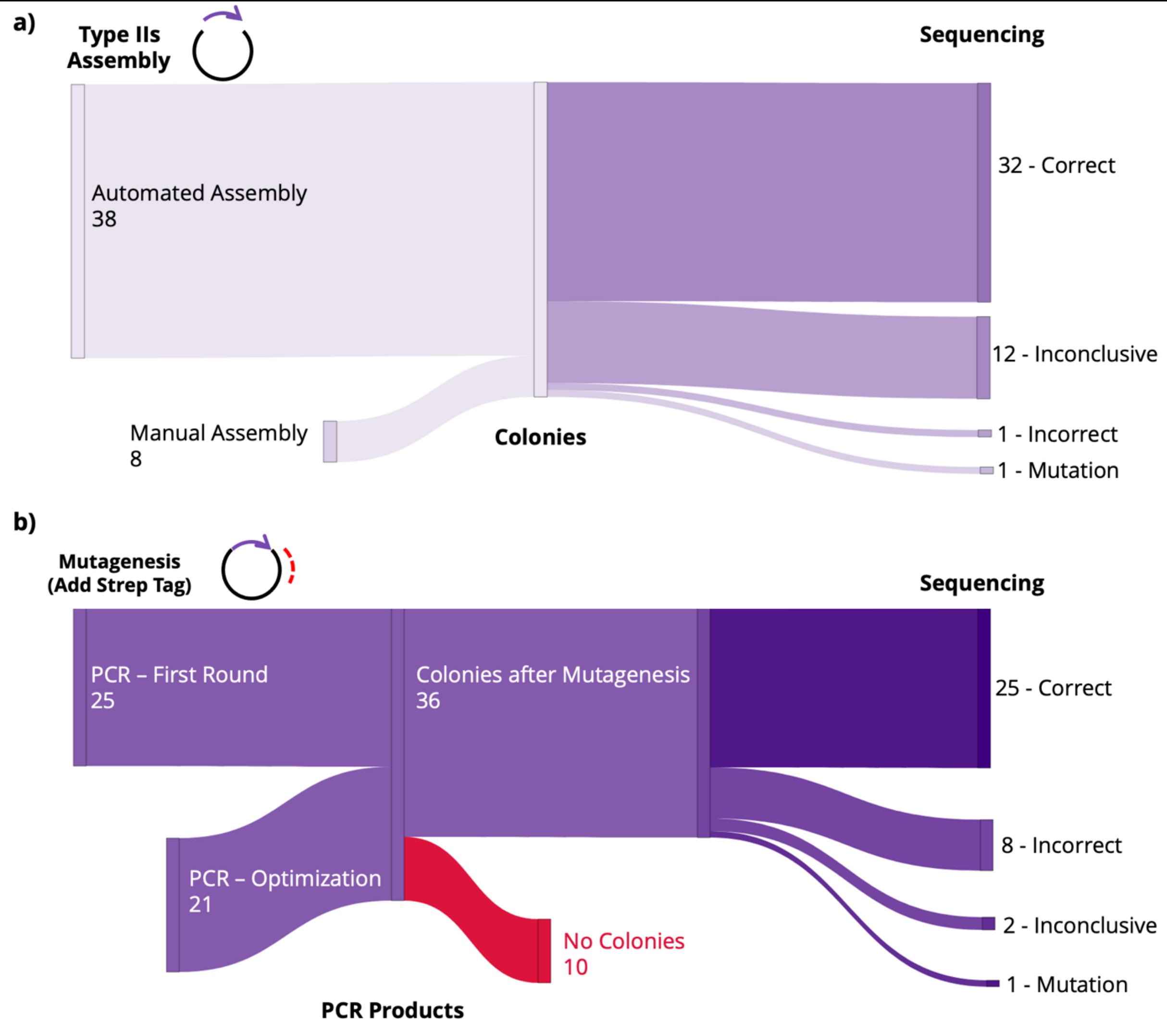


***Figure 5.2.3*: Parallel Cloning of Encapsulin Protein Designs**
a) Type-IIs DNA assembly results for all 46 designs. 38 designs gave colonies at the first attempt with the automated protocol, and the remaining 8 gave colonies with manual optimisation. Plasmid DNA was prepared and sequenced for all samples, and 32 were verified as correct by Sanger sequencing. 12 samples gave inconclusive sequencing results, where data from either primer read was of poor quality. 1 sample was sequenced as incorrect with a deletion, and 1 final sample was sequenced as correct but with a single residue mutation. Regardless of sequencing results, all 46 plasmids were carried forward to the next stage of cloning. **b)** Results of site-directed mutagenesis experiments. 25/46 constructs gave a PCR product after a single attempt, and the remaining 21 gave products after a round of optimisation. Following ligation and transformation, 36 mutagenesis products gave colonies, and of these 36 products with colonies, 25 were verified as correct by sequencing. 8 were incorrect (missing sequence fragments), 2 had inconclusive sequencing data, and 1 plasmid showed a single residue mutation.

Next, a Strep Tag II sequence was added to the C-terminus of each design using site-directed mutagenesis, to facilitate high-throughput solubility screening using the dot blot. A single forward primer binding to the stop codon region of the plasmid was used, containing an overhang with the Strep Tag II sequence (WSPHQFEK). 46 different reverse primers were designed using the primer3 python package, binding to the region of the protein immediately before the stop codon.

Figure 5.2.3b shows the results of the mutagenesis process. All 46 plasmids gave a PCR product, either at the first time or on optimisation (adding 3% DMSO and 1M Betaine to PCR reactions). 36 PCR products gave colonies when ligated and transformed, but 10 samples gave no colonies and were not further investigated. Colonies were grown in LB medium and plasmid DNA purified and sent for sequencing. 25/36 samples were verified correct by sequencing; however, 8 samples were verified as incorrect (missing the Strep Tag II sequence or other sequence regions). Of the remaining 3 samples, one was correct but with a single residue mutation (Gln2 mutated to Ser), and the other 2 showed inconclusive sequencing data. Again, all constructs were carried forward for solubility screening regardless of the sequencing results. A list of all protein sequence designs along with their sequencing status and links to Sanger sequencing files is shown in Table 7.5 (Appendix).

Next, encapsulin design plasmids were transformed into *E. coli* and proteins expressed in TB medium using IPTG induction as described in Section 4.4.2. As shown in Figure 5.2.4a, none of the 46 designs showed any signal on the solubility dot blot, compared to positive and negative controls. To confirm this negative result, a traditional western blot was carried out on total cell lysates, to investigate whether proteins were expressed at all. This analysis was carried out only for the sequence verified designs. This blot showed that some proteins expressed at a low level compared to the wild-type, where others showed no expression at all (Figure 5.2.4b).

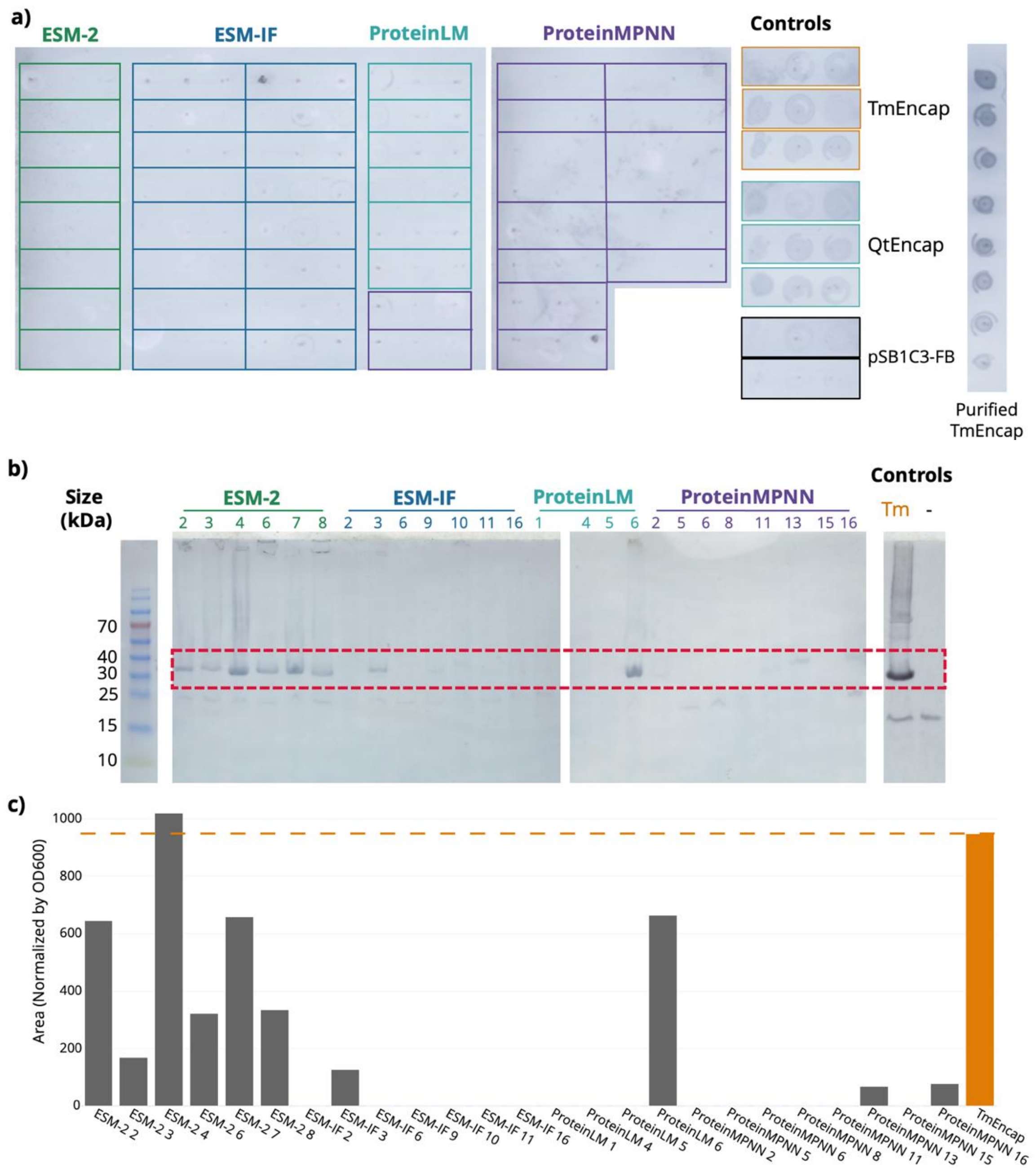


***Figure 5.2.4:* Solubility Screening of Encapsulin Designs**
a) Anti-Strep Tag dot blot of 46 encapsulin design soluble lysates. Controls shown are *T. maritima* and *Q. thermotolerans* encapsulins (TmEncap and QtEncap) and empty plasmid (pSB1C3-FB). Purified TmEncap was used as loading control. Each spot is a biological replicate (n=3), and each rectangle in the grid corresponds to a different design.). Positive controls showed detectable signal, and negative controls showed low background signal, but none of the designs showed detectable signal in any biological replicate. **b)** Anti-Strep Tag western blot on total cell lysates of the 25 sequence-verified encapsulin design constructs. Some samples show signal around the expected molecular weight (indicated in red) suggesting that these proteins show some expression, but not enough soluble yield to detect on the dot blot. Other samples show no signal, indicating failed expression. Expressed TmEncap protein and an empty plasmid were used as a control again. **c)** Plot showing areas under the intensity curves for the visible bands in **b)** measured by ImageJ and normalised by OD600. Only one sample showed higher intensity than TmEncap and this sample failed to show higher soluble yield in further experiments (Figure 7.1.4, Appendix).

When bands from this blot were quantified using ImageJ and normalised by OD600, it was revealed that only a single sample showed higher total expression than the wild-type. However, further investigation of the soluble fraction of this lysate confirmed that, while total expression was comparable to the wild type, soluble yield was still lower than the *T. maritima* encapsulin (Figure 7.1.4, Appendix). Overall, this data indicates that none of the 46 encapsulin designs were successful in achieving the original aim of increased soluble yield over the wild type protein.

5.3 Chimeric Encapsulin Variants

5.3.1 Design of Chimeric Encapsulins with Modified E-Loops

Next, two encapsulin variants were manually designed, to investigate the effect of the E-loop region on encapsulin assembly. Two wild-type encapsulin structures, from *T. maritima* and *Q. thermotolerans*, with T-numbers of 1 and 4 respectively, were used as a starting template. Sequence regions corresponding to the E-loop from these two wild-type structures were delineated according to a structural alignment and swapped to create two new chimeric designs (Figure 5.3.1b). Single-sequence AlphaFold2 predictions of these new chimeric proteins showed acceptable pLDDT and PAE values, apart from in these loop regions. This is expected given that these loop regions were swapped and are also expected to be flexible.

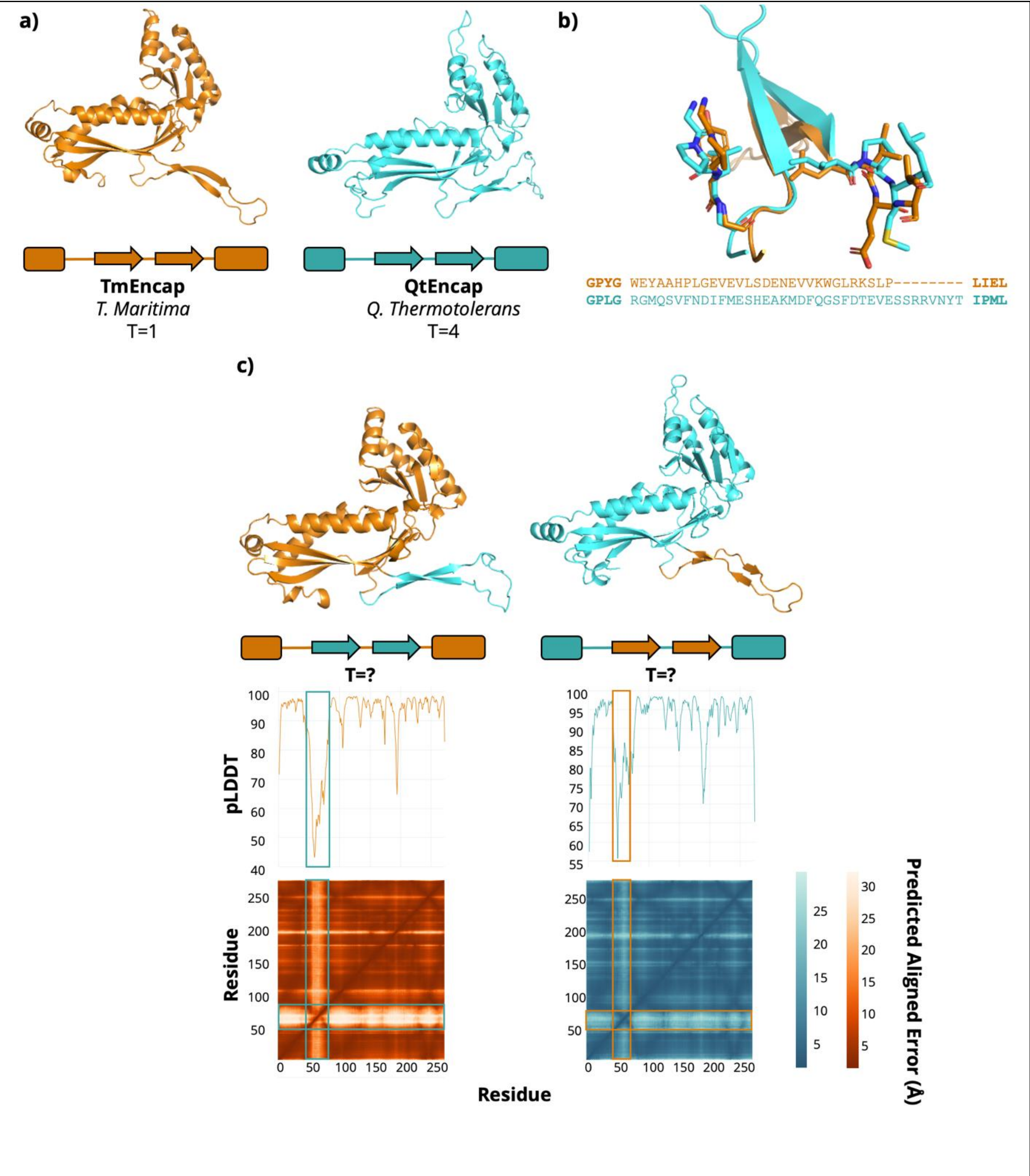

a)
TmEncap
T. Maritima
T=1
QtEncap
Q. Thermotolerans
T=4
b)
GPYG WEYAAHPLGEVEVLSDENEVVKWGLRKSLP--------- LIEL
GPLG RGMQSVFNDIFMESHEAKMDFQGSFDTEVESSRRVNYT IPML
c)
T=?
T=?
pLDDT
Residue
Residue
Predicted Aligned Error (Å)

Figure 5.3.1: **Design of Chimeric Encapsulin Variants**
a) Monomer structure of wild-type encapsulins from *T. maritima* (orange) and *Q. thermotolerans* (cyan), with schematic view of the HK97 fold (rectangles) and the E-loop region, with antiparallel β sheets (arrows). **b)** Structure and sequence alignment of the E-loop region from the two encapsulin structures. The four residues either side of the loop region (highlighted in bold) structurally align and show high sequence similarity. The sequence region between these four-residue motifs was swapped between the two sequences. **c)** Single-sequence AlphaFold2 predicted structures of the two chimeric designs with swapped E-loops, along with schematic depiction of the new topology (top). Plots showing pLDDT (middle) and PAE (bottom) of the two chimeric designs, with E-loop regions highlighted with rectangles. E-loop regions in the chimeric variants show low confidence and high PAE values compared to the rest of the HK97 fold.

Following design, protein sequences for these two chimeric variants were ordered as synthetic gene fragments, cloned into pSB1C3-FB, and screened for solubility in the same experiment as the deep learning encapsulin designs described in Section 5.4.1. Both chimeric designs were successfully cloned with the Strep Tag-II sequence added, and sequence verified. As shown in Figure 5.3.2, one of these chimeric designs appeared to show strong soluble signal, both on the dot blot and on a Western Blot, compared to the wild-type proteins. This chimera was comprised of the *Q. thermotolerans* encapsulin with the *T. maritima* E-loop region. This serendipitously aligns with the design aim of the previous section – finding new encapsulin variants with increased soluble yield. As such, this chimeric variant (henceforth referred to as CmEncap) was further characterised.

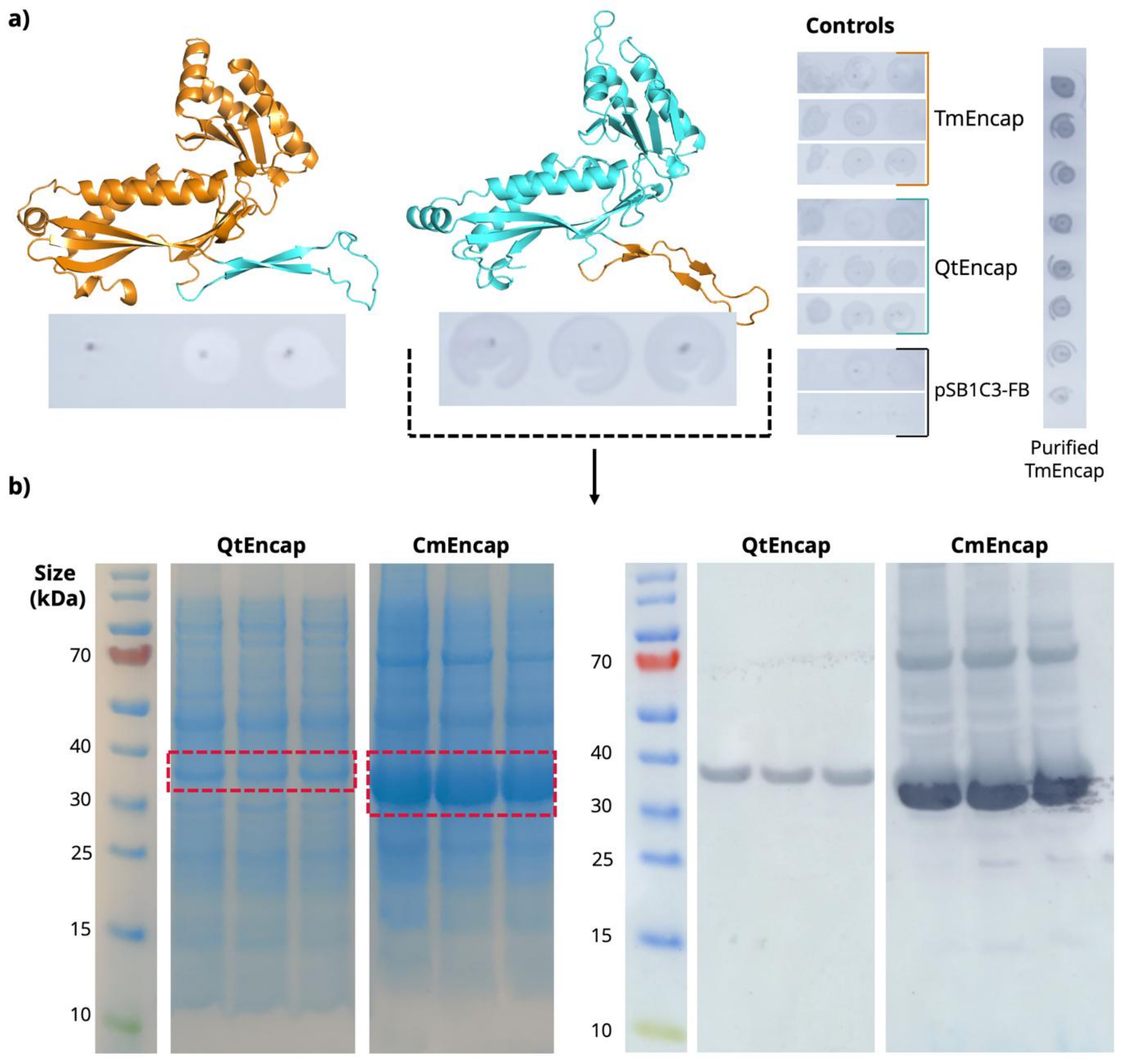


***Figure 5.3.2*: Solubility Screening of Chimeric Encapsulin Designs**
a) Dot blot of two encapsulin chimeric designs, from the same experiment as Figure 5.2.4 with same controls. The *T. maritima* encapsulin with *Q. thermotolerans* E-loop showed low signal, whereas the reciprocal design showed a strong signal on the dot blot. Each spot is a biological replicate. **b)** SDS-PAGE (left) and Western Blot against the Strep-Tag II (right) of soluble lysates from the dot blot. Samples shown are the wild type *Q. thermotolerans* encapsulin (QtEncap) and *Q. thermotolerans* encapsulin with *T. maritima* E-loop (CmEncap). CmEncap showed strong signal in SDS-PAGE. A western blot confirms these findings, with much stronger signal for CmEncap over wild-type. Samples shown are not normalised by OD600, but normalizing band intensities from the Blot gives fivefold higher intensity for CmEncap over wild-type (not shown). Each lane is a biological replicate.

