U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

MSc

2.º
CICLO

FCUP
2021

U.PORTO

A Docker-oriented Pipeline for the analysis of Protein-Protein Interactions

Paula Sofia Vilares Gouveia

FC

U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

A Docker-oriented Pipeline for the analysis of Protein-Protein Interactions

Paula Sofia Vilares Gouveia

Dissertação de Mestrado apresentada à
Faculdade de Ciências da Universidade do Porto em
Bioinformática e Biologia Computacional

2021

A Docker-oriented Pipeline for the analysis of Protein-Protein Interactions

Paula Sofia Vilares Gouveia

Dissertação de Mestrado apresentada à Faculdade de Ciências da Universidade do Porto em Bioinformática e Biologia Computacional

2021

Orientador
Jorge Manuel de Sousa Basto Vieira, Postdoctoral Researcher, Instituto de Investigação e Inovação em Saúde (I3S), Universidade do Porto, Portugal & Instituto de Biologia Molecular e Celular (IBMC)

Orientador
Cristina Alexandra Gonçalves Paula Vieira, Postdoctoral Researcher, Instituto de Investigação e Inovação em Saúde (I3S), Universidade do Porto, Portugal & Instituto de Biologia Molecular e Celular (IBMC)

Orientador
Hugo López-Fernández, Junior Researcher, Instituto de Investigação e Inovação em Saúde (I3S), Universidade do Porto, Portugal & Instituto de Biologia Molecular e Celular (IBMC)

Todas as correções determinadas pelo júri, e só essas, foram efetuadas.

O Presidente do Júri,

Porto,_____/_____/__

Abstract

Protein-protein interactions (PPI) describe the specific contact between two or more proteins within the cellular milieu. With knowledge on the interaction interface, it is possible to understand the underlying biological function of the proteins quaternary structure, ultimately aiding in several research topics, such as the identification of novel drug targets for the treatment of various diseases.

Several *in vivo* and *in vitro* methods for the prediction and analysis of PPI have been developed and are widely used. While they are able to predict PPI with high quality, they are expensive, time-consuming, and the resulting data sets are noisy and prone to false positives. Additionally, performing comparative analyses of PPI properties such as those presented in (Rocha et al. 2019) is laborious and most of the times the software used is not user-friendly, particularly to researchers with no expertise in bioinformatics.

For these reasons, *in silico* approaches can be employed to help with scaling down a wide set of original interactions to a group of more accurate and high-quality interactions, and to perform PPI analyses with ease.

These motivations culminated in the development of P_2I_2P (Protein-Protein Interactions Interface Prediction), a pipeline provided as a Docker image, that infers PPI interfaces. It is divided into mainly five steps producing interface information, such as the number of interfacing residues and solvent accessible area. First, the 3D structures of both interacting proteins are obtained with I-TASSER, then the interacting residues are predicted with CPORT. Next, a model for the docking of the interaction is obtained using HADDOCK and lastly, PDBePISA is used to derive information about the inferred PPI interface.

Experimental PPI data from (Rocha et al. 2019) and protein structures from the recently launched AlphaFold Protein Structure Database (AlphaFold DB) were ran with P