5.3.2 Experimental Characterisation of a Chimeric Encapsulin

Next, CmEncap was expressed in *E. coli* in large scale and purified using affinity chromatography. Pure fractions from affinity chromatography were then loaded onto a gel filtration column for size exclusion chromatography. QtEncap was also expressed and purified alongside CmEncap, and both were subjected to downstream analysis.

As shown in Figure 5.3.3b, the wild-type QtEncap protein shows a single, sharp peak around 8 ml when eluting from a Superose 6 size exclusion column, as expected for assembled encapsulin particles. In contrast, CmEncap shows a much smaller peak which appears to be formed of two components. The majority of CmEncap protein (≈97 % as estimated from peak area) appears to elute much later from the column, suggesting that only a small fraction of total protein is assembled into particles. Native-PAGE and TEM images (Figure 5.3.3c-d) show that this early peak is indeed formed of assembled particles, and that the late peak is formed of unassembled particles. Native-PAGE suggests that this unassembled population is formed mainly of monomers and dimers, with a smear of higher molecular weight species. Inspection of TEM images reveals that, whilst QtEncap assembles into a monodisperse population of ≈42 nm particles, the ≈3% fraction of assembled CmEncap particles forms two discrete populations of particles: a smaller ≈20 nm size and a larger ≈42 nm size (Figure 5.3.3e). These sizes are consistent with a T=1 and T=4 sized particle respectively.

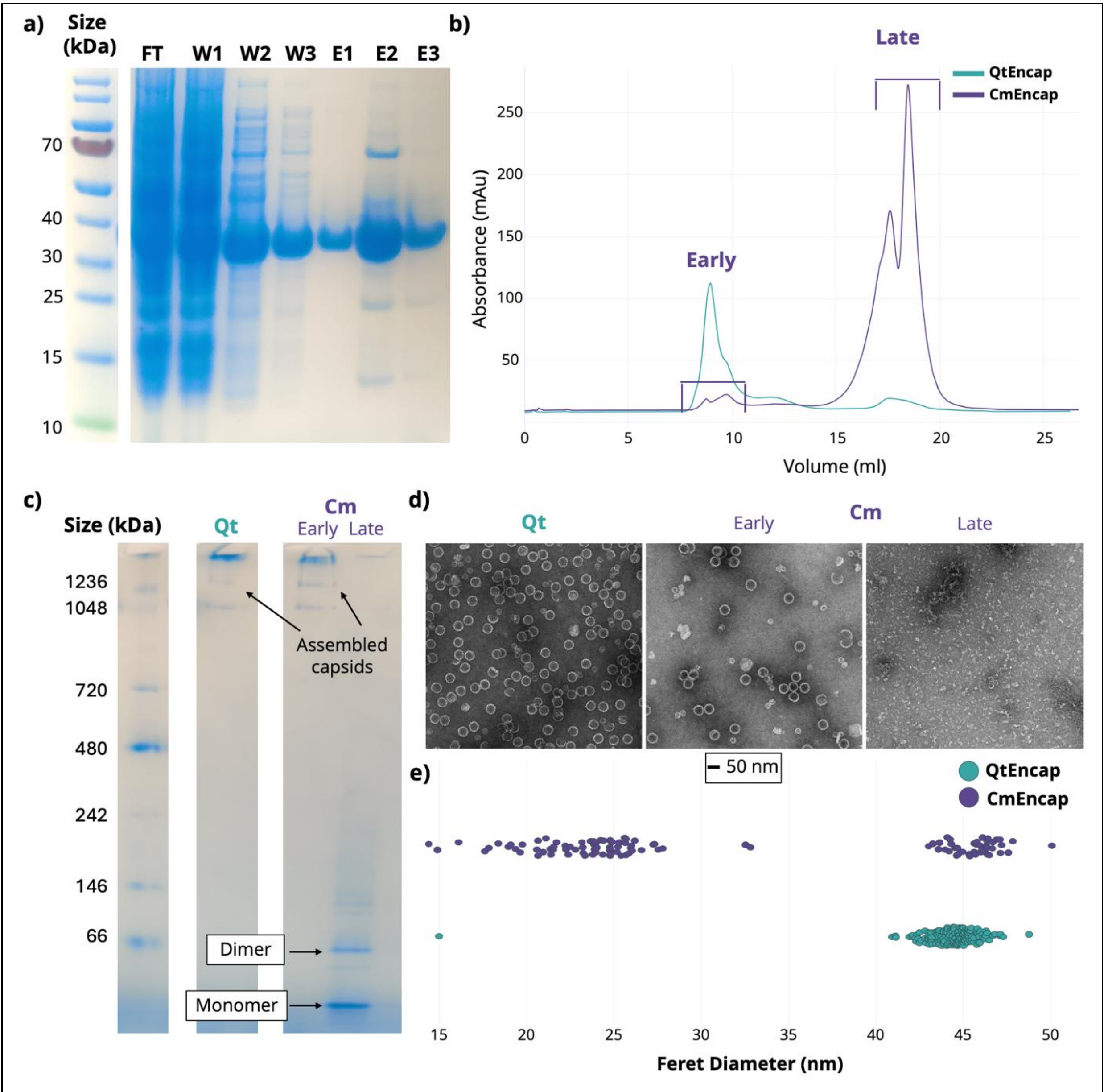


***Figure 5.3.3*: Experimental Characterisation of CmEncap**
a) SDS-PAGE of purified CmEncap fractions following large scale expression in *E. coli* and purification using affinity chromatography. **FT** = column flow-through, **W** = wash fractions, **E** = elution fractions. **b)** Elution fractions were pooled and loaded onto a Superose 6 gel filtration column. QtEncap shows the expected behaviour, showing a large, sharp peak around 8 ml corresponding to assembled particles (cyan). CmEncap shows a very small double peak around 8 ml, and a much larger peak around 14 ml. **c)** Native-PAGE of purified QtEncap and CmEncap both from the early and late peaks shown in **b)**. QtEncap shows the expected band pattern, with faint bands at the top of the gel around 1200 kDa. The early CmEncap peak also shows this behaviour, but the later peak consists of low molecular weight species which appear to correspond to a monomer, dimer, and other partial assembly products. **d)** TEM images of purified QtEncap and CmEncap early and late peaks. As in SEC and Native-PAGE, these images reveal that the early CmEncap peak corresponds to assembled particles, although there appears to be two populations of particles, one around ≈20 nm and one around the same size as QtEncap (≈42 nm). **e)** Plot showing the distribution of particle sizes from TEM images, as measured by Feret diameter in ImageJ, confirming two populations of particles for CmEncap.

5.4 Discussion

5.4.1 Redesigning Encapsulins for Increased Soluble Yield

In this chapter, a wild type encapsulin protein was redesigned, with the aim of increased soluble yield. 46 candidates were designed, however of these only 25 were successfully cloned and sequence verified. The greatest source of difficulty in cloning was the mutagenesis stage of adding Strep-Tag II sequences to all 46 designs – in future if this work is to be repeated, this sequence should be included in the initial synthetic DNA designs, rather than added later using PCR. However, none of these 46 designs showed any visible signal on the dot blot solubility screen, sequence verified or otherwise. DNA sequence errors are the most obvious cause of negative results in the 21 plasmids which were not verified as correct by Sanger sequencing. However, there are many other potential points of failure in the 25 sequence verified plasmids.

Western blot of total cell lysates in the 25 sequenced plasmids shows that most of these proteins express to a very low level in *E. coli* under the expression conditions used in this experiment. It is possible that higher expression levels could be achieved by varying these conditions – induction temperature, *E. coli* strain, medium type, and similar. However, the conditions used here give good soluble yield for the wild-type protein and so were used as a basis to screen for any improved variants. Only one design expressed to a level comparable with the wild type *T. maritima* protein, however this protein was mostly insoluble. These results may indicate that, in all but one case, designed proteins are either unstable or toxic when expressed in *E. coli*, causing low overall yield. Possible causes for these poor design results could lie in the protein design models and methods used, or in the computational screening process used to choose candidates for experimental validation.

During design, certain residues were fixed in an attempt to preserve wild type function. It is possible that too many residues were fixed in these design experiments, which could cause designs to have reduced solubility. Previous work has shown that too high a proportion of fixed residues in a wild-type protein can result in reduced solubility following redesign with ProteinMPNN, in optimization of a lysozyme and a myoglobin [211]. Outside of this, it is possible that the protein design methods used have limitations which affect the quality of encapsulin designs. ProteinMPNN and ProteinLM have both been validated experimentally [162, 171, 211, 262], suggesting that these models are capable of designing well performing proteins in some cases. However, both models have been used to design mostly small, single-domain proteins, which is a much easier design task than a large self-assembling protein organelle. The HK97 fold also contains several beta sheet regions, which can be difficult to design using deep learning methods, since sheet regions are less common than helical regions in the PDB which is used to train these models [263]. Indeed, more recent benchmarking efforts have shown that deep learning-based design tools such as ProteinMPNN and ESM-IF fail to outperform "classic" biophysics-based design tools such as Rosetta [264]. In this work, the "vanilla" ProteinMPNN model weights were used, trained on the entire PDB. However, the authors of this tool also provide a "soluble" model which is trained only on soluble protein structures (no membrane proteins). This model may have given better results in the pipeline used here.

However, while ProteinMPNN and ProteinLM have been experimentally validated previously, there are no examples in the literature of ESM-IF protein designs being tested experimentally. Anecdotal evidence suggests that other users have had issues with the solubility of ESM-IF designed proteins (Martin Pacesa, personal communication). These results are particularly interesting since ESM-IF and ProteinMPNN are both inverse folding models which set out to solve the same design task, however the former has not been proven to work in the real world, where the latter has been demonstrated to design successful proteins multiple times. Compared to ProteinMPNN, ESM-IF uses a larger neural network model with orders of magnitude more parameters, and most notably is trained on AlphaFold2 predictions as well as experimental structures from the PDB. Related work on protein design has shown that AlphaFold2 structures are not as "designable" as experimental structures, potentially due to many local inaccuracies in atomic positions [265], and in the ESM-IF paper it was shown that training solely on AlphaFold2 structures leads to worse performance [163]. As such, ESM-IF's poor performance could be due to the inclusion of predicted structures in the training data, which the better-performing ProteinMPNN does not use.

As for the negative ESM-2 results, there are several potential reasons for the poor performance of these designs. The design method used here involved removing residues from the template protein sequence and replacing them according to PLM likelihoods. This method has been documented in the literature previously [232, 256] but has not been applied to the latest ESM-2 models, and crucially, no previous work has subjected this design method to experimental validation. As such, it could be possible that these PLM likelihoods alone are not suitable for protein design or optimising protein sequences towards a given target property. It has been hypothesised previously that PLMs broadly learn the evolutionary landscape and statistics across protein families [266], as opposed to learning the precise determinants behind protein structure, function, and biophysical properties. It is thus possible that these PLM likelihoods alone are insufficient for protein design in some cases, and as such other methods have combined likelihoods from PLMs like ESM with potentials from more traditional physics-based design methods [267].

Finally on the topic of design tools, it is possible that the underlying rationale behind all the designs generated in this work is flawed. With all four models, design tools were used to generate sequences against the *T. maritima* encapsulin monomer, which is a gross simplification of the entire, assembled capsid with 60 subunits. Monomeric design ignores the true biological complexity of the encapsulin shell, to make protein design more computationally tractable. The monomer used in these experiments contains 264 residues, or ≈1320 backbone atoms, and designing sequences for this monomer required anywhere from 3-15 gigabytes of GPU memory, depending on the size of the neural network used for protein design. It is thus computationally infeasible to model an entire capsid of 60 monomers, without access to industrial-scale GPU clusters. And even in this case, the protein design methods tested in this work were not trained on or intended for use with such large protein complexes. However, in hindsight, it may have been possible to use the inverse folding methods (ProteinMPNN and ESM-IF) to perform multichain design against a single copy of the monomer in complex with either a pentamer or a capsid fragment as described in Section 5.2.1.

An alternative explanation for these negative results lies not in the protein designs themselves, but in the screening process used to choose candidates for experimental validation. 46 sequences were chosen from a pool of thousands of designs, and it is possible that suitable designs were found outside this small sample chosen for experiments. Designs were selected based on predicted solubility metrics calculated with NetSolP, and with multichain scores calculated using ProteinMPNN and ESM-IF. It is possible that none of these metrics are associated with experimentally successful proteins in this case, and that their use may have biased the selection of designs towards those with poor properties. There is also evidence in the literature that predicted structure pLDDT does not correlate with experimental success in terms of protein stability or function [268–270], however following an initial sequence-based screen all designs showed acceptable pLDDT, and so in effect no filtering based on this metric was used. In general, whilst screening based on predicted structures is a de facto standard in protein design currently, caution should always be used when filtering protein sequences based on predicted structures (and indeed, caution should be used when dealing with predicted structures in any capacity). Structure prediction tools like AlphaFold2 and ESMFold are not especially sensitive to small errors in protein sequence, but in reality the structure of a protein can be completely disrupted or destabilised by even a single point mutation [271].

5.4.2 Chimeric Encapsulin Designs

In this work, two chimeric encapsulin proteins were designed with modified E-loop regions. The rationale behind this work was initially to investigate encapsulin assembly and the role of this E-loop region, however it was discovered that one of these chimeric designs showed vastly increased solubility compared to the wild-type protein. This rather serendipitous discovery aligned with the aims of the deep learning-based protein design work, and so experimental efforts focused on characterising this newly discovered chimeric mutant, formed from a *Q. thermotolerans* encapsulin with an E-loop region from the *T. maritima* encapsulin. Despite showing vastly increased soluble yield, it was seen that swapping this E-loop region causes assembly to be almost completely abolished, with only ≈3% of total pure protein being in the assembled state. Interestingly, the small fraction of assembled protein appears in TEM images to form two separate populations, whose diameters are consistent with a T=1 and T=4 sized particle respectively. However, this remains a hypothesis until more high-resolution structural information is available. As a brief aside, these findings also demonstrate the utility of the dot blot assay as a means to discover new encapsulin variants with increased soluble yield.

These results empirically demonstrate that the E-loop region of the HK97 fold is directly involved in capsid assembly, which has not been demonstrated experimentally in encapsulins. However, its precise role in assembly is still unknown. This work shows that completely swapping the E-loop in one encapsulin abolishes assembly, but more precise mutagenesis experiments are required to determine exactly which sequence motifs are involved. It may be possible that single mutants of the wild-type E-loop region have a less pronounced effect on assembly. The relationship between E-loop sequence and T-number also remains unclear without a solved structure of the chimeric mutant capsid. The E-loop may directly determine T-number, or it may act in concert with residues in the main body of the HK97 fold to direct assembly and geometry. Matters are further complicated by the fact that the E-loop introduced into the *Q. thermotolerans* encapsulin here is shorter than the wild-type loop. Again, it is unknown whether varying the length alone of this region is enough to perturb assembly, or whether it is the precise amino acid sequence which directs assembly (or whether both are interlinked). Overall, these experiments show a relationship between E-loop and capsid assembly, but the precise nature of this relationship is unclear and requires more detailed investigation.

The preliminary CmEncap assembly data presented here make it possible to speculate on the assembly mechanism and dynamics of encapsulin proteins. Phage capsids use multiple protein subunits and accessory proteins, and show a complex assembly pathway involving multiple stable intermediates [272]. In contrast, encapsulins only use a single protein subunit and no accessory proteins, and no stable intermediate states have been isolated. This suggests that encapsulins exist in equilibrium between the unassembled and fully assembled states. When wild type encapsulins are recombinantly expressed, this equilibrium heavily favours the assembled state, to the point where no monomers or unassembled species can be observed or isolated. This inability to isolate intermediates directly means that the assembly pathway of encapsulin particles can only be probed indirectly – two main methods are to make mutations which disrupt assembly, or to purify whole capsids and subject them to chemical disassembly and reassembly.

How do the CmEncap findings fit into the current knowledge from these two experimental approaches? As far as making assembly mutants is concerned: there are examples in the literature of mutations in the monomer protein changing the shape or symmetry properties of the assembled capsid [29, 30]. However, there are no examples of mutations which abolish or otherwise reduce capsid assembly in the published work on encapsulins. Unpublished Frank Lab experiments describe a double mutant of TmEncap with almost complete assembly, but with a very small fraction of dimers visible on a Native-PAGE gel. In contrast, this work presents a CmEncap mutant which shows almost completely abolished assembly, where only 3% of total protein is present as assembled particles. Native-PAGE gels show that the unassembled fraction of protein contains several well-resolved oligomeric species (Figure 5.3.3). Clear bands are visible with molecular weights consistent with the size of a monomer and dimer, as well as a larger band which may be a trimer, tetramer, or some other higher-order species. Such a range of well-resolved intermediate species has not been observed for any previous encapsulin protein, either by mutation (as in this work) or by chemical disassembly/reassembly.

As for these chemical disassembly and reassembly experiments, this is a much more fruitful area of the encapsulin literature. Both the literature [51] and unpublished Frank Lab experiments on TmEncap have shown the presence of the aforementioned dimer species after capsids are chemically disassembled (using either high pH or chemical denaturants) and re-equilibrated back into an assembled state. Native mass spectrometry experiments on the cargo-loaded *B. linens* encapsulin (also a T=1 capsid like TmEncap) have observed a 58-subunit partially assembled capsid, missing two subunits [273]. Taken together, these findings suggest that some encapsulins assemble starting from dimeric species which form the “base unit” of the shell, and exist as equilibria between assembled capsids and dimers, as opposed to monomeric species.

In contrast, QtEncap does not exhibit this dimeric species in assembly/disassembly experiments, and only shows a band consistent with the molecular weight of a monomeric species as shown in [51] and unpublished experiments. It is also worth noting that a well resolved monomeric species is visible for CmEncap (which is a QtEncap mutant). This suggests that QtEncap does not assemble by this same pathway starting from a dimer, but instead starts from unassembled monomers. However, the CmEncap mutant shows well-resolved higher-order species such as dimers and a putative trimer or tetramer, which may also be assembly pathway intermediates. QtEncap is also a larger T=4 capsid which suggests that perhaps larger encapsulin particles assemble by a different pathway than smaller T=1 capsids. However, this all remains speculation until further experiments can be done. The CmEncap findings shown here represent a first step towards elucidating an assembly pathway, where the regular assembly equilibrium which so heavily favours the QtEncap assembled state has been significantly shifted.

Another interesting observation is that both CmEncap and the non-functional MGYP-61 encapsulin candidate from Section 3.8 showed high soluble yield but poor assembly characteristics (or no assembly in the case of MGYP-61). This may be a simple coincidence; however, it may also suggest some kind of trade-off between assembly and soluble yield. It must be stressed that this is purely conjecture, but it is possible that a hypothetical, stable encapsulin monomer is more soluble than an equivalent encapsulin protein which assembles into full particles. There are many speculative reasons why this might be the case; from a protein sequence perspective, it seems reasonable that the set of soluble monomeric proteins with the HK97 fold is much larger than the set of soluble HK97 fold proteins which assemble into full capsids. It also seems reasonable to assert that large particles may be more prone to aggregation than smaller monomeric species. If this is the case, then the screening process used here may not make much sense. In this work, candidates are first screened for soluble yield, before investigating hits for their assembly properties. However, if assembly comes at the cost of soluble yield, then it may be necessary to first screen all designs for assembly, and then choose the highest yielding designs. However, this is difficult since screening for assembly is currently done using Native-PAGE and TEM measurements, which are low-throughput and require purified protein. In future, a higher throughput method for screening capsid assembly may be required.

5.4.3 Future Work

In this chapter, an attempt was made to design *T. maritima* encapsulin variants with increased soluble yield for biotechnology applications. Whilst this was not accomplished in the present work, future work may attempt to achieve this goal using a different computational approach. As mentioned previously, this may involve designing encapsulin sequences in the context of the entire capsid or a fragment thereof, or using different design tools to those tested here. It should also be noted that only 25/46 designs were sequence verified, so future work could start with re-cloning the remainder of these designs and investigating these further. The dot blot assay used here is relatively insensitive, and so future work may focus on developing better high-throughput methods for detecting protein solubility, perhaps using split green fluorescent protein reporters in combination with flow cytometry and next-generation sequencing as in [172]. As mentioned above and as borne out in the results presented in this thesis, screening for encapsulin assembly is just as important as finding soluble high-yield candidates. Future work should focus on developing a method for detecting capsid assembly, in higher throughput than the currently-used and laborious method of purifying proteins and screening using Native-PAGE and negative stain TEM imaging.

The chimeric encapsulin mutant CmEncap presents some interesting expression and assembly properties which could be investigated further. For example, if the non-assembling protein fraction can be separated into monodisperse populations of monomers, dimers, or other oligomeric species, then these could be subjected to crystallization screens and potentially investigated using X-ray crystallography. SEC-MALS analysis could also shed light on the precise composition of this non-assembled fraction and determine the proportions of monomeric, dimeric, and higher order species. Despite only forming a small proportion of the total expressed protein, the fully assembled capsid population could be analysed using cryoEM and single particle analysis to solve high resolution structures of the two particle sizes seen in the TEM images. Additionally, CmEncap has an entirely swapped E-loop region, but future work could focus on making individual mutations from the wild-type QtEncap protein and investigate the effect these have on assembly, to obtain more fine-grained information on capsid assembly.

A final point of discussion is the second chimeric encapsulin mutant which was designed. Recall that the CmEncap protein investigated here is the *Q. thermotolerans* encapsulin with an E-loop from the *T. maritima* encapsulin. However, the reciprocal mutant was also designed, consisting of the *T. maritima* encapsulin with the *Q. thermotolerans* E-loop. This protein was not characterised in any depth, since solubility screens showed that CmEncap had desirable properties which were sought after in this work. Given the altered assembly seen in CmEncap, it would be very interesting to revisit the other chimeric design and determine first whether it assembles into particles at all, and their size distribution if present.

Chapter 6: Discussion and Conclusions

To conclude, this work presents an exploration of the currently available deep learning tools and methods for protein design and discovery. This includes tools for protein structure and function prediction, tools for searching vast databases of protein structures and sequences, protein language models, and various tools for protein design using structure or sequence. These tools and models were applied to encapsulins, which are a useful test case for two reasons; they represent a difficult challenge for both discovery and design of new variants, and they also come with a wealth of exciting attributes for use in synthetic biology. In broad summary, this work demonstrates some of the exciting opportunities afforded by these new deep learning methods, as well as the potential pitfalls of using these methods, and the challenges which lie ahead if these tools are to become part of the landscape in experimental biology.

6.1 Deep Learning in the Biological Sciences

At this juncture it may be useful to review how the findings from this work fit in with the overall outlook of deep learning for biology. Many practitioners see deep learning as reliant on three key prerequisites: model architecture, compute, and data [274]. These three elements are all interdependent on each other: training models on larger datasets requires more compute, but as datasets scale models must also increase in size and complexity to properly learn the underlying distribution and properties of the data, which requires more compute, and so on. In practice one of these three elements is always limiting, which requires the other two to be optimised to maximise efficiency. Many studies have been carried out to empirically derive the so-called "scaling laws" which define the relationship between data, compute, and model size, and define the optimal way to train large models given either a fixed dataset or compute budget [188, 274–276].

As far as deep learning in biology is concerned, which of these interlinked elements is currently most important? In terms of models, there have been many recent advances in implementation, training, and model scaling to billions or even trillions of parameters. A wide range of model architectures are available to researchers, including the famous Transformer model underlying most large language models and many protein models. As for compute, in recent years, computing power has become cheaper and more easily available than ever before, and GPU hardware can be accessible at a low cost or even for free in some cases. However, training very large deep learning models (sometimes called "frontier" or "foundation" models) is still out of reach to all but the biggest of corporations and academic institutes. For example, OpenAI have estimated that the cost of training GPT-4 was over $100 million [277]. This work contains far more modest compute experiments, and even these would have been impossible without an external grant from Oracle for Research. Outside of these exceptional cases, deep learning is now more accessible than ever, especially to smaller academic groups whose focus may be more grounded in experimental biology.

In deep learning in general, the commonly accepted wisdom is that data is the most important element of the three. It is widely observed that obtaining and curating a smaller dataset of higher quality can provide better performance than larger, lower quality datasets [278], and these smaller datasets also require less compute to train models on. However, data is much more difficult and expensive to collect in the biological sciences compared to other fields like computer vision or natural language. Despite this, high quality data for proteins specifically (as is the focus of this work) is plentiful and readily available. The PDB is almost a textbook example of an excellent data resource for deep learning, containing a wealth of well-curated protein structures and associated metadata. Databases like UniProt and the NCBI nr database contain billions of protein sequences with varying levels of curation and annotation. Together, these resources have already driven big breakthroughs in deep learning for protein structure prediction and design, to name just two examples. However, data is always a limiting factor in deep learning, and even the best resources have limitations and "blind spots": for example, the PDB is biased towards types of proteins and limited in others (such as membrane proteins, disordered proteins, and fold switching proteins) [279]. Similarly, sequence databases only sample a small region of the space of all possible protein sequences, and it is thought that there is a wealth of diversity in proteins that have not yet been discovered [186].

To conclude on the "holy trinity" of deep learning, there are many opportunities available with current data, model, and compute capabilities, but also many potential avenues for exploration and improvement. Whilst model and compute advances are most likely to come from large technology corporations and computer scientists, the responsibility for improving deep learning datasets lies largely with experimental biologists. The big challenge in future may lie in the interplay between wet lab experiments and computational work, and experimental biologists have a big role to play in generating breakthroughs towards this goal. The experiments presented in this work go some way towards demonstrating this fact – a strong set of *in silico* predictions and hypotheses were generated but failed to provide positive results when validated experimentally.

This leads to the key take-home message of this work: the vital importance of the experiment. Computational tools have the power to make biological experiments faster, easier, reduce the number of experiments required, and in some special cases provide insight where experiments fail to do so. This is especially the case with deep learning tools available now. However, the vast promise of computational tools is best exploited as a companion to experiments, as the negative experimental results in this work have shown. This is a point that is particularly worth emphasising, since many papers or preprints describing new deep learning models for proteins neglect to perform wet lab validation of their findings.

In other deep learning modalities like computer vision or language, models can be somewhat intuitively evaluated. A model for generating text or images can be tested by human investigators, and human feedback is indeed an important part of the training and alignment process for these models. However, in a biological setting, model outputs cannot always be evaluated fully by human inspection. For example, protein sequences generated by a protein design model cannot be easily inspected by humans to check their quality. Outputs can be computationally screened, and models can be evaluated against benchmark datasets, but again, as this work shows, the true test of a biological deep learning tool's performance is whether the proteins it generates or the predictions it makes are feasible in the real world.

To conclude: biological deep learning models generate hypotheses about the physical world. For example, protein sequences generated by protein design models are hypotheses that a given protein sequence will be stable, soluble, fold into a given structure, perform a given function, or similar. As this work has shown, these hypotheses can be a valuable source of ideas, narrow down the set of experiments to be carried out, or provide useful information about a system which cannot be obtained experimentally. However, whilst some of these hypotheses lead to useful insights about biology, others will not always stand up to validation using experimental methods. However, it is worth noting that many experimental methods are often treated as ground truth, but suffer from similar limitations of fitting models to limited experimental data (such as X-ray crystallography, for example).

The metagenomic proteins and *de novo* designed proteins produced in this work are two large sets of hypotheses, and it was shown in experiments that some of these hypotheses did not provide the expected result when tested. The process of going from a large set of predictions to experimental results is not trivial by any means. As such, despite the vast promise shown by deep learning tools in biology, it remains the case that domain knowledge, experience, and intuition are still prerequisites in any biological investigation.

Chapter 7: Appendix

7.1 Supplementary Figures

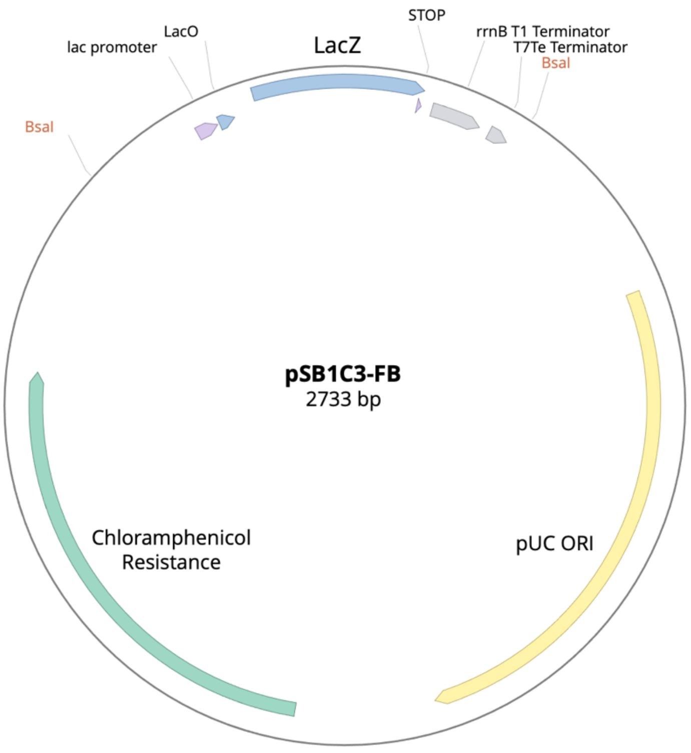
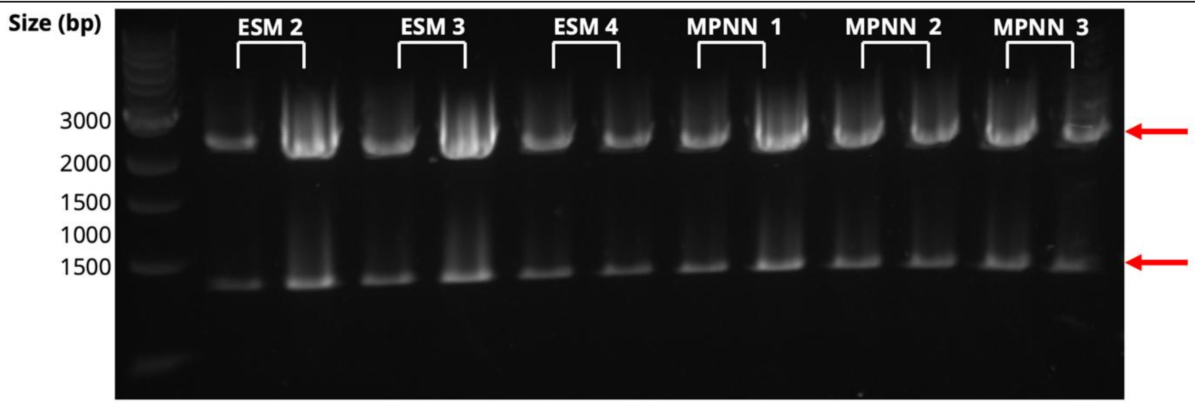


Figure 7.1.1: **pSB1C3-FB Plasmid Map**
Plasmid map for the pSB1C3-FB vector assembled in Section 4.3.1 and used in protein expression in this work. Sanger sequencing data for the BsaI sites (and the region between them) in the form of .ab1 and .seq files is available in this GitHub folder: https://github.com/naailkhan28/encapsulin_bioinformatics_exploration/tree/master/Experimental%20Data/Sequences/Sequencing%20Data/pSB1C3-FB



Enzymes	Expected Band Sizes (bp)
NdeI	875
NcoI	2286

Figure 7.1.2: **Control Type IIS DNA Assemblies**
Agarose gel electrophoresis showing a diagnostic restriction digest verifying manual Type IIS assemblies for 6/8 control inserts (3 each from ESM-IF and ProteinMPNN) as described in Section 4.3.1. Plasmid DNA was purified from two colonies per assembly and digested with NdeI and NcoI, and digests showed bands consistent with the sizes expected for this double digest (indicated with red arrows).

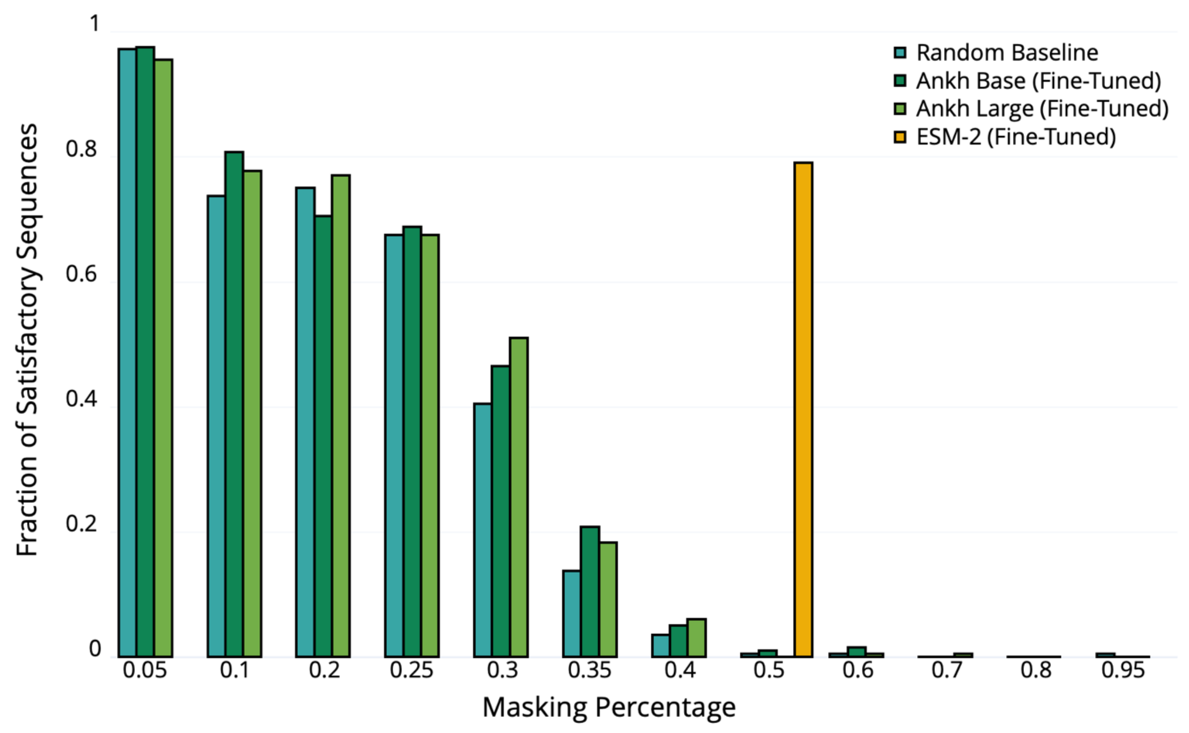
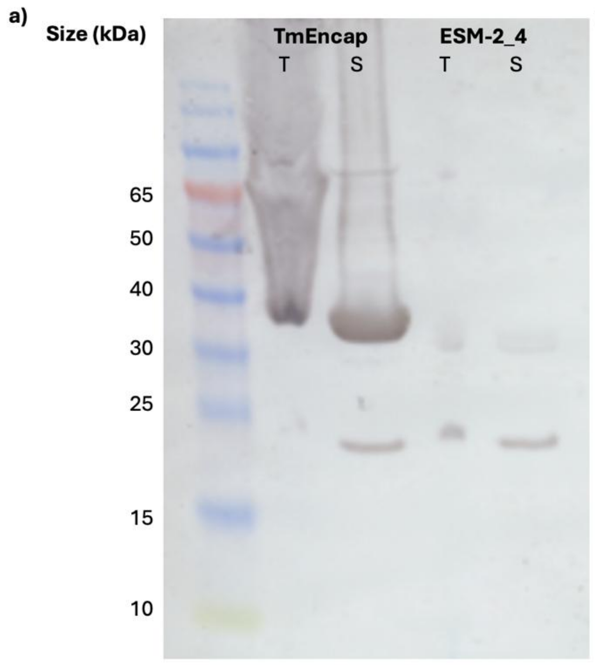


Figure 7.1.3: **Sequence Generation Performance of Fine-Tuned Ankh Models**
The fraction of satisfactory sequences is shown for fine-tuned Ankh Base and Large models versus a random baseline. Results shown are in MLM performance. Neither model greatly outperforms the random baseline at any masking level, and the fine-tuned ESM-2 model greatly outperforms both fine-tuned Ankh models. There is no consistent pattern in performance between Base and Large models.



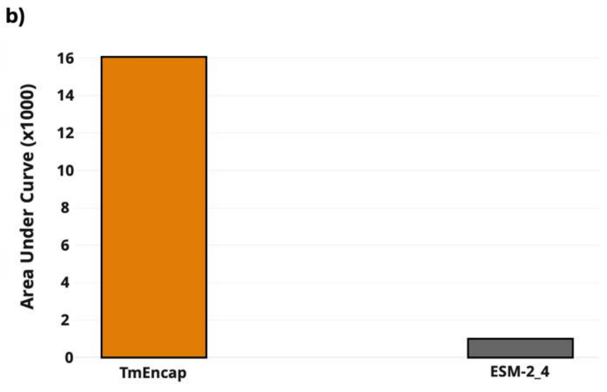


Figure 7.1.4: **Solubility Screening of ESM-2_4**
The ESM-2_4 design was seen to have comparable levels of total expression to the wild-type TmEncap protein in Figure 5.2.4. **a)** Anti-Strep-Tag II western blot showing either total cell lysates (T) or soluble fraction only (S), normalized by OD600. The low signal seen in ESM-2_4, T is likely a loading artefact since total cell lysates are very viscous and difficult to load and quantify in SDS-PAGE and western blots. Despite this, TmEncap shows a much stronger signal in the soluble fraction compared to the design. **b)** Quantification of the soluble signal from **a)** by measuring the area under the intensity curves for the bands of interest in ImageJ. TmEncap shows an order of magnitude stronger signal. Overall, these data confirm that ESM-2_4 exhibits much lower soluble expression yield than the wild-type.

7.2 Supplementary Tables

Table 7.1: **Fixed Residues in Encapsulin Design**

Residues shown were fixed in the redesign of the *T. maritima* T=1 encapsulin. The residue numbering scheme used is the same from the cryoEM structure (PDB **7MU1**).

Residues	Function	Source
Trp90, Asp93	Flavin Binding	[32], Frank Lab unpublished data
Arg73, Arg240	Capsid Assembly	Frank lab unpublished data
Asp23, Arg27, Lys31, Leu34, Arg37, Val40, Val42, Gln231-Ile235	Cargo Binding	[1]
Met4, Phe6, Leu7, Arg9, Phe11, Ala12, Gln18, Trp19, Ile22, Arg25, Ala26, Glu28, Ile29, Tyr35, Lys38, Asp41, Pro45, Gly47, Ala52, Gly56, Glu64, Val67, Trp70, Pro77, Ile79, Glu80, Leu81, Arg82, Thr84, Phe85, Leu89, Leu95, Arg97, Gly98, Asn101, Ala115, Glu118, Asp119, Val121, Gly125, Cys126, Ser129, Val131, Cys144, Gly145, Leu152, Ala158, Phe162, Asp165, Gly166, Ile167, Gly169, Pro170, Tyr171, Ile175, Trp180, His190, Tyr191, Pro192, Asp214, Arg221, Gly222, Asp224, Phe225, Leu227, Gly230, Gly236, Tyr237, Val245, Leu247, Phe248, Glu251, Thr254, Phe255, Asn259, Glu261, Ala262, Ile264, Leu266	Conserved Residues	MSA Analysis (see text)

***Table 7.2*: Raw Images Presented In This Document**
The type and description of each raw image is shown, along with a link to the figure where the image is used, and a GitHub link is provided to the raw image.

Figure	Description	Image Type	GitHub URL
Figure 3.7.1	MGYP-61 expression screening gel	SDS-PAGE	https://github.com/naailkhan28/encapsulin_design/blob/master/plots/images/2023-07-06-MGYP_61_Temperature_screen.jpg
Figure 3.7.1	MGYP-11 expression screening gel	SDS-PAGE	https://github.com/naailkhan28/encapsulin_design/blob/master/plots/images/2023-07-07-MGYP_11_small_scale_expression.jpg
Figure 3.7.1	MGYP-87 expression screening gel	SDS-PAGE	https://github.com/naailkhan28/encapsulin_design/blob/master/plots/images/2023-07-10-MGYP_87_sma_scale_temperature_screen.jpg
Figure 3.8.1	MGYP-61 assembly screening gel	Native-PAGE	https://github.com/naailkhan28/encapsulin_design/blob/master/plots/images/2023-07-18-MGYP_61_Streptactin_E3_Native_PAGE.jpg
Figure 4.3.2	DNA assembly optimization – outgrowth plating	Agar Plate	https://github.com/naailkhan28/encapsulin_design/blob/master/plots/images/2023-03-31-MPNN_3_GG_Opentrons_Plating_test.tif
Figure 4.3.2	DNA assembly optimization – silicone oil and foil seals	Agar Plate	https://github.com/naailkhan28/encapsulin_design/blob/master/plots/images/2023-06-14-GG_Opentrons_Transformation.png
Figure 4.3.2	DNA assembly optimization – foil seals only	Agar Plate	https://github.com/naailkhan28/encapsulin_design/blob/master/plots/images/2023-06-16-GG_Opentrons_Transformation.png
Figure 4.3.3	Optimised solubility screening protocol blot	Dot Blot	https://github.com/naailkhan28/encapsulin_design/blob/master/plots/images/2023-09-20-96well_encapsulins_clarified_lysates_TmEncap_control.tif
Figure 5.2.4, Figure 5.3.2	Encapsulin design solubility screening blot	Dot Blot	https://github.com/naailkhan28/encapsulin_design/blob/master/plots/images/2024-02-23-Encapsulin_design_cloning_v2_expression_dot_blot.tif
Figure 5.2.4	Encapsulin design expression screening blot	Western Blot	https://github.com/naailkhan28/encapsulin_design/blob/master/plots/images/2024-04-17-Encapsulin_design_expression_v3_lysate_antiStrep_Western_x2.tif
Figure 5.2.4	Encapsulin design expression screening blot (controls)	Western Blot	https://github.com/naailkhan28/encapsulin_design/blob/master/plots/images/2024-04-08-Encapsulin_design_expression_v3_lysate_antiStrep_Western_x3.tif
Figure 5.3.2	CmEncap and QtEncap solubility gel	SDS-PAGE	https://github.com/naailkhan28/encapsulin_design/blob/master/plots/images/2023-02-27-Encapsulin_Design_TmEncap_QtEncap_pSB1C3_CmEncap_A2_SDS-PAGE.jpg
Figure 5.3.2	CmEncap and QtEncap solubility blot	Western Blot	https://github.com/naailkhan28/encapsulin_design/blob/master/plots/images/2024-02-27-Encapsulin_design_TmEncap_QtEncap_pSB1C3_CmEncap_A2_western_blot_antistrep.tif
Figure 5.3.3	CmEncap purification gel	SDS-PAGE	https://github.com/naailkhan28/encapsulin_design/blob/master/plots/images/2024-05-09-CmEncap_Superose6.jpg
Figure 5.3.3	CmEncap assembly gel	Native-PAGE	https://github.com/naailkhan28/encapsulin_design/blob/master/plots/images/2024-03-15-Tm_Qt_Cm_early_late_native.jpg

Table 7.3 **- Protein Sequences Used In Experimental Work**
All protein sequences were codon optimized for *E. coli* K12 using the IDT Codon Optimization Tool.

Name	Sequence	Location
TmEncap	MSEFLKRSFAPLTEKQWQEIDNRAREIFKTQLYGRKFVDVEGPYGWEYAAHPLGEVEVLSDENEVVKWGLRKSLPLIELRATFTLDLWELDNLERGKPNVDLSSLEETVRKVAEFEDEVIFRGCEKSGVKGLLSFEERKIECGSTPKDLLEAIVRALSIFSKDGIEGPYTLVINTDRWINFLKEEAGHYPLEKRVEECLRGGKIITTPRIEDALVVSERGGDFKLILGQDLSIGYEDREKDAVRLFITETFTFQVVNPEALILLKFSGASDDDDKGAWSHPQFEKTG	Section 4.3.2, Section 5.2.2
QtEncap	MNKSQLYPDSPLTDQDFNQLDQTVIEAARRQLVGRRFIELYGPLGRGMQSVFNDIFMESHEAKMDFQGSFDTEVESSRRVNYTIPMLYKDFVLYWRDLEQSKALDIPIDFSVAANAARDVAFLEDQMIFHGSKEFDIPGLMNVKGRLTHLIGNWYESGNAFQDIVEARNKLLEMNHNGPYALVLSPELYSLLHRVHKDTNVLEIEHVRELITAGVFQSPVLKGKSGVIVNTGRNNLDLAISEDFETAYLGEEGMNHPFRVYETVVLRIKRPAAICTLIDPEETSGAWSHPQFEKTG	Section 5.2.2, Section 5.3.2
MGYP-61	METQIVEGFAGASHVKGAPLTTTITREVSEELLRNEIDERIVKIRPMSTPIDQISRLADARLSSSMIVDYYSVDTPPSVCKLFEGVESSQTSDIAELSVDNIAMFSPSDTILLPGVKGKAPGSCLMLYVISTGDKLRVKAINPPTENTFPALNFEQSMIRMGRAAAELDVQTSQSEALPIKRRNFCQIFKCQVEQSILQRLSAKEVGWSLTDQEETALIDMRLSMEKNFLFGARTRFEKDNTHGEVFTTEGIWTQAGKEFSYVKDKFNEEELVRLSRAAFTGNAGSSRKVLIGGSGFIEQLSMLPHVKTAGPAETVTRWGIDFTEITTKFGRLYVVLSEVFDACGHADEAMVLDPEYIQKYSHIPFKAMPIDLRSSGQRNCEAIVLTEASCIVLRYPDAHLRIVTKWSHPQFEK	Section 3.7
MGYP-11	MENNATIVNGVENGQHVVNGPLTTEVTNQASPSLLVNEIDRQIVKIRPMATPVDQLSRCAGAKRTGSMIVDYYNVDTKPTSVSLVSSSTNLAGNVRGRLKTDNNDSIDVSDTLLVEGVTGYTDAGEPDENSRLVLYVVGKDDVELMVMAVNGEPNGAGERIMPELDAGERLIRMGRAATELDVQTAQFEALPQKAQNYCQIFKMQIEQSTLQKIANKEVDWTMTDQEEAAIYDMRLGMEKNFLFGVKNKLWDSNKKETVMLTGGIWGQAGKTFTYSDGALTQDTVIDLMREAFTGNAGSKRKVLIGGSGLIGRLNKLDYTKMISAGENVAKWGIDFTEMRSKFGKLYVLLSEVFDECGMPDNGMIIDPEYLQKYSHLPFGTEALNLKSAGVRNTDALVLTEASCLVLRYPKAHVRIVMEWSHPQFEK	Section 3.7
MGYP-87	MNEMTQEPKIVEGTNGGAHVVDGPLTTTAARAASPGLLMSEIDKRVVKVRPMATPIDQISRMVGARNAKSMVVEYYSVDTKGVSARSKSMTASEESDVEGMTTYLLKTDNDGIFSATETIFVPTLSGKDASGKDAGNLQLYVLDKTDAGLKVVMLNAAGNASTLATAINGGVQIVRMGRAAAELDVQTPQFEAMPKKSSNYCQIFKAQIEQSVFAKLSAKEVGWGFSDQEEVAIMDMRMGMEKNFLFGVKARITDPKKMDEVLFTQGIWNQCGSQETLDVYNLQMADLVRIMRTAFTGDASGSNKKVFVAGSALIEAINNLEYTRVVGASQTTTRWGIDFQELRSKFGTLYVVHGEIFDQCGLEECGIIIDPEYMTKYVHVPFKVEHLDLKKSGVRNVDALVVTEASCLVLRHPKAHVRIVPAWSHPQFEK	Section 3.7
ESM_masked_T_0.5_4448	MTNLNTLKAPLTERQWSYISAVAKRIAGETSYGLKWADVLKPTGYDKNLTNVTDLIDYEKVLTKKALEVEITKENAVVKKSFDVNLDDMRALDKGKPDVDLSSLEETTNEVMALIDNYIFNGNKDTGEVGLLAYTDRTIATGTDPNALLTAITDANARFDADGVKGGKVLIISDDLWASYQATQDGSAPLDSAIKTVLDGGDIIKTPRIANALIVSKEGGTFDIVQGQKLSIGFDRYNNGAARLYIVAGYYVKVKDPKGLVRLGWSHPQFEK	Section 4.3.1
ESM_masked_T_0.5_6073	MERLKRAQAPLSEKQWELIQKRASRIAGESDFGLQFVDVGRPRKSAAIKEANLPQLDYDTGEEIHQADAERLEINQTVLLSFEVDLEELKKLDQGKPNVDLTALEKAVYALCQKINDRIFNGDEKAGTVGLLQHEERTIATGNNPDDLLAAITKARLQFDADGIDGGIVLVINRDKWAAFQEQAKGGEPLAQAVEDKLNGGRIIKTPNIQDALIVSLDGGDFELELGQDVDIGYLRQTKDAALLYIVVSFNFNVKRPQALIRLGWSHPQFEK	Section 4.3.1
ESM_masked_T_0.5_3423	MDQLKRSEAPLTEKQWELIDKVAEKIYGENNYGARFIDVEEPTGQTVEAAKEALDLEKLKEKGAVVELAKLKLGKIELEYAFEIDLEELRKLDEGKPDVDLSALEETVLNVAKEIDDVVFYGSGRAGTKGLLAYSERKIDTGVDPASLLAAITEAKAIFKKEGIIGPVVLVINKELWKKYEEETKGVAPLEKAVKEALGGGDIIKTPRIKNALIVSLKGGDFRLVLGRGLEIGFKKEVEGKVRLYIKAGYTVEVLNPDNLILLEWSHPQFEK	Section 4.3.1
ESM_masked_T_0.5_7372	MKELDIEKAPLTEEQWELIKERASDIFEQNDYGSRFVDVIEPTGEDSVEASDTDVLIESVEGSVGVKIASLGQKTIEKRFSFDVDLEELRKFSEGLPNVDLSSLEETVKQVCGYINNVIFDGSDETGSKGLLAYEERTIETGSTPEDLIKAIVEANRKFDKAGINGKKVLVINKSEWEKYKEENKGEKPLDEDVKDKLNGGEIIKTNKIDDAMIVSMSGGDFKLKMKTSVSIGYEKYNEDTVECFIEIKFSFEVLNPNSLIRLGWSHPQFEK	Section 4.3.1
ProteinMPNN_T_0.3_23	MSSLNLEKAPLTEKQKQYLADRFNKIFNENSYCLKFADVVPAQGEDFKEYPLNKVEETSSPNAKKKKGKKLTLPMIELKIPFEVPLKELQALDAGKKDVDVSNLDAATKKFCDKINDVFFYGDEKYGIKGLMQIKERTIETGDTPEELIEAIKKANKIFKEAGVNGKKVLIINKKKFEEYKEKNKGEGDIEEEIKKALEGGEIIETPKIDNALIVNLKGGNIKFYIGEPPSIGFLRENEDSVELYIYSKYGFLVENEDAIILLKWSHPQFEK	Section 4.3.1
ProteinMPNN_T_0.5_783	MPSLKLNQAPLTHEQIKLITNRFETVYKNNSYAEKISDVAPEKGSTYEAYPLNEVDWTSSDFEKYKTGTKRTLPMKELTIPFDLPLAELKKHDLGQKDVDLSNLDYATRALCREINNIFFKGDPKLGIKGLLELKERTIPTGDTVSSMLEAIKNAVQIFKELGVDGKKVLVINVDRFAQLKSKNQGNSNIESLIKSTLDGGVIIRTPEISGGLIVSLKGGHLRMNYGIRPSIGYMSTNQSSVNLYLKVKYAFEVINKDAIILLEWSHPQFEK	Section 4.3.1
ProteinMPNN_T_0.3_813	MSMLNLDKAPLTEKQKKYLEDVFNKIYNENNYSLKFVDVVPERGPDAKEFPLNSIKWTSSPLAKNKTGEKETLPFKELEIEFEIDLKEMEKLDEGKKDVDVSNLKEATKKFCKEINDLVFNGDEKYGIKGILQIKEQTIETGDTLEELLKAIKKAVEIFKKRGINGKRVLIINEDKWKKILEESKGEGDPEELIKKELEGGEIIVTPYIENAIIVSLEGNHLLFHVGLEPSIGYKEENNDKVVLYIKAKYAFLIKNPDAIIILKWSHPQFEK	Section 4.3.1
ProteinMPNN_T_0.5_593	MSELNLDKAPLTSKQKDYLTTVFNQIYNENSYSLKFADIIPATGETATAYDLNKTIDTTSPSAVNKSGTKETLPMKEIEIEFEVPLVELQALDNGTQNVDVSNLIEATKKFCAEINDLAFYGSKEIGVVGVMQLKERTIETGETPEELLAAIDAARARFKAAGVNGKLVLIINEKRYEELLSKTDGTENLEEKIREKLGGGEIIKTPYIEKGVIVSLKGNDLYFYTGLPPSIGYKKKNKNSVVLFIKSKFAFLVKNPDAIIILRWSHPQFEK	Section 4.3.1
experimental_screens_esm2_t33_650M_UR50D_0.1_masking_1	MEFLKRSFAPLTEKQWQEIDNRAREIFKTQLYGRKFVDVEGPYGWEYAAHPLGEVEVLSDENEVVKWGLRKSLPLIELRATFTLDLWELDNLERGKPNVDLSSLEETVRKVAEFEDEVIFRGIEKSGVKGLLSFEERKIECGSTPKDLLRAIVRALEIFSKDGIEGPYTLVINTDRWINFYKEEAGHYPLEKRVEECLRDGKIIKTPRIEDALVVSERGGDFKLVLGQDLSIGYEDREKDAVRLFITETFTFQVVNPEALILLKWSHPQFEK	Section 5.2.1, Section 5.2.2
experimental_screens_esm2_t33_650M_UR50D_0.1_masking_2	MEFLKRSFAPLTEKQWQEIDNRAREIFKEQLYGRKFVDVEGPYGWEYAAHPLGEVEVLSDENEVVKWGLRKVLPLIELRATFTLDLWELDNLERGKPGVDLSALEETVRKVAEFEDEVIFRGCEKSGVKGLLSFEERKIECGSTPKDLLEAVYRALSIFSKDGIEGPYTLVINTDRWINFLKEEAGHYPLEKRVEECLRGGKIITTPRIEDAIVVSERGGDFKLILGQDLSIGYEDREKDAVRLFITETFTFQVVNPEALILLKWSHPQFEK	Section 5.2.1, Section 5.2.2

experimental_screens_esm2_t33_650M_UR50D_0.1_masking_3	MEFLKRSFAPLTEKQWAEIDNRAREIFKTQLYGRKFVDVEGPYGWEYAAKPLGEVEVLSDENEVVKWGLR KSLPLIELRATFTLDLWELDNLERGKPNVDLSSLEETVRKVAEFEDEVIFRGCEKSGVKGLLSFEERKIE CGSTPKDLLEAIGRALSIFSKDGIEGPYTLVINTDRWINILKEEAGHYPYEKRVRECLRGGKIITTPRIE DALVVSERGGDFKLILGQDLSIGYEDREKDAVRLFITETFTFQVVNPEALILLKWSHPQFEK	Section 5.2.1, Section 5.2.2
experimental_screens_esm2_t33_650M_UR50D_0.1_masking_4	MEFLKRSFAPLTEKQWQEIDNRAREIFKTQLYGRKFVDVEGPYGWEYAAHPLGEVEVLSDENEVVKWGLR KSLPLIELRATFTLDLWELDNLERGNPNVDLSSLEETVRKVAEFEDEVIFRGDLKSGVKGLLSFEERKIE CGSTPKDLLEAIVRALSIFSKDGIEGPYTLVINSDRWINFLKEEAGHYPLEKRVEECLRGGKIITTPRIE DALVLSERGGDFKLILGQDLSIGYEDREKDAVRLFITETFTFQVVNPEALILLKWSHPQFEK	Section 5.2.1, Section 5.2.2
experimental_screens_esm2_t33_650M_UR50D_4_epochs_fine_tuned_0.1_masking_5	MEFLKRSFAPLTEKQWQEIENRAREVFKTQLYGRKFVDVEGPYGWEYAAHPLGKVEVLSDENEVVKWGLR KSLPLIELRATFTLDLWELDNLERGKPNVDLSSLEETVRKVAEFEDEVIFRGCEKSGVKGLLSFEERKIE CGSTPKDLLEAVVRALQIFRKDGKEGPYTLVINTDRWINFLKEEAGHYPLEKRVEKCLRGGKIITTPRIE DALVVSERGGDFKLILGQDLSIGYEDREKDAVRLFITETFTFQVVNPEALILLKWSHPQFEK	Section 5.2.1, Section 5.2.2
experimental_screens_esm2_t33_650M_UR50D_4_epochs_fine_tuned_0.1_masking_6	MEFLKRSFAPLTEKQWQEIDNRAREIFKTQLYGRKFVDIEGPYGWEYAAHPLGEVEVLSDENEVVKWGLR KSLPLIELRATFTLDLWELDNLERGKPNVDLSSLEETVLKVAEFEDEVIFRGCEKSGVKGLLSFEERKIE CGSTPKDLLEAIVRALSIFSKDGIEGPYTLVINTDRWINFLKEEAGHYPLEKRVEEMLRGGKIITTPRIE DALVVSERGGDFKLILGQDLSIGYEDREKDAVRLFITETFTFQVVTPEALILLKWSHPQFEK	Section 5.2.1, Section 5.2.2
experimental_screens_esm2_t33_650M_UR50D_4_epochs_fine_tuned_0.1_masking_7	MEFLKRSFAPLTEKQWQEIDNRAREIFKTQLYGRKFVDVEGPYGWEYAAHPLGEVEVLSDENEVLKWGLR KSLPLIELRATFTLDLDELDNLERGKPNVDLSSLVETVRKVAEFEDEVIFRGCEKSGVKGLLSLEERKIE CGSTPKDLLEAIVRALSILSKDGIEGPYTLVINTDRWINFLKEEAGHYPLEKRVEEKLRGGKIITTPRIE DALVVSERGGDFKLILGQDLSIGYEDREKDAVRLFITETFTFQVVNPEALILLKWSHPQFEK	Section 5.2.1, Section 5.2.2
experimental_screens_esm2_t33_650M_UR50D_4_epochs_fine_tuned_0.1_masking_8	MEFLKRSFAPLTEKQWQEIDNRAREVFKTQLYGRKFVDVEGPYGWEYAAHPLGEVAVLSDEKEVVKWGLR KSLPLIELRATFTLDLWELDNLERGKPNVDLSSLEEAVRKVAEFEDEVIFRGCEKSGVKGLLNFEERKIE CGSTPKDLLEAIGRALSIFSKDGIEGPYVLVINTDRWINFLKEEAGHYPLEKRVEECLRGGKIITTPRIE DALVVSERGGDFKLILGQDLSIGYEDREKDAVRLFITESFTFQVVNPEALILLKWSHPQFEK	Section 5.2.1, Section 5.2.2
ESM-IF_experimental_screen_1	MAFLKRSFAPLTARQWELIDARAREIIKENLYGRKFVDVLGPKGEDFKAYPLGEIDVTSGEDEVFQWGSR VTLPLIELRLTFDVDLWDLDLLDRGLPNVDLSELEKTTLDLANAEDDVIYNGCKKSGVKGLMAYKEREIS CGSDPKALLEAITKARAKFAEDGIRGPYVLVINEDLWKKYQEKTKGHYPLAKAVEDKLKGGRIIKTPRIK DALIVSERGGDFYLIEGQDLSIGYKKRTKGKVRLFMREIFTFYVKNPEALIRLGWSHPQFEK	Section 5.2.1, Section 5.2.2
ESM-IF_experimental_screen_2	MDFLDRDFAPLSAAQWELIDARAREISKENLYGRKFVDVEGPAGADFKAYPLGKVNVTSGEDEVIRWGSR AARPLIELRKTFDIDLWELDLLDRGLPNVDLSELEKTTLDLAEAEDDVIFNGCSKSGVVGLMAYTDRTIS CGDDPAALLAAITKARAKFAEDGIKGPYVLVINESLWESYQAKNAGHYPLADAVKTELNGGDIIKTPRID DALIVSKRGGDFYLIEGQDLSIGYNRRSDGKVELFMEEKFTFEVRNPEALIRLGWSHPQFEK	Section 5.2.1, Section 5.2.2
ESM-IF_experimental_screen_3	MEFLNREFAPLTEAQWELIDRRAREISKENLYGRKFVDVLGPKGADFKAYPLGAIDVTSGEDEVIKWGSR VVRPMIELRLTFAVDLWELDLLERGLPNVDLSELEKTTLAFAKAEDDVIFYGCKESGVKGLMAYKDRTIS CGDDPESLLKAITTARARFAEDGIRGPYVLVINEELWSEYKRKTEGHYPLEEAVKRELDGGDIIKTPRIK DALIVSKRGGDFFLVEGQDLSIGYKRREEGKVLLFMEEKFTFEVKNPEALIRLGWSHPQFEK	Section 5.2.1, Section 5.2.2
ESM-IF_experimental_screen_4	MAFLNRAFAPLTEAQWRLIDARAREIAKENLYGRKFVDVLGPKGRSAKAYPLGAIKVTSGEDEVLKWGDR VVRPLIELRLTFDVDLWELDQLDRGKPNVNLDELEKTTLALAKAEDAVIYYGCKKSGVKGLMAYKERTIA CGTDPEALLKAITTARAKFKKDGIRGPYVLVINTDLWAEYQAKTAGHYPLEEAVKEKLDGGDIIKTPRID DALIVSKRGGDFYLVEGQDLSIGYDRRVKGKVRLFMREIFTFEVLNPEALIRLGWSHPQFEK	Section 5.2.1, Section 5.2.2
ESM-IF_experimental_screen_5	MSFLDRSFAPLTKKQWELIDARAREIAKQNLYGRKFVDVEGPKGRDFKAYPLGEVAVTSGEDDVIVWGDR VVQPLIELRLTFDVDLWELDLLDRGLPNVNLDNLEAATLDLADAEDDVIFYGCEESGVTGLMAYEERTIA CGDNPDALLDAITKARAKFAEDGIKGPYVLVINEDLWAAYQEQNVGHYPLADAVKARLNGGDIIKTPRIT DALIVSKRGGDFKLIEGQDLSIGYDRRVDGSVRLFMEEKFTFLVKNPEALIRLGWSHPQFEK	Section 5.2.1, Section 5.2.2
ESM-IF_experimental_screen_6	MKFLNRQFAPLTAKQWELIDARAREIAKENLYGRKFVDVEGPQGSLFKAYPLGEVDITSGEYEVEQWGTR KTLPLIELRLTFEVDLWELDQLDRGKPNVDLSNLEATTLALAKKEDKVIFEGCKASGVVGLMAHEERTIS CGTDPAALLTAITTARAKFAEDGIKGPYVLVINEDLWASYQAANDGHYPLADAVKTNLAGGSIIKTPRIK DALIVSLRGGDFFLIEGQDLSIGYKRRVNGKVELFMEEKFTFLVKNPEALIRLKWSHPQFEK	Section 5.2.1, Section 5.2.2
ESM-IF_experimental_screen_7	MKFLNRKFAPLSAEQWELIDKRAREIAKNSLYGRKFVDVLGPKGSDFKAEPLGAVNVTSGELEVLKWGDR AELPLIELRLTFDIDLWDLDLLDRGLPNVNLEALEAATLAFADAEDEVIFNGCAESGVVGLMDHEARTIS CGDDPSALLSAITNARLRFAEDGIRGPYVLVINEDLWAAYQASNVGHYPLEDAVKENLNGGDIIKTPRIN DALIVSLRGGDFFLVEGQDLSIGYLRRNEGSVSLFMEERFTFKVKNPEALIRLGWSHPQFEK	Section 5.2.1, Section 5.2.2
ESM-IF_experimental_screen_8	MKFLKRDFAPLSEKQWELIDARAREIAKENLYGRKFVDVEGPKGADFEAEPLGAVEITSGEDEVIRWGKR AARPLIELRKTFAVDLWELDLLERGKPNVNLSELEKTTLDLAATEDDVIFNGCKTSGVVGLMQWEERKIT CGTDPKALLDAITKARAKFDEDGIKGPYVLVINKDLWKAYLKKTKGHYPLADAVKKHLKGGRIIETPRIS DALIVSLRGGDFKLVEGQDLSIGYLTRVKGKVKLFMEERFTFEVVNPEALIRLGWSHPQFEK	Section 5.2.1, Section 5.2.2
ESM-IF_experimental_screen_9	MEFLNREFAPLSELQWSLIDARAREIYKESLYGRKFVDVEGPKGVDFSAYPLGKVNVTSGEDEVIRWGSR ETLPLIELRFTFDIDLWELDLLERGEPNVNLSNLEKTTLELAKAEDNVIFYGCETSGVKGLMEYEERKIA CGTDPKALLAAITKARARFAEDGIRGPYVLVINENLWAAYQAKTQGHYPLSEAVKEKLSGGEIIKTPQIK DALIVSKRGGDFKLVEGQDLSIGYEKRVNGKVRLFIQEKRTFLVVNPEALIRLGWSHPQFEK	Section 5.2.1, Section 5.2.2
ESM-IF_experimental_screen_10	MVFLNREFAPLSEKQWELIDQRAREISKENLYGRKFVDVLGPQGADFKAYPLGEVEITSGEDEVLVWGKR KVLPLIELRKTFEVDLWDLDLLDRGLPNVDLSELEKTTLDVAKKEDEVIFYGCEESGVVGLMEYEDRRIE CGKDPEALLKAITKAIEKFREDGIEGPYVLVINEELWAEYLKRSIGHYPLEEAVKEKLKGGSIIKTPRIK DALIVSLRGGDFFLVEGQDLSIGYLRRKDGKVALFIEEKFTFLVKNPEALIRLGWSHPQFEK	Section 5.2.1, Section 5.2.2
ESM-IF_experimental_screen_11	MLFLKRDFAPLTERQWSLIDARAREIAKENLYGRKFVDVEDPQGEDFRAYPLGEVDVTSGEDEVIRWGTR KALPLIELRLTFDIDLWELDQLDRGLPNVNLEALEVTVKQFADKEDTVIYYGCEESGVVGLMAHAEREIE CGDTPGALLEAITKARDRFAEDGIRGPYVLVINEDLWKKLQEKNKGHYPLADAVKEKLDGGDIIRTPRIK DALIVSTRGGDFYLHVGQDLSIGYSKRTRDSVRLFMKEKFTFEVKNPEALIRLGWSHPQFEK	Section 5.2.1, Section 5.2.2
ESM-IF_experimental_screen_12	MKFLRRDFAPLSERQWNLIDARAREISKENLYGRKFVDVLDPKGIDAKAYPLGEIKVTSGELDVDIWGER VTQPLIELRLTFDIDLWELDLLDRGLPNVDLSELEKTTKAFAKAEDNVIYNGCKKSGVVGLMEHEDRTIT CGETPAALLNAITTARARFAEDGIKGPYVLVINKDLWAKYQAKTNGHYPLAEAVKTNLNGGSIIETPRIT DALIVSTRGGDFELVEGQDLSIGYSRRTDGKVSLFMVEKFTFIVKNPEALIRLGWSHPQFEK	Section 5.2.1, Section 5.2.2
ESM-IF_experimental_screen_13	MSFLDRSFAPLSEAQWSYIDARAREIAKENLYGRKFVDVGGPQGVDASAYPLGEVDITSGEDEVIKWGTR KARPLIELRFTFDIDLWELDLLDRGKPNVDLSNLEKTTLDLANEEDDVIFYGCKKSGVVGLLSYKERTIE CGDDPEALLDAITSALDKFAADGIRGPYVLVINQDLWKEYQEKNKGHYPLEDAVKKRLNGGDIITTPRIS DALIVSKRGGDFYLVEGQDLSIGYEKRNDGKVSLFMREIFTFEVKNPEALIRLGWSHPQFEK	Section 5.2.1, Section 5.2.2

ESM-IF_experimental_screen_14	MDFLNRAFAPLTEKQWSLIDARAREILKEELYGRKFVDVLGPRGFDFAAYPLGEVDITSGENEVNVWGGR KVRPLIELRLTFSVNLWELDLLDRGLPNVDLSELEKTTRQFAAIEDNVIFYGCDVSGVKGLTAFSEQEIE CGTDPEALLEAITKAREKFDEDGIKGPYVLVINKKLWESYKEKTKGHYPLEEAVKEKLRGGDIIKTPRIE DALIVSKRGGDFDLVLGQDLSIGYLRRVDGSVDLFMVERFTFEVVNPEALIRLKWSHPQFEK	Section 5.2.1, Section 5.2.2
ESM-IF_experimental_screen_15	MTFLNRDFAPLTAAQWQLIDSRAREIYKKQLYGRKFVDVEAPKGFNFKAYPLGEIEVTSSEDDVVIWGNR VARPLIELRLTFDVDLWELDKLDRGKPNVDLSALDKTTLQLANAEDDVIYHGCSASGVKGLMSYKDRKCG TDPAALLDAITQANAKFAEDGIDGPYVLVINTDLWQKYQAETKGHYPLADAVKEKLNGGSIIQTPRIDDA LIVSTRGGDFYLHEGQDLSIGYERRVNGNVRLFMEEKFTFEVVNPEALIRLAWSHPQFEK	Section 5.2.1, Section 5.2.2
ESM-IF_experimental_screen_16	MSFLKRDFAPLTDLQWELIDARAREISKDNLYGRKFVDVEGPRGIDAKAYPLGAIDVTSGEMDVIRWGTR KARPLIELRLTFDVDLWDLDLLDRGKPNVDLSALEATTKALAKSEDDVIFYGCDKSGVTGLMAHTDRTIA CGDDPESLLEAITDARTRFAEDGIDGPYVLVINEDLWAAYRDEAEGHYPLAEAVKDFLRGGSIIATPRIS DALIVSERGGDFELVEGQDLSIGYSRRIRGRVRLFMRESFTFEVRNPEALIRLGWSHPQFEK	Section 5.2.1, Section 5.2.2
protein_LM_mutated_experimental_screen_1	MANLPRSLAPLTEEAWRVIDEEARRTIKRYLAGRRLVDVYAPTGPGASAIEVGIDDVEVSPDGGTVAVTR KTVPVVELRKPFTLTRWDVDSVERGGRDLDLEALREAAGKVAEAEDKAVLEGYEKAGIKGLTNLADAKIK LPTDVDGLPSSISNAVAELEKAGVTGPYVLVLSPETYAKLKDALKTGYPLKELIEESLGKARVIVSSKVT GSLLIRARGDDFTLVPGQDLSIGYLKREGGKVTLYVSEAFTFTLRTAEAVVELSWSHPQFEK	Section 5.2.1, Section 5.2.2
protein_LM_mutated_experimental_screen_2	MTNLRRYLAPLTEEAWKKIDEEARRTAKKHLAARRIVDVEGPLGPGTVAIVSGTAEFELDPDGGPKATSR KTVPIVELRVPFTLTRWELDAVQRGRLDLLEKKVKDAAEKIARAEDKAIIEGYEKAGIKGLKNVKPETIK LPASPTDVPNAITKAVATLESAGVVGPYVLVLSPEVYATILASLKGGYPIKEYLKELLGEGEVITSSEIS GIVAISKRGDDFRLVIGQDLSIGYLKREGENVVLYVSESYTFAVLTPEAVVKLSWSHPQFEK	Section 5.2.1, Section 5.2.2
protein_LM_mutated_experimental_screen_3	MSNLLRKLAPLPKEAWTKIDEEARRTVKSKLAARRIVDVYTPPGGESVAIPVGGELEKVKPGDGVSASSR VRVPLVELRKPFTLTRWQLDAVKRGAKDLDVNVLENAATQVAKKEDDAILNGYEEAGIKGLRKVEGNEVS LPSDPERIPEDTIKAIEKLEKAGVTGPYVLVLSPTKYAKLLKAGTEGYPVAEYVKDIIGKGKVVSDSTVD GGVLISQRGQDFKLTIGQDLSIGYIGRKDGKVELYVVESLTFTVTTKEAVVTLSWSHPQFEK	Section 5.2.1, Section 5.2.2
protein_LM_mutated_experimental_screen_4	MANLLRYLAGLTKEAWETIDKEARRTLKKYLAARRLVDVEYPVGSAGVALEAGVITTEVTPKDGTKAVTR KVVPLVELRFPFTLTRWDLDAVARGESDITLSDVEELAETIAEAEDSAIIDGYPEAGISGIKNLPENTVE LPANVKEIPGKLIEAKAVLESAGVPGPYTLVLSADDYLSILSEVKAGYPLKDVVKELLGSGEVVTSSVTE GGVLVSSRGTDFRLLIGQDLSIGYLKREDGKVTLYGIEYITFEVLTTEAIVVLKWSHPQFEK	Section 5.2.1, Section 5.2.2
protein_LM_mutated_experimental_screen_5	MANLERKLAELTAEAWLQIDDEARRTLKETLAVRRVVDVVGPLGAGTKAVKIGKTTVEIDPEDGKPAVER EEVPIVELRVPFPVTRWEIDAVNRGLSDLDLTPLEDAAKEVARTEDKAIISGYKTAGITGLTELSKVTVS LPSDPSELPDAIINAAEKLKDAGVSGPYVLVLSPSVYRKLHKLEKAGYPLLDSVKEILGGSRIVVSSEIT GPVIITPRGPDFRLVIGQDLSIGYLARSDGNVVLYVSEVFTFEIKTPEAAVTLSWSHPQFEK	Section 5.2.1, Section 5.2.2
protein_LM_mutated_experimental_screen_6	MANLLRDLAALTEEAWKVIDSEARRTLKEYLAARRVVDVIEPPGGTGYAFVKGEIKTVLDPEGGPKAVKR TLTPLVELRVPFELTRWEIDAVQRGEPDPDISTLENAAKEIAKAEDEAIINGYEKAGIKGLKNLNSVKKT LPSSAESVPSEIAKAVGALKSAGVSGPYTLVLSPDLYEKLISAVKGGYPLEDYILSLVKDGRVIVERSIK GALLVSSRGEDFTLAIGQDLSIGYVDREGDKVRLYVVETATFGVVTPEATVVLTWSHPQFEK	Section 5.2.1, Section 5.2.2
ProteinMPNN_experimental_screen_1	MSFLRREFAPLTEKQWQYIDERAREIAKNNLYSRKFVDVVPPRGPDAKAYPLGEIEETSPETAVNKWGKR KTLPFIELRETFEVPLWELDELDRGKENVNLSNLDEAVKKMADKEDDVVFNGCEESGVKGLLQLKERRIP CGDTLEELLAAIKKAVERFKKDGIDGPYVLVINEERWEKIKETTKGHYPPEEAIKEALEGGEIIKTPKIK DGLIVSKRGGDFLLYVGQDLSIGYLRRNEDSVELFLYEKFTFLVKNPEAIILLNWSHPQFEK	Section 5.2.1, Section 5.2.2
ProteinMPNN_experimental_screen_2	MSFLNRKFAPLTEKQWEYIDKRAREIAKNNLYSRKFVDVVPPLGPDAKAYPLGEIEWTSPETAVNKWGTR KTLPFIELRITFDLPLWELDKLDRGEENVDLSNLDKAVLEMAKKEDDVVFYGCKESGVVGLMQLKERTIP CGDTPEDLLAAIRRAVEIFREDGIDGPYVLVINKKRWEEYLKKTKGHYPLEEAVKEALEGGEIIVTDRIE DGLIVNKRGGDFYLHVGQDLSIGYESRNEDSVNLFLYEKFTFEVKNPEAVILLNWSHPQFEK	Section 5.2.1, Section 5.2.2
ProteinMPNN_experimental_screen_3	MSFLRRKFAPLTKKQWEYIDKRAREIAKKNLYSRKFVDVVPPLGPDATAYPLGEVEETSPETAVNKWGTR KTLPFIELRKTFDVPLWELDKLDRGEKNVDLSNLDKAVLEMAKEEDDVVFNGCEESGVKGLLQLKERTIP CGDTPEELLAAIKEAVARFKEDGIDGPYVLVINKERWEKYLEKTKGHYPLEEAVKEALEGGEIIKTPRIK DGLIVNLRGGDFLLHVGQDLSIGYESRNKDSVRLFLYEKFTFLVKNPEAVIILNWSHPQFEK	Section 5.2.1, Section 5.2.2
ProteinMPNN_experimental_screen_4	MSFLRREFAPLTEKQWEYIDKRAREIAKNNLYSRKFVDVVPPRGPDAKAFPLGEVEWTSAETDVNKWGTR KTLPFIELRITFNVPLWELDELDRGKKNVDLSNLDKATLEMAKKEDDVVFYGCKESGVKGLLQFKERTIP CGDTVEDLLNAIKEAVERFKKDGIDGPYVLVINKERWEKYLEKTKGHYPLEEAVKEALEGGEIIKTPRIK DGLIVNKRGGDFLLHVGQDLSIGYLRRNEDSVELFLYEKFTFLVKNPEALILLNWSHPQFEK	Section 5.2.1, Section 5.2.2
ProteinMPNN_experimental_screen_5	MSFLRREFAPLTEKQWEYIDKRAREIAKKNLYSRKFVDVVPPRGPDAKAYPLGEVLETSPETAVDKWGER KTLPFIELRITFDLPLWELDELDRGKENVDLSNLDKAVKEFAKKEDEVVFYGCKESGVVGILQLKERRIP CGDTPEELLEAIKEAIERFKEDGIDGPYVLVINKERWEKIKEKTKGHYPIEEKVKEALEGGEIIKTPYIK DGLIVNKRGGDFLLHVGQDLSIGYLERNEDSVRLFLYEKFTFEVVNPEAVILLNWSHPQFEK	Section 5.2.1, Section 5.2.2
ProteinMPNN_experimental_screen_6	MSFLRREFAPLTEKQWEYIDKRAREIYKNNLYSRKFVDVVPPQGPDAKAYPLGEVEWTSPETAVNKWGKR KTLPFIELRKTFSLPLWELDELDRGKKNVNLSNLDKAVKEMADEEDKVVFYGCKESGVKGLLQFKERTIE CGDTLESLLEAIKKAIERFKEDGIDGPYALIINEKKWEKILETSKGHYPPEEKVKEALEGGEIIKTPKIE DGLIVNLRGGDFLLHVGQDLSIGYESRDETSVNLFLYEKFTFLVKNPEALILLKWSHPQFEK	Section 5.2.1, Section 5.2.2
ProteinMPNN_experimental_screen_7	MSFLRREFAPLTEKQWAYIDERAREIAKKNLYSRKFVDVVPPRGPDAKALPLGEVLETSPETAVRRWGER RVLPFIELRITFSVPLWELDALDRGKENVDLSNLDKATLEFAEREDRVVFYGCEESGVVGLMQLKERTIP CGDTLESLLEAIRRAVERFKEDGIDGPYVLVINEERWKEILKTTKGHYPAEERIKEALEGGEIIKTPRIK DGLIVNLRGGDFELHVGQDLSIGYESRDEDSVRLFLYEKFTFRVVNPEAVIILNWSHPQFEK	Section 5.2.1, Section 5.2.2
ProteinMPNN_experimental_screen_8	MSFLRREFAPLTKEQWEYIDKRAREIYKNNLYSRKFVDVVPPQGPDFTAYPLGEVEWDSPETAVNKWGRR KTLPMIELRITFDLPLWQLDALDRGEKNVDLSNLDKATLEFADKEDDVVFYGCKESGVVGLMQLKERTIP CGKTPESLLEAIRRAREIFREDGIDGPYVLVINEKRWEELEKKTKGHYPLEEAIKEELEGGEIIKTPKIE DGLIVNKRGGDFLLVVGQDLSIGYESRNEDSVRLFLYEKFTFLVKNPEAIIRLNWSHPQFEK	Section 5.2.1, Section 5.2.2
ProteinMPNN_experimental_screen_9	MSFLRREFAPLTEEQWQYIDDRAREIAKNNLYSRKFVDVVPPQGPDAKAYPLGEVKWTSPETAVNKWGER VTLPFIELRITFELPLWELDELDRGKKNVDLSNLDKATLEFAKREDEVVFYGCEESGVVGLLQIKERTIP CGDTVEELLAAIKEAIARFKKDGIDGPYVLVINEKKWEELLEKTKGHYPLEEAIKEALEGGEIIKTPYIE DGLIVNKRGGDFLLHVGQDLSIGYERRTESSVQLFLYERFTFLVKNPEAIILLKWSHPQFEK	Section 5.2.1, Section 5.2.2
ProteinMPNN_experimental_screen_10	MSFLRREFAPLTKKQWEYIDKRAREIAKNNLYSRKFVDVVPPRGPDATAYPLGEVEEDSPETAVRKWGRR KTLPFIELRITFDLPLWELDRLDRGEKNVDLSNLDKAVKEFADKEDDVVFYGCKESGVVGLLQLKERTIP CGDTLEELLAAIKRAVERFKEDGIDGPYVLVINKERWEKILKTTKGHYPAEEKIKEALEGGEIIVTPKIK DGLIVSKRGGDFLLHVGQDLSIGYLERNEDSVRLFLYEKFTFEVKNPEAVILLNWSHPQFEK	Section 5.2.1, Section 5.2.2
ProteinMPNN_experimental_screen_11	MSFLRREFAPLTEKQWEYIDKRAREIYKNNLYSRKFVDVVPPQGPDATAYPLGEVEWTSPETAVNKWGTR KTLPFIELRKTFELPLWKLDELDRGEENVDLSNLDKATLEMAKEEDRVVFYGCEESGVKGILQLKERRIP CGDTFEELLKAIKEAIERFKKDGIDGPYALIINEERWKKILATTKGHYPPEEAVKEALEGGEIIKTPYIK DGLIVSLRGGDFLLHVGQDLSIGYESRNEDSVKLFLYEKFTFLVKNPEAIILLNWSHPQFEK	Section 5.2.1, Section 5.2.2

ProteinMPNN_experimental_screen_12	MSFLRREFAPLTEEQWEYIDKRAREIAKKNLYSRKFVDVVPPQGPEARAYPLGEVEWTSPETAVRKWGRRKTLPFIELRKTFEVPLWELDELDRGKKNVDLSNLDKATLEMAKEEDDVVFYGCKESGVKGILQLKERTIPCGDTPESLLEAIKKAVEIFKKDGIKGPYVLVINEERWEQYLKQTDGHYPLEEAVKEALEGGEIIKTPRIKDGLIVSKRGGDFLLHVGQDLSIGYLRRNEKSVELFLYEKFTFLVKNPEAIILLNWSHPQFEK	Section 5.2.1, Section 5.2.2
ProteinMPNN_experimental_screen_13	MDFLRRKFAPLTEKQWEYIDERAREIAKKNLYSRKFVDVVPPRGPDAKAYPLGEVEWDSPETAVNKWGRRKTLPFIELRKTFSLPLWELDALDRGKKNVDLSNLDKAVLEMAKEEDDVVFYGCEESGVVGLMQLKERTIPCGDTFEELLEAIKRAVAIFKEDGIDGPYALVINEKRWEEIKKQTKGHYPPEEAVKEALEGGKIIKTPRIEDGLIVNLRGGDFLLYVGQDLSIGYLRRDKESVELFLYEKFTFKVVNPEAIIRLKWSHPQFEK	Section 5.2.1, Section 5.2.2
ProteinMPNN_experimental_screen_14	MSFLNREFAPLTEKQWEYIDKRAREIYKNNLYSRKFVDVVPPQGPDATAYPLGEVEETSPETAVNKWGTRKTLPFIELRITFNVPLWELDELDRGKENVNLSNLDKAVKEFADKEDDVVFNGCKESGVKGLLQLKERTIPCGETFEELLEAIKRAVERFKEDGIDGPYVLVINKELWEKILETTKGHYPPEEAVKEALEGGEIIKTPKIKDALIVNLRGGDFYLYVGQDLSIGYLSRNEDSVELFLYEKFTFLVKNPEAIILLNWSHPQFEK	Section 5.2.1, Section 5.2.2
ProteinMPNN_experimental_screen_15	MSFLKREFAPLSKKQWEYIDKRAREIAKNNLYSRKFVDVVPPLGPEAKALPLGEVEELSAETDVFKWGRRKVLPFIELRITFDVPLWELDELDRGKENVDLSNLDKATLEMAKKEDDVVFYGCKESGVTGVLQLKERTIPCGDTPEDLLAAIKRAIAIFKEDGIDGPYALIINEKKWKELLEKTKGHYPLEERIKEALEGGKIIKTPRIEDGLIVNLRGGDFLLHVGQDLSIGYESRNKDSVRLFLYEKFTFKVVNPEAVILLNWSHPQFEK	Section 5.2.1, Section 5.2.2
ProteinMPNN_experimental_screen_16	MSFLNREFAPLTEKQWEYIDKRAREIYKNNLYSRKFVDVVPPLGPDATAYPLGEIEWVSGEDDVVKWGKRKTLPFIELRITFSVPLWELDELDRGKENVDLSNLDKATLEFADKEDDVVFNGCKESGVVGLLQLKERRIPCGDTLESLLAAIKEAVERFKKDGIDGPYVLVINKKRWEQILATAKGHYPPEELVKEALEGGEIIRTDKIEDGLIVNLRGGDFLLIVGQDLSIGYLRRNEDSVELFLYERFTFLVKNPEALILLNWSHPQFEK	Section 5.2.1, Section 5.2.2
CmEncap	MNKSQLYPDSPLTDQDFNQLDQTVIEAARRQLVGRRFIELYGPLGWEYAAHPLGEVEVLSDENEVVKWGLRKSLTIPMLYKDFVLYWRDLEQSKALDIPIDFSVAANAARDVAFLEDQMIFHGSKEFDIPGLMNVKGRLTHLIGNWYESGNAFQDIVEARNKLLEMNHNGPYALVLSPELYSLLHRVHKDTNVLEIEHVRELITAGVFQSPVLKGKSGVIVNTGRNNLDLAISEDFETAYLGEEGMNHPFRVYETVVLRIKRPAAICTLIDPEEWSHPQFEK	Section 5.3.1, Section 5.3.2
7MU1_Qt_E-loop	MEFLKRSFAPLTEKQWQEIDNRAREIFKTQLYGRKFVDVEGPYGRGMQSVFNDIFMESHEAKMDFQGSFDTEVESSRRVNYPLIELRATFTLDLWELDNLERGKPNVDLSSLEETVRKVAEFEDEVIFRGCEKSGVKGLLSFEERKIECGSTPKDLLEAIVRALSIFSKDGIEGPYTLVINTDRWINFLKEEAGHYPLEKRVEECLRGGKIITTPRIEDALVVSERGGDFKLILGQDLSIGYEDREKDAVRLFITETFTFQVVNPEALILLKWSHPQFEK	Section 5.3.1

***Table 7.4:* Opentron OT-2 Automation Experiment Details**
The aim and protocol of each experiment is shown along with the results and any notes. Experiments are numbered in chronological order.

Experiment/Aim	Protocol	Colonies?	Notes
Experiment 1 – Establish Fully Automated Protocol	Run Type IIS reaction in OT-2 thermocycler Mix with competent cells Heat shock and outgrowth in temperature module Pipette outgrowth onto agar plates	N/A	Run aborted - thermocycler temperatures were set incorrectly, needed fixing
Experiment 2 – Update Fully Automated Protocol	Same as **Experiment 1** but with fixed thermocycler settings Plating of outgrowth mixtures was attempted both with the OT-2 and by hand	No	Pipetting errors when plating outgrowth mixture with the OT-2 – volume discrepancies and air bubbles were seen. No liquid was seen deposited onto the agar plate so manual plating was also attempted. No colonies observed even when cells were plated manually.
Experiment 3 – Manual Plating Optimization	Entire protocol was carried out manually, but using the same volumes and conditions as used in OT-2 protocols from **Experiments 1 and 2.** Type IIS reactions assembled, run in benchtop thermocycler, mixed with competent cells, heat shock and outgrowth (in water bath and thermomixer respectively), and then plated in varying volumes and dilutions	Yes	Results shown in Figure 5.2.3

Experiment 4 – Optimizing Plating Distance and Pipetting	Plated 10 µl of orange loading dye onto an agar plate with the OT-2 at varying distances from the agar surface, with or without blowout step Labware calibration was performed such that the pipette tip was placed at the very surface of the agar	N/A	All distance settings appeared to work with or without blowout, but blowout steps gave fewer volume errors Pipetting errors occur when liquid level runs low in source wells
Experiment 5 – Testing the Fully Automated Protocol with new Plating Settings	Same protocol as in **Experiment 2** but with new plating settings from **Experiments 3 and 4**	No	Liquid is transferred to agar plate correctly, but no colonies were seen This indicates a problem with thermocycling, heat shock, or outgrowth
Experiment 6 – Investigating OT-2 Heat Shock and Outgrowth Steps vs Manual	Manually assembled Type IIS reactions and ran in benchtop thermocycler Transform reactions as in **Experiment 5** with the OT-2	N/A	Run aborted after thermocycling – Type IIS reactions all evaporated from the 96 well plate in the thermocycler
Experiment 7 – repeating **Experiment 6** with plate seals	Same conditions as **Experiment 6** but plate was sealed with foil plate seal to prevent evaporation	No	Some evaporation still seen from plate – foil seals used incorrectly
Experiment 8 – Optimizing Silicone Oil Pipetting with the OT-2	Tested pipetting silicone oil from a reservoir into a 96-well plate using the OT-2, and optimized settings recommended by the manufacturer: 64.75 µl/s aspiration and dispense rate, 4 µl/s blowout rate, withdraw pipette from oil at 1 mm/s, delay 8 seconds after aspirating or dispensing, touch tip on sides of well before dispensing liquid aspirated from underneath the oil layer	N/A	Settings worked well, with no visible volumetric errors, air bubbles, or missing liquids
Experiment 9 – Testing Type IIS Assembly with Silicone Oil Seals	Same conditions as **Experiment 6** but with 135 µl of silicone oil added on top of Type IIS reaction mixtures to prevent evaporation	No	No evaporation seen after thermocycling, so issue must lie in heat shock, outgrowth, or with adding silicone oil to Type IIS mixes
Experiment 10 – Establishing a Semi-Automated Type IIS Assembly Protocol	Assemble Type IIS reactions and add silicone oil using the OT-2, seal with foil seal Run reactions in benchtop thermocycler Mix reactions with competent cells using the OT-2 Heat shock in a water bath manually and return to OT-2 for recovery at 4 ˚C and addition of outgrowth medium Seal with breathable plate seal and carry out outgrowth in a benchtop thermomixer	Yes	Results shown in Figure 5.2.3 Despite optimization, volumetric errors and air bubbles were still seen in some Type IIS reactions with silicone oil Attempts to transfer only the aqueous layer of these wells into a fresh plate were unsuccessful

	Return to OT-2 for final plating		
Experiment 11 – Optimizing the Semi-Automated Type IIS protocol	Same as **Experiment 10** but with no silicone oil, only foil seals	Yes	Results shown in Figure 5.2.3 Detailed protocol in Sections 2.3.2.4 and 2.4.1
Experiment 12 – Transforming Purified Plasmid DNA with the OT-2	Transformed pure plasmid stocks into T7-Express for protein expression, using the same OT-2 protocol as in **Experiment 11** (transformation steps only, no DNA assembly or thermocycling required) Purified plasmid stocks were diluted 5x before transformation -amounts of DNA transformed range from 47.5 ng to 142.5 ng 11 plasmids were tested: MGYP-61, MGYP-11, MGYP-87, ESM_masked_T_0.5_6073, ESM_masked_T_0.5_3423, ESM_masked_T_0.5_7372, ProteinMPNN_T_0.3_23, ProteinMPNN_T_0.5_783, ProteinMPNN_T_0.3_813, TmEncap, and pSB1C3-FB	Yes	Only 1/11 plasmids gave colonies after 24 hours at 37 ˚C, and a further 3 plasmids gave colonies after another 24 hours incubation. Colonies were too small and too close together, and after the second 24 hour incubation some colonies had started to grow into each other.
Experiment 13 – Optimizing OT-2 Transformation of Purified Plasmid DNA	Same conditions as **Experiment 12**, but only used two plasmids: ESM_masked_T_0.5_7372 and pSB1C3-FB Tested a range of dilutions of the plasmid stocks: 10x, 50x, 100x, 200x Amounts of DNA transformed range from 4.75 to 142.5 ng	Yes	All dilutions gave colonies but 100x dilution gave the best number, size, and spacing of colonies These conditions were used to transform purified plasmid DNA for protein expression in future experiments

Table 7.5: **Sanger Sequencing of Designed Encapsulin Sequences**
For each design, Sanger sequencing results are briefly described, and filenames for forward and reverse primer sequencing data are provided. Sequencing data files for each forward and reverse primer filename (in .ab1, .seq, .scf, and unclipped .seq formats) are available at https://github.com/naailkhan28/encapsulin_design/tree/master/experimental_screen/sequencing_data . "Verified" indicates that sequencing data aligned to the DNA template, and "Incorrect" sequencing indicates missing sequence regions, either the Strep-Tag II or other parts of the protein coding sequence. "Inconclusive" indicates poor quality Sanger sequencing data, and "N/A (no colonies)" means that designs failed to give colonies following site-directed mutagenesis to add the Strep-Tag II sequence.

Design Name	Sequencing Status	Sequencing Data Filename (Forward Primer)	Sequencing Data Filename (Reverse Primer)
experimental_screens_esm2_t33_650M_UR50D_0.1_masking_1	Incorrect	A1_NKK153-A01	A1_NKK160-A01
experimental_screens_esm2_t33_650M_UR50D_0.1_masking_2	Verified	B1_NKK153-B01	B1_NKK160-B01
experimental_screens_esm2_t33_650M_UR50D_0.1_masking_3	Verified	C1_NKK153-C01	C1_NKK160-C01
experimental_screens_esm2_t33_650M_UR50D_0.1_masking_4	Verified	D1_NKK153-D01	D1_NKK160-D01
experimental_screens_esm2_t33_650M_UR50D_4_epochs_fine_tuned_0.1_masking_5	N/A (no colonies)	E1_NKK153-E01	E1_NKK160-E01
experimental_screens_esm2_t33_650M_UR50D_4_epochs_fine_tuned_0.1_masking_6	Verified	F1_NKK153-F01	F1_NKK160-F01
experimental_screens_esm2_t33_650M_UR50D_4_epochs_fine_tuned_0.1_masking_7	Verified	G1_NKK153-G01	G1_NKK160-G01
experimental_screens_esm2_t33_650M_UR50D_4_epochs_fine_tuned_0.1_masking_8	Verified	H1_NKK153-H01	H1_NKK160-H01
ESM-IF_experimental_screen_1	N/A (no colonies)	A2_NKK153-A02	A2_NKK160-A02
ESM-IF_experimental_screen_2	Verified	B2_NKK153-B02	B2_NKK160-B02
ESM-IF_experimental_screen_3	Verified	C2_NKK153-C02	C2_NKK160-C02
ESM-IF_experimental_screen_4	Incorrect	D2_NKK153-D02	D2_NKK160-D02
ESM-IF_experimental_screen_5	N/A (no colonies)	E2_NKK153-E02	E2_NKK160-E02
ESM-IF_experimental_screen_6	Verified	F2_NKK153-F02	F2_NKK160-F02
ESM-IF_experimental_screen_7	Inconclusive	G2_NKK153-G02	G2_NKK160-G02
ESM-IF_experimental_screen_8	N/A (no colonies)	H2_NKK153-H02	H2_NKK160-H02
ESM-IF_experimental_screen_9	Verified	A3_NKK153-A03	A3_NKK160-A03
ESM-IF_experimental_screen_10	Single residue mutation (Gln2 $\rightarrow$ Ser)	B3_NKK153-B03	B3_NKK160-B03
ESM-IF_experimental_screen_11	Verified	C3_NKK153-C03	C3_NKK160-C03

ESM-IF_experimental_screen_12	Incorrect	D3_NKK153-D03	D3_NKK160-D03
ESM-IF_experimental_screen_13	Incorrect	E3_NKK153-E03	E3_NKK160-E03
ESM-IF_experimental_screen_14	Inconclusive	F3_NKK153-F03	F3_NKK160-F03
ESM-IF_experimental_screen_15	Incorrect	G3_NKK153-G03	G3_NKK160-G03
ESM-IF_experimental_screen_16	Verified	H3_NKK153-H03	H3_NKK160-H03
protein_LM_mutated_experimental_screen_1	Verified	A4_NKK153-A04	A4_NKK160-A04
protein_LM_mutated_experimental_screen_2	Incorrect	B4_NKK153-B04	B4_NKK160-B04
protein_LM_mutated_experimental_screen_3	Incorrect	C4_NKK153-C04	C4_NKK160-C04
protein_LM_mutated_experimental_screen_4	Verified	D4_NKK153-D04	D4_NKK160-D04
protein_LM_mutated_experimental_screen_5	Verified	E4_NKK153-E04	E4_NKK160-E04
protein_LM_mutated_experimental_screen_6	Verified	F4_NKK153-F04	F4_NKK160-F04
ProteinMPNN_experimental_screen_1	Incorrect	G4_NKK153-G04	G4_NKK160-G04
ProteinMPNN_experimental_screen_2	Verified	H4_NKK153-H04	H4_NKK160-H04
ProteinMPNN_experimental_screen_3	N/A (no colonies)	A5_NKK153-A05	A5_NKK160-A05
ProteinMPNN_experimental_screen_4	N/A (no colonies)	B5_NKK153-B05	B5_NKK160-B05
ProteinMPNN_experimental_screen_5	Verified	C5_NKK153-C05	C5_NKK160-C05
ProteinMPNN_experimental_screen_6	Verified	D5_NKK153-D05	D5_NKK160-D05
ProteinMPNN_experimental_screen_7	Verified	E5_NKK153-E05	E5_NKK160-E05
ProteinMPNN_experimental_screen_8	Verified	F5_NKK153-F05	F5_NKK160-F05
ProteinMPNN_experimental_screen_9	Verified	G5_NKK153-G05	G5_NKK160-G05
ProteinMPNN_experimental_screen_10	N/A (no colonies)	H5_NKK153-H05	H5_NKK160-H05
ProteinMPNN_experimental_screen_11	Verified	A6_NKK153-A06	A6_NKK160-A06
ProteinMPNN_experimental_screen_12	N/A (no colonies)	B6_NKK153-B06	B6_NKK160-B06
ProteinMPNN_experimental_screen_13	Verified	C6_NKK153-C06	C6_NKK160-C06
ProteinMPNN_experimental_screen_14	Verified	D6_NKK153-D06	D6_NKK160-D06
ProteinMPNN_experimental_screen_15	N/A (no colonies)	E6_NKK153-E06	E6_NKK160-E06
ProteinMPNN_experimental_screen_16	N/A (no colonies)	F6_NKK153-F06	F6_NKK160-F06

References

1. Sutter,M., Boehringer,D., Gutmann,S., Günther,S., Prangishvili,D., Loessner,M.J., Stetter,K.O., Weber-Ban,E. and Ban,N. (2008) Structural basis of enzyme encapsulation into a bacterial nanocompartment. *Nat Struct Mol Biol*, **15**, 939–947.

2. Koppers,M., Özkan,N. and Farías,G.G. (2020) Complex Interactions Between Membrane-Bound Organelles, Biomolecular Condensates and the Cytoskeleton. *Frontiers in Cell and Developmental Biology*, **8**.

3. Boeynaems,S., Alberti,S., Fawzi,N.L., Mittag,T., Polymenidou,M., Rousseau,F., Schymkowitz,J., Shorter,J., Wolozin,B., Van Den Bosch,L., *et al.* (2018) Protein Phase Separation: A New Phase in Cell Biology. *Trends Cell Biol*, **28**, 420–435.

4. Alberts,B., Johnson,A., Lewis,J., Raff,M., Roberts,K. and Walter,P. (2002) The Compartmentalization of Cells. *Molecular Biology of the Cell. 4th edition*.

5. Schwarz,D.S. and Blower,M.D. (2016) The endoplasmic reticulum: structure, function and response to cellular signaling. *Cell Mol Life Sci*, **73**, 79–94.

6. Allen,J.F. (2015) Why chloroplasts and mitochondria retain their own genomes and genetic systems: Colocation for redox regulation of gene expression. *Proc Natl Acad Sci U S A*, **112**, 10231–10238.

7. Gray,M.W. (2017) Lynn Margulis and the endosymbiont hypothesis: 50 years later. *Mol Biol Cell*, **28**, 1285–1287.

8. Martin,W.F., Garg,S. and Zimorski,V. (2015) Endosymbiotic theories for eukaryote origin. *Philos Trans R Soc Lond B Biol Sci*, **370**, 20140330.

9. Doolittle,W.F. (1998) A paradigm gets shifty. *Nature*, **392**, 15–16.

10. van Teeseling,M.C.F., Neumann,S. and van Niftrik,L. (2013) The anammoxosome organelle is crucial for the energy metabolism of anaerobic ammonium oxidizing bacteria. *J Mol Microbiol Biotechnol*, **23**, 104–117.

11. Huokko,T., Ni,T., Dykes,G.F., Simpson,D.M., Brownridge,P., Conradi,F.D., Beynon,R.J., Nixon,P.J., Mullineaux,C.W., Zhang,P., *et al.* (2021) Probing the biogenesis pathway and dynamics of thylakoid membranes. *Nat Commun*, **12**, 3475.

12. Lower,B.H. and Bazylinski,D.A. (2013) The bacterial magnetosome: a unique prokaryotic organelle. *J Mol Microbiol Biotechnol*, **23**, 63–80.

13. Hurek,T., Van Montagu,M., Kellenberger,E. and Reinhold-Hurek,B. (1995) Induction of complex intracytoplasmic membranes related to nitrogen fixation in Azoarcus sp. BH72. *Mol Microbiol*, **18**, 225–236.

14. Ladenstein,R. and Morgunova,E. (2020) Second career of a biosynthetic enzyme: Lumazine synthase as a virus-like nanoparticle in vaccine development. *Biotechnology Reports*, **27**, e00494.

15. Rivera,M. (2017) Bacterioferritin: Structure, Dynamics, and Protein–Protein Interactions at Play in Iron Storage and Mobilization. *Acc Chem Res*, **50**, 331–340.

16. Tamura,A., Fukutani,Y., Takami,T., Fujii,M., Nakaguchi,Y., Murakami,Y., Noguchi,K., Yohda,M. and Odaka,M. (2015) Packaging guest proteins into the encapsulin nanocompartment from Rhodococcus erythropolis N771. *Biotechnol Bioeng*, **112**, 13–20.

17. Giessen,T.W., Orlando,B.J., Verdegaal,A.A., Chambers,M.G., Gardener,J., Bell,D.C., Birrane,G., Liao,M. and Silver,P.A. (2019) Large protein organelles form a new iron sequestration system with high storage capacity. *eLife*, **8**, e46070.

18. Giessen,T.W. and Silver,P.A. (2017) Widespread distribution of encapsulin nanocompartments reveals functional diversity. *Nat Microbiol*, **2**, 1–11.

19. Nichols,R.J., LaFrance,B., Phillips,N.R., Radford,D.R., Oltrogge,L.M., Valentin-Alvarado,L.E., Bischoff,A.J., Nogales,E. and Savage,D.F. (2021) Discovery and characterization of a novel family of prokaryotic nanocompartments involved in sulfur metabolism. *eLife*, **10**, e59288.

20. Andreas,M.P. and Giessen,T.W. (2021) Large-scale computational discovery and analysis of virus-derived microbial nanocompartments. *Nat Commun*, **12**, 4748.

21. Jones,J.A., Andreas,M.P. and Giessen,T.W. (2023) Structural basis for peroxidase encapsulation in a protein nanocompartment. 10.1101/2023.09.18.558302.

22. Eren,E., Wang,B., Winkler,D.C., Watts,N.R., Steven,A.C. and Wingfield,P.T. (2022) Structural characterization of the Myxococcus xanthus encapsulin and ferritin-like cargo system gives insight into its iron storage mechanism. *Structure*, **0**.

23. Benisch,R., Andreas,M.P. and Giessen,T.W. (2024) A widespread bacterial protein compartment sequesters and stores elemental sulfur. *Science Advances*, **10**.

24. Andreas,M.P. and Giessen,T.W. (2024) The biosynthesis of the odorant 2-methylisoborneol is compartmentalized inside a protein shell. 10.1101/2024.04.23.590730.

25. Bader,C.D., Panter,F. and Müller,R. (2020) In depth natural product discovery - Myxobacterial strains that provided multiple secondary metabolites. *Biotechnol Adv*, **39**, 107480.

26. Andreas,M.P. and Giessen,T.W. (2021) Large-scale computational discovery and analysis of virus-derived microbial nanocompartments. *Nat Commun*, **12**, 4748.

27. McHugh,C.A., Fontana,J., Nemecek,D., Cheng,N., Aksyuk,A.A., Heymann,J.B., Winkler,D.C., Lam,A.S., Wall,J.S., Steven,A.C., *et al.* (2014) A virus capsid-like nanocompartment that stores iron and protects bacteria from oxidative stress. *EMBO J*, **33**, 1896–1911.

28. Louten,J. (2016) Virus Structure and Classification. *Essential Human Virology*, 10.1016/B978-0-12-800947-5.00002-8.

29. Szyszka,T.N., Andreas,M.P., Lie,F., Miller,L.M., Adamson,L.S.R., Fatehi,F., Twarock,R., Draper,B.E., Jarrold,M.F., Giessen,T.W., *et al.* (2024) Point mutation in a virus-like capsid drives symmetry reduction to form tetrahedral cages. 10.1101/2024.02.05.579038.

30. Bhattacharya,S., Jenkins,M.C., Keshavarz-Joud,P., Bourque,A.R., White,K., Barkane,A.M.A., Bryksin,A.V., Hernandez,C., Kopylov,M. and Finn,M.G. (2024) Heterologous Prime-Boost with Immunologically Orthogonal Protein Nanoparticles for Peptide Immunofocusing. 10.1101/2024.02.24.581861.

31. Lee,D.Y., Bartels,C., McNair,K., Edwards,R.A., Swairjo,M.A. and Luque,A. (2022) Predicting the capsid architecture of phages from metagenomic data. *Computational and Structural Biotechnology Journal*, **20**, 721–732.

32. LaFrance,B.J., Cassidy-Amstutz,C., Nichols,R.J., Oltrogge,L.M., Nogales,E. and Savage,D.F. (2021) The encapsulin from Thermotoga maritima is a flavoprotein with a symmetry matched ferritin-like cargo protein. *Sci Rep*, **11**, 22810.

33. Tang,Y., Mu,A., Zhang,Y., Zhou,S., Wang,W., Lai,Y., Zhou,X., Liu,F., Yang,X., Gong,H., *et al.* (2021) Cryo-EM structure of Mycobacterium smegmatis DyP-loaded encapsulin. *PNAS*, **118**.

34. Lončar,N., Rozeboom,H.J., Franken,L.E., Stuart,M.C.A. and Fraaije,M.W. (2020) Structure of a robust bacterial protein cage and its application as a versatile biocatalytic platform through enzyme encapsulation. *Biochemical and Biophysical Research Communications*, **529**, 548–553.

35. Ross,J., McIver,Z., Lambert,T., Piergentili,C., Bird,J.E., Gallagher,K.J., Cruickshank,F.L., James,P., Zarazúa-Arvizu,E., Horsfall,L.E., *et al.* (2022) Pore dynamics and asymmetric cargo loading in an encapsulin nanocompartment. *Science Advances*, **8**, eabj4461.

36. C. Allende-Ballestero, D. Luque, R. Klem, J.J.L.M. Cornelissen, and J.R. Caston (2022) Brevibacterium linens encapsulin structure (PDB ID 7BCV).

37. Jones,J.A., Andreas,M.P. and Giessen,T.W. (2023) Exploring the Extreme Acid Tolerance of a Dynamic Protein Nanocage. *Biomacromolecules*, **24**, 1388–1399.

38. Akita,F., Chong,K.T., Tanaka,H., Yamashita,E., Miyazaki,N., Nakaishi,Y., Suzuki,M., Namba,K., Ono,Y., Tsukihara,T., *et al.* (2007) The Crystal Structure of a Virus-like Particle from the Hyperthermophilic Archaeon Pyrococcus furiosus Provides Insight into the Evolution of Viruses. *Journal of Molecular Biology*, **368**, 1469–1483.

39. Clancy Kelley,L.-L., Dillard,B.D., Tempel,W., Chen,L., Shaw,N., Lee,D., Newton,M.G., Sugar,F.J., Jenney,F.E., Lee,H.S., *et al.* (2007) Structure of the hypothetical protein

PF0899 from *Pyrococcus furiosus* at 1.85 Å resolution. *Acta Crystallogr F Struct Biol Cryst Commun*, **63**, 549–552.

40. Suhanovsky,M.M. and Teschke,C.M. (2015) Nature's favorite building block: Deciphering folding and capsid assembly of proteins with the HK97-fold. *Virology*, **479–480**, 487–497.

41. Kanhere,A. and Vingron,M. (2009) Horizontal Gene Transfers in prokaryotes show differential preferences for metabolic and translational genes. *BMC Evol Biol*, **9**, 9.

42. Krupovic,M. and Koonin,E.V. (2017) Multiple origins of viral capsid proteins from cellular ancestors. *Proc Natl Acad Sci U S A*, **114**, E2401–E2410.

43. Hua,J., Huet,A., Lopez,C.A., Toropova,K., Pope,W.H., Duda,R.L., Hendrix,R.W. and Conway,J.F. (2017) Capsids and Genomes of Jumbo-Sized Bacteriophages Reveal the Evolutionary Reach of the HK97 Fold. *mBio*, **8**, e01579-17.

44. Duda,R.L., Oh,B. and Hendrix,R.W. (2013) Functional Domains of the HK97 Capsid Maturation Protease and the Mechanisms of Protein Encapsidation. *Journal of Molecular Biology*, **425**, 2765–2781.

45. Hendrix,R.W. and Johnson,J.E. (2012) Bacteriophage HK97 capsid assembly and maturation. *Adv Exp Med Biol*, **726**, 351–363.

46. Forterre,P. (2006) The origin of viruses and their possible roles in major evolutionary transitions. *Virus Res*, **117**, 5–16.

47. Engineered protein nanocages for concurrent RNA and protein packaging in vivo | bioRxiv.

48. Lagoutte,P., Mignon,C., Stadthagen,G., Potisopon,S., Donnat,S., Mast,J., Lugari,A. and Werle,B. (2018) Simultaneous surface display and cargo loading of encapsulin nanocompartments and their use for rational vaccine design. *Vaccine*, **36**, 3622–3628.

49. Khaleeq,S., Sengupta,N., Kumar,S., Patel,U.R., Rajmani,R.S., Reddy,P., Pandey,S., Singh,R., Dutta,S., Ringe,R.P., *et al.* (2023) Neutralizing Efficacy of Encapsulin Nanoparticles against SARS-CoV2 Variants of Concern. *Viruses*, **15**, 346.

50. Tetter,S., Terasaka,N., Steinauer,A., Bingham,R.J., Clark,S., Scott,A.J.P., Patel,N., Leibundgut,M., Wroblewski,E., Ban,N., *et al.* (2021) Evolution of a virus-like architecture and packaging mechanism in a repurposed bacterial protein. *Science*, **372**, 1220–1224.

51. Van de Steen,A., Wilkinson,H.C., Dalby,P.A. and Frank,S. (2024) Encapsulation of Transketolase into In Vitro-Assembled Protein Nanocompartments Improves Thermal Stability. *ACS Appl. Bio Mater.*, **7**, 3660–3674.

52. Van de Steen,A., Khalife,R., Colant,N., Mustafa Khan,H., Deveikis,M., Charalambous,S., Robinson,C.M., Dabas,R., Esteban Serna,S., Catana,D.A., *et al.* (2021) Bioengineering

bacterial encapsulin nanocompartments as targeted drug delivery system. *Synthetic and Systems Biotechnology*, **6**, 231–241.

53. Sigmund,F., Berezin,O., Beliakova,S., Magerl,B., Drawitsch,M., Piovesan,A., Gonçalves,F., Bodea,S.-V., Winkler,S., Bousraou,Z., *et al.* (2023) Genetically encoded barcodes for correlative volume electron microscopy. *Nat Biotechnol*, 10.1038/s41587-023-01713-y.

54. Medema,M.H., Kottmann,R., Yilmaz,P., Cummings,M., Biggins,J.B., Blin,K., de Bruijn,I., Chooi,Y.H., Claesen,J., Coates,R.C., *et al.* (2015) Minimum Information about a Biosynthetic Gene cluster. *Nat Chem Biol*, **11**, 625–631.

55. Craney,A., Ahmed,S. and Nodwell,J. (2013) Towards a new science of secondary metabolism. *J Antibiot*, **66**, 387–400.

56. Donia,M.S., Cimermancic,P., Schulze,C.J., Wieland Brown,L.C., Martin,J., Mitreva,M., Clardy,J., Linington,R.G. and Fischbach,M.A. (2014) A systematic analysis of biosynthetic gene clusters in the human microbiome reveals a common family of antibiotics. *Cell*, **158**, 1402–1414.

57. Fouillaud,M. and Dufossé,L. (2022) Microbial Secondary Metabolism and Biotechnology. *Microorganisms*, **10**, 123.

58. Kallscheuer,N., Classen,T., Drepper,T. and Marienhagen,J. (2019) Production of plant metabolites with applications in the food industry using engineered microorganisms. *Curr Opin Biotechnol*, **56**, 7–17.

59. Faccio,G. (2020) Plant Complexity and Cosmetic Innovation. *iScience*, **23**, 101358.

60. FLEMING,A. (1944) THE DISCOVERY OF PENICILLIN. *British Medical Bulletin*, **2**, 4–5.

61. Bo,G. (2000) Giuseppe Brotzu and the discovery of cephalosporins. *Clinical Microbiology and Infection*, **6**, 6–8.

62. Furey,A. (2010) Assessing bioactivity. *Pharmacognosy Res*, **2**, 203–204.

63. Wani,M.C. and Horwitz,S.B. (2014) Nature as a Remarkable Chemist: A Personal Story of the Discovery and Development of Taxol®. *Anticancer Drugs*, **25**, 482–487.

64. Hofmann,A. (1980) LSD, my problem child McGraw-Hill, New York.

65. Davies,J. (2006) Where have All the Antibiotics Gone? *Can J Infect Dis Med Microbiol*, **17**, 287–290.

66. Dias,D.A., Urban,S. and Roessner,U. (2012) A Historical Overview of Natural Products in Drug Discovery. *Metabolites*, **2**, 303–336.

67. Ziemert,N., Alanjary,M. and Weber,T. (2016) The evolution of genome mining in microbes – a review. *Natural Product Reports*, **33**, 988–1005.

68. Nasko,D.J., Koren,S., Phillippy,A.M. and Treangen,T.J. (2018) RefSeq database growth influences the accuracy of k-mer-based lowest common ancestor species identification. *Genome Biology*, **19**, 165.

69. Routley,N. (2017) Visualizing the Trillion-Fold Increase in Computing Power. *Visual Capitalist*.

70. Blin,K., Shaw,S., Kloosterman,A.M., Charlop-Powers,Z., van Wezel,G.P., Medema,M.H. and Weber,T. (2021) antiSMASH 6.0: improving cluster detection and comparison capabilities. *Nucleic Acids Res*, **49**, W29–W35.

71. Hannigan,G.D., Prihoda,D., Palicka,A., Soukup,J., Klempir,O., Rampula,L., Durcak,J., Wurst,M., Kotowski,J., Chang,D., *et al.* (2019) A deep learning genome-mining strategy for biosynthetic gene cluster prediction. *Nucleic Acids Research*, **47**, e110.

72. Hughes,R.A. and Ellington,A.D. (2017) Synthetic DNA Synthesis and Assembly: Putting the Synthetic in Synthetic Biology. *Cold Spring Harb Perspect Biol*, **9**, a023812.

73. López-Pérez,P.M., Grimsey,E., Bourne,L., Mikut,R. and Hilpert,K. (2017) Screening and Optimizing Antimicrobial Peptides by Using SPOT-Synthesis. *Frontiers in Chemistry*, **5**.

74. Gregorio,N.E., Levine,M.Z. and Oza,J.P. (2019) A User's Guide to Cell-Free Protein Synthesis. *Methods Protoc*, **2**, 24.

75. Weinshilboum,R.M., Otterness,D.M., Aksoy,I.A., Wood,T.C., Her,C. and Raftogianis,R.B. (1997) Sulfation and sulfotransferases 1: Sulfotransferase molecular biology: cDNAs and genes. *FASEB J*, **11**, 3–14.

76. Hasebe,F., Matsuda,K., Shiraishi,T., Futamura,Y., Nakano,T., Tomita,T., Ishigami,K., Taka,H., Mineki,R., Fujimura,T., *et al.* (2016) Amino-group carrier-protein-mediated secondary metabolite biosynthesis in Streptomyces. *Nat Chem Biol*, **12**, 967–972.

77. Nivina,A., Yuet,K.P., Hsu,J. and Khosla,C. (2019) Evolution and Diversity of Assembly-Line Polyketide Synthases. *Chem. Rev.*, **119**, 12524–12547.

78. Gulick,A.M. (2017) Nonribosomal Peptide Synthetase Biosynthetic Clusters of ESKAPE Pathogens. *Nat Prod Rep*, **34**, 981–1009.

79. Gorges,J., Panter,F., Kjaerulff,L., Hoffmann,T., Kazmaier,U. and Müller,R. (2018) Structure, Total Synthesis, and Biosynthesis of Chloromyxamides: Myxobacterial Tetrapeptides Featuring an Uncommon 6-Chloromethyl-5-methoxypipecolic Acid Building Block. *Angew Chem Int Ed Engl*, **57**, 14270–14275.

80. Sutter,M., Melnicki,M.R., Schulz,F., Woyke,T. and Kerfeld,C.A. (2021) A catalog of the diversity and ubiquity of bacterial microcompartments. *Nat Commun*, **12**, 3809.

81. Bishop,C.M. (2006) Pattern recognition and machine learning Springer, New York.

82. Vollmar,M. and Evans,G. (2021) Machine learning applications in macromolecular X-ray crystallography. *Crystallography Reviews*, **27**, 54–101.

83. Chollet,F. (2021) Deep learning with Python Second edition. Manning Publications, Shelter Island.

84. TURING,A.M. (1950) I.—COMPUTING MACHINERY AND INTELLIGENCE. *Mind*, **LIX**, 433–460.

85. Samuel,A.L. (2000) Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, **44**, 206–226.

86. Brain,I.T. and Rosenblatt,F. The Perceptron: A Probabilistic Model for Information Storage and Organization.

87. Chang,P. (2022) Multi-Layer Perceptron Neural Network for Improving Detection Performance of Malicious Phishing URLs Without Affecting Other Attack Types Classification. 10.48550/arXiv.2203.00774.

88. Mehlig,B. (2021) Machine learning with neural networks.

89. Rumelhart,D.E., Hinton,G.E. and Williams,R.J. (1986) Learning representations by back-propagating errors. *Nature*, **323**, 533–536.

90. Toosi,A., Bottino,A.G., Saboury,B., Siegel,E. and Rahmim,A. (2021) A Brief History of AI: How to Prevent Another Winter (A Critical Review). *PET Clinics*, **16**, 449–469.

91. Minsky,M. and Papert,S.A. (1972) Perceptrons: an introduction to computational geometry 2. print. with corr. The MIT Press, Cambridge/Mass.

92. Fukushima,K. (1980) Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biol. Cybernetics*, **36**, 193–202.

93. He,K., Zhang,X., Ren,S. and Sun,J. (2015) Deep Residual Learning for Image Recognition.

94. Zhang,S., Yao,L., Sun,A. and Tay,Y. (2020) Deep Learning based Recommender System: A Survey and New Perspectives. *ACM Comput. Surv.*, **52**, 1–38.

95. Vaswani,A., Shazeer,N., Parmar,N., Uszkoreit,J., Jones,L., Gomez,A.N., Kaiser,L. and Polosukhin,I. (2017) Attention Is All You Need. *arXiv:1706.03762 [cs].*

96. Dosovitskiy,A., Beyer,L., Kolesnikov,A., Weissenborn,D., Zhai,X., Unterthiner,T., Dehghani,M., Minderer,M., Heigold,G., Gelly,S., *et al.* (2021) An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. 10.48550/arXiv.2010.11929.

97. Brown,T.B., Mann,B., Ryder,N., Subbiah,M., Kaplan,J., Dhariwal,P., Neelakantan,A., Shyam,P., Sastry,G., Askell,A., *et al.* (2020) Language Models are Few-Shot Learners. 10.48550/arXiv.2005.14165.

98. Karras,T., Aila,T., Laine,S. and Lehtinen,J. (2018) Progressive Growing of GANs for Improved Quality, Stability, and Variation.

99. A robot wrote this entire article. Are you scared yet, human? (2020) *The Guardian*.

100. Qin,C., Zhang,A., Zhang,Z., Chen,J., Yasunaga,M. and Yang,D. (2023) Is ChatGPT a General-Purpose Natural Language Processing Task Solver?

101. Yang,L., Zhang,Z., Song,Y., Hong,S., Xu,R., Zhao,Y., Shao,Y., Zhang,W., Cui,B. and Yang,M.-H. (2022) Diffusion Models: A Comprehensive Survey of Methods and Applications.

102. Rombach,R., Blattmann,A., Lorenz,D., Esser,P. and Ommer,B. (2022) High-Resolution Image Synthesis with Latent Diffusion Models. 10.48550/arXiv.2112.10752.

103. Ramesh,A., Dhariwal,P., Nichol,A., Chu,C. and Chen,M. (2022) Hierarchical Text-Conditional Image Generation with CLIP Latents.

104. Vincent,J. (2023) AI art tools Stable Diffusion and Midjourney targeted with copyright lawsuit. *The Verge*.

105. Ignatov,A., Chiang,C.-M., Kuo,H.-K., Sycheva,A., Timofte,R., Chen,M.-H., Lee,M.-Y., Xu,Y.-S., Tseng,Y., Xu,S., *et al.* (2021) Learned Smartphone ISP on Mobile NPUs with Deep Learning, Mobile AI 2021 Challenge: Report.

106. Hey Siri: An On-device DNN-powered Voice Trigger for Apple's Personal Assistant *Apple Machine Learning Research*.

107. Zhao,Z.-Q., Zheng,P., Xu,S. and Wu,X. (2019) Object Detection with Deep Learning: A Review.

108. Dada,E.G., Bassi,J.S., Chiroma,H., Abdulhamid,S.M., Adetunmbi,A.O. and Ajibuwa,O.E. (2019) Machine learning for email spam filtering: review, approaches and open research problems. *Heliyon*, **5**, e01802.

109. Understanding searches better than ever before (2019) *Google*.

110. Zhang,S., Roller,S., Goyal,N., Artetxe,M., Chen,M., Chen,S., Dewan,C., Diab,M., Li,X., Lin,X.V., *et al.* (2022) OPT: Open Pre-trained Transformer Language Models.

111. Patterson,D., Gonzalez,J., Le,Q., Liang,C., Munguia,L.-M., Rothchild,D., So,D., Texier,M. and Dean,J. Carbon Emissions and Large Neural Network Training.

112. Strubell,E., Ganesh,A. and McCallum,A. (2019) Energy and Policy Considerations for Deep Learning in NLP.

113. Dickson,B. (2022) Can large language models be democratized? - TechTalks.

114. Rudin,C. and Radin,J. (2019) Why Are We Using Black Box Models in AI When We Don't Need To? A Lesson From an Explainable AI Competition. *Harvard Data Science Review*, **1**.

115. Zablocki,É., Ben-Younes,H., Pérez,P. and Cord,M. (2022) Explainability of deep vision-based autonomous driving systems: Review and challenges.

116. Sathyan,A., Weinberg,A.I. and Cohen,K. (2022) Interpretable AI for bio-medical applications. *Complex Eng Syst*, **2**, 18.

117. Han,H. and Liu,X. (2022) The challenges of explainable AI in biomedical data science. *BMC Bioinformatics*, **22**, 443.

118. Lötsch,J., Kringel,D. and Ultsch,A. (2022) Explainable Artificial Intelligence (XAI) in Biomedicine: Making AI Decisions Trustworthy for Physicians and Patients. *BioMedInformatics*, **2**, 1–17.

119. Li,X., Xiong,H., Li,X., Wu,X., Zhang,X., Liu,J., Bian,J. and Dou,D. (2021) Interpretable Deep Learning: Interpretation, Interpretability, Trustworthiness, and Beyond. 10.48550/arXiv.2103.10689.

120. Ahn,J. and Oh,A. (2021) Mitigating Language-Dependent Ethnic Bias in BERT.

121. Liu,R., Jia,C., Wei,J., Xu,G., Wang,L. and Vosoughi,S. (2021) Mitigating Political Bias in Language Models Through Reinforced Calibration.

122. Srivastava,A., Rastogi,A., Rao,A., Shoeb,A.A.M., Abid,A., Fisch,A., Brown,A.R., Santoro,A., Gupta,A., Garriga-Alonso,A., *et al.* (2022) Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models.

123. Lin,T.-Y., Maire,M., Belongie,S., Bourdev,L., Girshick,R., Hays,J., Perona,P., Ramanan,D., Zitnick,C.L. and Dollár,P. (2015) Microsoft COCO: Common Objects in Context.

124. Measuring Goodhart's law.

125. Marcus,G. Deep Learning: A Critical Appraisal.

126. LeCun,Y. A Path Towards Autonomous Machine Intelligence Version 0.9.2, 2022-06-27.

127. Jones,D.T., Taylor,W.R. and Thornton,J.M. (1992) The rapid generation of mutation data matrices from protein sequences. *Comput Appl Biosci*, **8**, 275–282.

128. Loewenstein,Y., Raimondo,D., Redfern,O.C., Watson,J., Frishman,D., Linial,M., Orengo,C., Thornton,J. and Tramontano,A. (2009) Protein function annotation by homology-based inference. *Genome Biology*, **10**, 207.

129. Eswar,N., Webb,B., Marti-Renom,M.A., Madhusudhan,M.S., Eramian,D., Shen,M., Pieper,U. and Sali,A. (2006) Comparative Protein Structure Modeling Using Modeller. *Curr Protoc Bioinformatics*, **0 5**, Unit-5.6.

130. Sievers,F., Wilm,A., Dineen,D., Gibson,T.J., Karplus,K., Li,W., Lopez,R., McWilliam,H., Remmert,M., Söding,J., *et al.* (2011) Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. *Mol Syst Biol*, **7**, 539.

131. Finn,R.D., Clements,J. and Eddy,S.R. (2011) HMMER web server: interactive sequence similarity searching. *Nucleic Acids Research*, **39**, W29–W37.

132. The UniProt Consortium (2021) UniProt: the universal protein knowledgebase in 2021. *Nucleic Acids Research*, **49**, D480–D489.

133. Berman,H.M., Westbrook,J., Feng,Z., Gilliland,G., Bhat,T.N., Weissig,H., Shindyalov,I.N. and Bourne,P.E. (2000) The Protein Data Bank. *Nucleic Acids Research*, **28**, 235–242.

134. Jumper,J., Evans,R., Pritzel,A., Green,T., Figurnov,M., Ronneberger,O., Tunyasuvunakool,K., Bates,R., Z'idek,A., Potapenko,A., *et al.* (2021) Highly accurate protein structure prediction with AlphaFold. *Nature*, 10.1038/s41586-021-03819-2.

135. Abramson,J., Adler,J., Dunger,J., Evans,R., Green,T., Pritzel,A., Ronneberger,O., Willmore,L., Ballard,A.J., Bambrick,J., *et al.* (2024) Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature*, **630**, 493–500.

136. Rives,A., Meier,J., Sercu,T., Goyal,S., Lin,Z., Liu,J., Guo,D., Ott,M., Zitnick,C.L., Ma,J., *et al.* (2021) Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proc Natl Acad Sci U S A*, **118**, e2016239118.

137. Lin,Z., Akin,H., Rao,R., Hie,B., Zhu,Z., Lu,W., Smetanin,N., Verkuil,R., Kabeli,O., Shmueli,Y., *et al.* (2023) Evolutionary-scale prediction of atomic level protein structure with a language model. *Science*, **379**, 1123–1130.

138. Bileschi,M.L., Belanger,D., Bryant,D., Sanderson,T., Carter,B., Sculley,D., DePristo,M.A. and Colwell,L.J. (2019) Using Deep Learning to Annotate the Protein Universe. 10.1101/626507.

139. Elnaggar,A., Heinzinger,M., Dallago,C., Rehawi,G., Wang,Y., Jones,L., Gibbs,T., Feher,T., Angerer,C., Steinegger,M., *et al.* (2021) ProtTrans: Towards Cracking the Language of Lifes Code Through Self-Supervised Deep Learning and High Performance Computing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10.1109/TPAMI.2021.3095381.

140. Rao,R., Meier,J., Sercu,T., Ovchinnikov,S. and Rives,A. (2020) Transformer protein language models are unsupervised structure learners. 10.1101/2020.12.15.422761.

141. Rao,R., Liu,J., Verkuil,R., Meier,J., Canny,J.F., Abbeel,P., Sercu,T. and Rives,A. (2021) MSA Transformer. 10.1101/2021.02.12.430858.

142. Meier,J., Rao,R., Verkuil,R., Liu,J., Sercu,T. and Rives,A. (2021) Language models enable zero-shot prediction of the effects of mutations on protein function. 10.1101/2021.07.09.450648.

143. Hayes,T., Rao,R., Akin,H., Sofroniew,N.J., Oktay,D., Lin,Z., Verkuil,R., Tran,V.Q., Deaton,J., Wiggert,M., *et al.* (2024) Simulating 500 million years of evolution with a language model. 10.1101/2024.07.01.600583.

144. Varadi,M., Anyango,S., Deshpande,M., Nair,S., Natassia,C., Yordanova,G., Yuan,D., Stroe,O., Wood,G., Laydon,A., *et al.* (2022) AlphaFold Protein Structure Database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic Acids Research*, **50**, D439–D444.

145. Bank,R.P.D. PDB Reaches a New Milestone: 200,000+ Entries.

146. Richardson,J.S. and Richardson,D.C. (1989) The de novo design of protein structures. *Trends in Biochemical Sciences*, **14**, 304–309.

147. Privett,H.K., Kiss,G., Lee,T.M., Blomberg,R., Chica,R.A., Thomas,L.M., Hilvert,D., Houk,K.N. and Mayo,S.L. (2012) Iterative approach to computational enzyme design. *Proceedings of the National Academy of Sciences*, **109**, 3790–3795.

148. Norman,R.A., Ambrosetti,F., Bonvin,A.M.J.J., Colwell,L.J., Kelm,S., Kumar,S. and Krawczyk,K. (2020) Computational approaches to therapeutic antibody design: established methods and emerging trends. *Briefings in Bioinformatics*, **21**, 1549–1567.

149. Ben-Sasson,A.J., Watson,J.L., Sheffler,W., Johnson,M.C., Bittleston,A., Somasundaram,L., Decarreau,J., Jiao,F., Chen,J., Mela,I., *et al.* (2021) Design of biologically active binary protein 2D materials. *Nature*, **589**, 468–473.

150. Kendrew,J.C. (1961) The three-dimensional structure of a protein molecule. *Sci Am*, **205**, 96–110.

151. Balakrishnan,S., Kamisetty,H., Carbonell,J.G., Lee,S.-I. and Langmead,C.J. (2011) Learning generative models for protein fold families. *Proteins: Structure, Function, and Bioinformatics*, **79**, 1061–1078.

152. Rost,B. (1999) Twilight zone of protein sequence alignments. *Protein Eng*, **12**, 85–94.

153. Savva,R. (2019) Targeting uracil-DNA glycosylases for therapeutic outcomes using insights from virus evolution. *Future Medicinal Chemistry*, **11**, 1323–1344.

154. Redler,R.L., Das,J., Diaz,J.R. and Dokholyan,N.V. (2016) Protein Destabilization as a Common Factor in Diverse Inherited Disorders. *J Mol Evol*, **82**, 11–16.

155. Leman,J.K., Weitzner,B.D., Lewis,S.M., Adolf-Bryfogle,J., Alam,N., Alford,R.F., Aprahamian,M., Baker,D., Barlow,K.A., Barth,P., *et al.* (2020) Macromolecular modeling and design in Rosetta: recent methods and frameworks. *Nat Methods*, **17**, 665–680.

156. Bale,J.B., Gonen,S., Liu,Y., Sheffler,W., Ellis,D., Thomas,C., Cascio,D., Yeates,T.O., Gonen,T., King,N.P., *et al.* (2016) Accurate design of megadalton-scale two-component icosahedral protein complexes. *Science*, **353**, 389–394.

157. Weitzner,B.D., Jeliazkov,J.R., Lyskov,S., Marze,N., Kuroda,D., Frick,R., Adolf-Bryfogle,J., Biswas,N., Dunbrack,R.L. and Gray,J.J. (2017) Modeling and docking of antibody structures with Rosetta. *Nat Protoc*, **12**, 401–416.

158. Alley,E.C., Khimulya,G., Biswas,S., AlQuraishi,M. and Church,G.M. (2019) Unified rational protein engineering with sequence-based deep representation learning. *Nat Methods*, **16**, 1315–1322.

159. Zhang,Y., Chen,Y., Wang,C., Lo,C.-C., Liu,X., Wu,W. and Zhang,J. (2020) ProDCoNN: Protein design using a convolutional neural network. *Proteins: Structure, Function, and Bioinformatics*, **88**, 819–829.

160. Eguchi,R.R., Choe,C.A. and Huang,P.-S. (2022) Ig-VAE: Generative Modeling of Protein Structure by Direct 3D Coordinate Generation. 10.1101/2020.08.07.242347.

161. Repecka,D., Jauniskis,V., Karpus,L., Rembeza,E., Rokaitis,I., Zrimec,J., Poviloniene,S., Laurynenas,A., Viknander,S., Abuajwa,W., *et al.* (2021) Expanding functional protein sequence spaces using generative adversarial networks. *Nat Mach Intell*, **3**, 324–333.

162. Dauparas,J., Anishchenko,I., Bennett,N., Bai,H., Ragotte,R.J., Milles,L.F., Wicky,B.I.M., Courbet,A., de Haas,R.J., Bethel,N., *et al.* (2022) Robust deep learning–based protein sequence design using ProteinMPNN. *Science*, **378**, 49–56.

163. Hsu,C., Verkuil,R., Liu,J., Lin,Z., Hie,B., Sercu,T., Lerer,A. and Rives,A. (2022) Learning inverse folding from millions of predicted structures. In *Proceedings of Machine Learning Research*.Vol. PMLR 162, pp. 8946–8970.

164. Madani,A., Krause,B., Greene,E.R., Subramanian,S., Mohr,B.P., Holton,J.M., Olmos,J.L., Xiong,C., Sun,Z.Z., Socher,R., *et al.* (2023) Large language models generate functional protein sequences across diverse families. *Nat Biotechnol*, 10.1038/s41587-022-01618-2.

165. Watson,J.L., Juergens,D., Bennett,N.R., Trippe,B.L., Yim,J., Eisenach,H.E., Ahern,W., Borst,A.J., Ragotte,R.J., Milles,L.F., *et al.* (2022) Broadly applicable and accurate protein design by integrating structure prediction networks and diffusion generative models. 10.1101/2022.12.09.519842.

166. Microsoft foldingdiff - GitHub Repository.

167. Trippe,B.L., Yim,J., Tischer,D., Baker,D., Broderick,T., Barzilay,R. and Jaakkola,T. (2023) Diffusion probabilistic modeling of protein backbones in 3D for the motif-scaffolding problem. 10.48550/arXiv.2206.04119.

168. Yim,J., Trippe,B.L., De Bortoli,V., Mathieu,E., Doucet,A., Barzilay,R. and Jaakkola,T. (2023) SE(3) diffusion model with application to protein backbone generation.

169. Qi,Y. and Zhang,J.Z.H. (2020) DenseCPD: Improving the Accuracy of Neural-Network-Based Computational Protein Sequence Design with DenseNet. *J. Chem. Inf. Model.*, **60**, 1245–1252.

170. Shin,J.-E., Riesselman,A.J., Kollasch,A.W., McMahon,C., Simon,E., Sander,C., Manglik,A., Kruse,A.C. and Marks,D.S. (2021) Protein design and variant prediction using autoregressive generative models. *Nat Commun*, **12**, 2403.

171. Verkuil,R., Kabeli,O., Du,Y., Wicky,B.I.M., Milles,L.F., Dauparas,J., Baker,D., Ovchinnikov,S., Sercu,T. and Rives,A. (2022) Language models generalize beyond natural proteins. 10.1101/2022.12.21.521521.

172. Ingraham,J.B., Baranov,M., Costello,Z., Barber,K.W., Wang,W., Ismail,A., Frappier,V., Lord,D.M., Ng-Thow-Hing,C., Van Vlack,E.R., *et al.* (2023) Illuminating protein space with a programmable generative model. *Nature*, 10.1038/s41586-023-06728-8.

173. Zhang,L., Chen,F., Zeng,Z., Xu,M., Sun,F., Yang,L., Bi,X., Lin,Y., Gao,Y., Hao,H., *et al.* (2021) Advances in Metagenomics and Its Application in Environmental Microorganisms. *Frontiers in Microbiology*, **12**.

174. Kellenberger,E. (2001) Exploring the unknown. *EMBO Rep*, **2**, 5–7.

175. Nayfach,S., Roux,S., Seshadri,R., Udwary,D., Varghese,N., Schulz,F., Wu,D., Paez-Espino,D., Chen,I.-M., Huntemann,M., *et al.* (2021) A genomic catalog of Earth's microbiomes. *Nat Biotechnol*, **39**, 499–509.

176. Segata,N., Waldron,L., Ballarini,A., Narasimhan,V., Jousson,O. and Huttenhower,C. (2012) Metagenomic microbial community profiling using unique clade-specific marker genes. *Nat Methods*, **9**, 811–814.

177. Payne,M., Azana,R. and Hoang,L.M.N. (2016) Review of 16S and ITS Direct Sequencing Results for Clinical Specimens Submitted to a Reference Laboratory. *Can J Infect Dis Med Microbiol*, **2016**, 4210129.

178. Pérez-Cobas,A.E., Gomez-Valero,L. and Buchrieser,C. (2020) Metagenomic approaches in microbial ecology: an update on whole-genome and marker gene sequencing analyses. *Microb Genom*, **6**, mgen000409.

179. Breitwieser,F.P., Lu,J. and Salzberg,S.L. (2019) A review of methods and databases for metagenomic classification and assembly. *Briefings in Bioinformatics*, **20**, 1125.

180. Blaser,M.J. (2014) The microbiome revolution. *J Clin Invest*, **124**, 4162–4165.

181. Ahn,J. and Hayes,R.B. (2021) Environmental Influences on the Human Microbiome and Implications for Noncommunicable Disease. *Annu Rev Public Health*, **42**, 277–292.

182. Modi,S.R., Collins,J.J. and Relman,D.A. (2014) Antibiotics and the gut microbiota. *J Clin Invest*, **124**, 4212–4218.

183. Shreiner,A.B., Kao,J.Y. and Young,V.B. (2015) The gut microbiome in health and in disease. *Curr Opin Gastroenterol*, **31**, 69–75.

184. Thomas,T., Gilbert,J. and Meyer,F. (2012) Metagenomics - a guide from sampling to data analysis. *Microbial Informatics and Experimentation*, **2**, 3.

185. Richardson,L., Allen,B., Baldi,G., Beracochea,M., Bileschi,M.L., Burdett,T., Burgin,J., Caballero-Pérez,J., Cochrane,G., Colwell,L.J., *et al.* (2023) MGnify: the microbiome sequence data analysis resource in 2023. *Nucleic Acids Research*, **51**, D753–D759.

186. Lobb,B., Kurtz,D.A., Moreno-Hagelsieb,G. and Doxey,A.C. (2015) Remote homology and the functions of metagenomic dark matter. *Front Genet*, **6**, 234.

187. Godzik,A. (2011) Metagenomics and the protein universe. *Curr Opin Struct Biol*, **21**, 398–403.

188. Nijkamp,E., Ruffolo,J., Weinstein,E.N., Naik,N. and Madani,A. (2022) ProGen2: Exploring the Boundaries of Protein Language Models. 10.48550/arXiv.2206.13517.

189. McPherson,A. and Gavira,J.A. (2013) Introduction to protein crystallization. *Acta Crystallogr F Struct Biol Commun*, **70**, 2–20.

190. Giegé,R. (2013) A historical perspective on protein crystallization from 1840 to the present day. *The FEBS Journal*, **280**, 6456–6497.

191. Blount,Z.D. (2015) The unexhausted potential of E. coli. *eLife*, **4**, e05826.

192. Brondyk,W.H. (2009) Chapter 11 Selecting an Appropriate Method for Expressing a Recombinant Protein. In Burgess,R.R., Deutscher,M.P. (eds), *Methods in Enzymology*, Guide to Protein Purification, 2nd Edition. Academic Press, Vol. 463, pp. 131–147.

193. McKenzie,G.J. and Craig,N.L. (2006) Fast, easy and efficient: site-specific insertion of transgenes into Enterobacterial chromosomes using Tn7 without need for selection of the insertion event. *BMC Microbiol*, **6**, 39.

194. del Solar,G., Giraldo,R., Ruiz-Echevarría,M.J., Espinosa,M. and Díaz-Orejas,R. (1998) Replication and Control of Circular Bacterial Plasmids. *Microbiol Mol Biol Rev*, **62**, 434–464.

195. Jahn,M., Vorpahl,C., Hübschmann,T., Harms,H. and Müller,S. (2016) Copy number variability of expression plasmids determined by cell sorting and Droplet Digital PCR. *Microbial Cell Factories*, **15**, 211.

196. Kallunki,T., Barisic,M., Jäättelä,M. and Liu,B. (2019) How to Choose the Right Inducible Gene Expression System for Mammalian Studies? *Cells*, **8**, 796.

197. Rosano,G.L. and Ceccarelli,E.A. (2014) Recombinant protein expression in Escherichia coli: advances and challenges. *Front Microbiol*, **5**, 172.

198. Zhou,Z., Dang,Y., Zhou,M., Li,L., Yu,C., Fu,J., Chen,S. and Liu,Y. (2016) Codon usage is an important determinant of gene expression levels largely through its effects on transcription. *Proceedings of the National Academy of Sciences*, **113**, E6117–E6125.

199. Burgess-Brown,N.A., Sharma,S., Sobott,F., Loenarz,C., Oppermann,U. and Gileadi,O. (2008) Codon optimization can improve expression of human genes in Escherichia coli: A multi-gene study. *Protein Expression and Purification*, **59**, 94–102.

200. Chen,X., Zaro,J. and Shen,W.-C. (2013) Fusion Protein Linkers: Property, Design and Functionality. *Adv Drug Deliv Rev*, **65**, 1357–1369.

201. Pantazatos,D., Kim,J.S., Klock,H.E., Stevens,R.C., Wilson,I.A., Lesley,S.A. and Woods,V.L. (2004) Rapid refinement of crystallographic protein construct definition employing enhanced hydrogen/deuterium exchange MS. *Proc. Natl. Acad. Sci. U.S.A.*, **101**, 751–756.

202. Mishra,V. (2020) Affinity Tags for Protein Purification. *Curr Protein Pept Sci*, **21**, 821–830.

203. Singh,A., Upadhyay,V., Upadhyay,A.K., Singh,S.M. and Panda,A.K. (2015) Protein recovery from inclusion bodies of Escherichia coli using mild solubilization process. *Microbial Cell Factories*, **14**, 41.

204. Tripathi,N.K. (2016) Production and Purification of Recombinant Proteins from Escherichia coli. *ChemBioEng Reviews*, **3**, 116–133.

205. Rustandi,R.R. (2013) Hydrophobic interaction chromatography to analyze glycoproteins. *Methods Mol Biol*, **988**, 211–219.

206. Hong,P., Koza,S. and Bouvier,E.S.P. (2012) Size-Exclusion Chromatography for the Analysis of Protein Biotherapeutics and their Aggregates. *J Liq Chromatogr Relat Technol*, **35**, 2923–2950.

207. Radford,D.R. (2014) Understanding the Encapsulins:Prediction and Characterization of Phage Capsid-like Nanocompartments in Prokaryotes.

208. Tracey,J.C., Coronado,M., Giessen,T.W., Lau,M.C.Y., Silver,P.A. and Ward,B.B. (2019) The Discovery of Twenty-Eight New Encapsulin Sequences, Including Three in Anammox Bacteria. *Sci Rep*, **9**, 20122.

209. Lutz,I.D., Wang,S., Norn,C., Courbet,A., Borst,A.J., Zhao,Y.T., Dosey,A., Cao,L., Xu,J., Leaf,E.M., *et al.* (2023) Top-down design of protein architectures with reinforcement learning. *Science*, **380**, 266–273.

210. Sheffler,W., Yang,E.C., Dowling,Q., Hsia,Y., Fries,C.N., Stanislaw,J., Langowski,M.D., Brandys,M., Li,Z., Skotheim,R., *et al.* (2023) Fast and versatile sequence-independent protein docking for nanomaterials design using RPXDock. *PLOS Computational Biology*, **19**, e1010680.

211. Sumida,K.H., Núñez-Franco,R., Kalvet,I., Pellock,S.J., Wicky,B.I.M., Milles,L.F., Dauparas,J., Wang,J., Kipnis,Y., Jameson,N., *et al.* (2024) Improving protein expression, stability, and function with ProteinMPNN. *J. Am. Chem. Soc.*, 10.1021/jacs.3c10941.

212. Mistry,J., Chuguransky,S., Williams,L., Qureshi,M., Salazar,G.A., Sonnhammer,E.L.L., Tosatto,S.C.E., Paladin,L., Raj,S., Richardson,L.J., *et al.* (2021) Pfam: The protein families database in 2021. *Nucleic Acids Research*, **49**, D412–D419.

213. Bileschi,M., Belanger,D., Bryant,D., Sanderson,T., Carter,B., Sculley,D., DePristo,M. and Colwell,L. (2019) Deep Learning Classifies the Protein Universe. *Nature Biotechnology*.

214. Kempen,M. van, Kim,S.S., Tumescheit,C., Mirdita,M., Lee,J., Gilchrist,C.L.M., Söding,J. and Steinegger,M. (2023) Fast and accurate protein structure search with Foldseek. 10.1101/2022.02.07.479398.

215. Mirdita,M., Steinegger,M. and S"oding,J. (2019) MMseqs2 desktop and local web server app for fast, interactive sequence searches. *Bioinformatics*, **35**, 2856–2858.

216. The UniProt Consortium (2023) UniProt: the Universal Protein Knowledgebase in 2023. *Nucleic Acids Research*, **51**, D523–D531.

217. Google Colaboratory.

218. Holm,L. (2022) Dali server: structural unification of protein families. *Nucleic Acids Research*, **50**, W210–W215.

219. Virtanen,P., Gommers,R., Oliphant,T.E., Haberland,M., Reddy,T., Cournapeau,D., Burovski,E., Peterson,P., Weckesser,W., Bright,J., *et al.* (2020) SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nat Methods*, **17**, 261–272.

220. Schrödinger, LLC (2015) The PyMOL Molecular Graphics System, Version 1.8.

221. Inc,P.T. (2015) Collaborative data science.

222. Sayers,E.W., Bolton,E.E., Brister,J.R., Canese,K., Chan,J., Comeau,D.C., Connor,R., Funk,K., Kelly,C., Kim,S., *et al.* (2022) Database resources of the national center for biotechnology information. *Nucleic Acids Res*, **50**, D20–D26.

223. Richardson,L. beautifulsoup4: Screen-scraping library.

224. Wu,R., Ding,F., Wang,R., Shen,R., Zhang,X., Luo,S., Su,C., Wu,Z., Xie,Q., Berger,B., *et al.* (2022) High-resolution de novo structure prediction from primary sequence. 10.1101/2022.07.21.500999.

225. Zhang,Y. and Skolnick,J. (2004) Scoring function for automated assessment of protein structure template quality. *Proteins: Structure, Function, and Bioinformatics*, **57**, 702–710.

226. Hie,B., Candido,S., Lin,Z., Kabeli,O., Rao,R., Smetanin,N., Sercu,T. and Rives,A. (2022) A high-level programming language for generative protein design. 10.1101/2022.12.21.521526.

227. Evolutionary Scale Modeling (2022).

228. Paysan-Lafosse,T., Blum,M., Chuguransky,S., Grego,T., Pinto,B.L., Salazar,G.A., Bileschi,M.L., Bork,P., Bridge,A., Colwell,L., *et al.* (2023) InterPro in 2022. *Nucleic Acids Research*, **51**, D418–D427.

229. Wolf,T., Debut,L., Sanh,V., Chaumond,J., Delangue,C., Moi,A., Cistac,P., Ma,C., Jernite,Y., Plu,J., *et al.* (2020) Transformers: State-of-the-Art Natural Language Processing.

230. Loshchilov,I. and Hutter,F. (2019) Decoupled Weight Decay Regularization. 10.48550/arXiv.1711.05101.

231. wandb/wandb (2024).

232. Sgarbossa,D., Lupo,U. and Bitbol,A.-F. (2023) Generative power of a protein language model trained on multiple sequence alignments. *eLife*, **12**, e79854.

233. Ravfogel,S., Goldberg,Y. and Goldberger,J. (2023) Conformal Nucleus Sampling. In Rogers,A., Boyd-Graber,J., Okazaki,N. (eds), *Findings of the Association for Computational Linguistics: ACL 2023*. Association for Computational Linguistics, Toronto, Canada, pp. 27–34.

234. Thumuluri,V., Martiny,H.-M., Almagro Armenteros,J.J., Salomon,J., Nielsen,H. and Johansen,A.R. (2022) NetSolP: predicting protein solubility in Escherichia coli using language models. *Bioinformatics*, **38**, 941–946.

235. Benchling Inc (2023) Benchling [Biology Software].

236. New England Biolabs (2023) Tm Calculator.

237. ExPASy - ProtParam tool.

238. Kashif-Khan,N., Savva,R. and Frank,S. (2024) Mining metagenomics data for novel bacterial nanocompartments. *NAR Genomics and Bioinformatics*, **6**, lqae025.

239. Giessen,T.W. (2022) Encapsulins. *Annu Rev Biochem*, **91**, 353–380.

240. Sagot,B., Gaysinski,M., Mehiri,M., Guigonis,J.-M., Le Rudulier,D. and Alloing,G. (2010) Osmotically induced synthesis of the dipeptide N-acetylglutaminylglutamine amide is mediated by a new pathway conserved among bacteria. *Proceedings of the National Academy of Sciences*, **107**, 12652–12657.

241. Corpuz,J.C., Sanlley,J.O. and Burkart,M.D. (2022) Protein-protein interface analysis of the non-ribosomal peptide synthetase peptidyl carrier protein and enzymatic domains. *Synthetic and Systems Biotechnology*, **7**, 677–688.

242. Jones,J.A. and Giessen,T.W. (2021) Advances in encapsulin nanocompartment biology and engineering. *Biotechnol Bioeng*, **118**, 491–505.

243. Eida,A.A. and Mahmud,T. (2019) The secondary metabolite pactamycin with potential for pharmaceutical applications: biosynthesis and regulation. *Appl Microbiol Biotechnol*, **103**, 4337–4345.

244. Prihoda,D., Maritz,J.M., Klempir,O., Dzamba,D., Woelk,C.H., Hazuda,D.J., Bitton,D.A. and Hannigan,G.D. (2021) The application potential of machine learning and genomics for understanding natural product diversity, chemistry, and therapeutic translatability. *Nat. Prod. Rep.*, **38**, 1100–1108.

245. Boyton,I., Goodchild,S.C., Diaz,D., Elbourne,A., Collins-Praino,L.E. and Care,A. (2022) Characterizing the Dynamic Disassembly/Reassembly Mechanisms of Encapsulin Protein Nanocages. *ACS Omega*, **7**, 823–836.

246. Eren,E., Watts,N.R., Conway,J.F. and Wingfield,P.T. (2024) Myxococcus xanthus encapsulin cargo protein EncD is a flavin-binding protein with ferric reductase activity. *Proceedings of the National Academy of Sciences*, **121**, e2400426121.

247. Stetefeld,J., McKenna,S.A. and Patel,T.R. (2016) Dynamic light scattering: a practical guide and applications in biomedical sciences. *Biophys Rev*, **8**, 409–427.

248. Arora,D. and Kambhampati,S. (2023) Learning and Leveraging Verifiers to Improve Planning Capabilities of Pre-trained Language Models.

249. Xu,J. and Zhang,Y. (2010) How significant is a protein structure similarity with TM-score = 0.5? *Bioinformatics*, **26**, 889–895.

250. Liu,Y. and Kuhlman,B. (2006) RosettaDesign server for protein design. *Nucleic Acids Res*, **34**, W235–W238.

251. Eren,Y., Küçükdemiral,İ.B. and Üstoğlu,İ. (2017) Chapter 2 - Introduction to Optimization. In Erdinç,O. (ed), *Optimization in Renewable Energy Systems*. Butterworth-Heinemann, Boston, pp. 27–74.

252. Raffel,C., Shazeer,N., Roberts,A., Lee,K., Narang,S., Matena,M., Zhou,Y., Li,W. and Liu,P.J. (2023) Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer.

253. Heinzinger,M., Weissenow,K., Sanchez,J.G., Henkel,A., Mirdita,M., Steinegger,M. and Rost,B. (2024) Bilingual Language Model for Protein Sequence and Structure. 10.1101/2023.07.23.550085.

254. Devlin,J., Chang,M.-W., Lee,K. and Toutanova,K. (2019) BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 10.48550/arXiv.1810.04805.

255. Wang,A. and Cho,K. (2019) BERT has a Mouth, and It Must Speak: BERT as a Markov Random Field Language Model. 10.48550/arXiv.1902.04094.

256. Johnson,S.R., Monaco,S., Massie,K. and Syed,Z. (2021) Generating novel protein sequences using Gibbs sampling of masked language models. 10.1101/2021.01.26.428322.

257. Elnaggar,A., Essam,H., Salah-Eldin,W., Moustafa,W., Elkerdawy,M., Rochereau,C. and Rost,B. (2023) Ankh: Optimized Protein Language Model Unlocks General-Purpose Modelling.

258. Weber,E., Engler,C., Gruetzner,R., Werner,S. and Marillonnet,S. (2011) A Modular Cloning System for Standardized Assembly of Multigene Constructs. *PLoS One*, **6**.

259. Bird,J.E., Marles-Wright,J. and Giachino,A. (2022) A User's Guide to Golden Gate Cloning Methods and Standards. *ACS Synth. Biol.*, **11**, 3551–3563.

260. Green,M.R. and Sambrook,J. (2019) Screening Bacterial Colonies Using X-Gal and IPTG: α-Complementation. *Cold Spring Harb Protoc*, **2019**.

261. Kasprzyk,M., Herrera,M.A. and Stracquadanio,G. (2024) APEX: Automated Protein EXpression in Escherichia coli. 10.1101/2024.08.13.607171.

262. Kao,H.-W., Lu,W.-L., Ho,M.-R., Lin,Y.-F., Hsieh,Y.-J., Ko,T.-P., Danny Hsu,S.-T. and Wu,K.-P. (2023) Robust Design of Effective Allosteric Activators for Rsp5 E3 Ligase Using the Machine Learning Tool ProteinMPNN. *ACS Synth. Biol.*, **12**, 2310–2319.

263. Chronowska,M., Stam,M.J., Woolfson,D.N., Costanzo,L.F.D. and Wood,C.W. (2024) The Protein Design Archive (PDA): insights from 40 years of protein design. 10.1101/2024.09.05.611465.

264. Ertelt,M., Moretti,R., Meiler,J. and Schoeder,C.T. (2024) Self-supervised machine learning methods for protein design improve sampling, but not the identification of high-fitness variants. 10.1101/2024.06.20.599843.

265. Lin,Y., Lee,M., Zhang,Z. and AlQuraishi,M. (2024) Out of Many, One: Designing and Scaffolding Proteins at the Scale of the Structural Universe with Genie 2. 10.48550/arXiv.2405.15489.

266. Bepler,T. and Berger,B. (2021) Learning the Protein Language: Evolution, Structure and Function. *Cell Syst*, **12**, 654-669.e3.

267. Ertelt,M., Meiler,J. and Schoeder,C.T. (2024) Combining Rosetta Sequence Design with Protein Language Model Predictions Using Evolutionary Scale Modeling (ESM) as Restraint. *ACS Synth Biol*, **13**, 1085–1092.

268. Li,Q., Vlachos,E.N. and Bryant,P. (2024) Design of linear and cyclic peptide binders of different lengths only from a protein target sequence. 10.1101/2024.06.20.599739.

269. Johnson,S.R., Fu,X., Viknander,S., Goldin,C., Monaco,S., Zelezniak,A. and Yang,K.K. (2023) Computational Scoring and Experimental Evaluation of Enzymes Generated by Neural Networks. 10.1101/2023.03.04.531015.

270. Pak,M.A., Markhieva,K.A., Novikova,M.S., Petrov,D.S., Vorobyev,I.S., Maksimova,E.S., Kondrashov,F.A. and Ivankov,D.N. (2023) Using AlphaFold to predict the impact of single mutations on protein stability and function. *PLOS ONE*, **18**, e0282689.

271. Liu,Y. and Liu,H. (2024) Protein sequence design on given backbones with deep learning. *Protein Engineering, Design and Selection*, **37**, gzad024.

272. Xiao,H., Zhou,J., Yang,F., Liu,Z., Song,J., Chen,W., Liu,H. and Cheng,L. (2023) Assembly and Capsid Expansion Mechanism of Bacteriophage P22 Revealed by High-Resolution Cryo-EM Structures. *Viruses*, **15**, 355.

273. Snijder,J., Kononova,O., Barbu,I.M., Uetrecht,C., Rurup,W.F., Burnley,R.J., Koay,M.S.T., Cornelissen,J.J.L.M., Roos,W.H., Barsegov,V., *et al.* (2016) Assembly and Mechanical Properties of the Cargo-Free and Cargo-Loaded Bacterial Nanocompartment Encapsulin. *Biomacromolecules*, **17**, 2522–2529.

274. Hoffmann,J., Borgeaud,S., Mensch,A., Buchatskaya,E., Cai,T., Rutherford,E., Casas,D. de L., Hendricks,L.A., Welbl,J., Clark,A., *et al.* (2022) Training Compute-Optimal Large Language Models.

275. Kaplan,J., McCandlish,S., Henighan,T., Brown,T.B., Chess,B., Child,R., Gray,S., Radford,A., Wu,J. and Amodei,D. (2020) Scaling Laws for Neural Language Models. 10.48550/arXiv.2001.08361.

276. Su,H., Tian,Z., Shen,X. and Cai,X. (2024) Unraveling the Mystery of Scaling Laws: Part I. 10.48550/arXiv.2403.06563.

277. Knight,W. OpenAI's CEO Says the Age of Giant AI Models Is Already Over. *Wired.*

278. Luca,A.R., Ursuleanu,T.F., Gheorghe,L., Grigorovici,R., Iancu,S., Hlusneac,M. and Grigorovici,A. (2022) Impact of quality, type and volume of data used by deep learning models in the analysis of medical images. *Informatics in Medicine Unlocked*, **29**, 100911.

279. Peng,K., Obradovic,Z. and Vucetic,S. (2004) Exploring bias in the Protein Data Bank using contrast classifiers. *Pac Symp Biocomput*, 10.1142/9789812704856_0041.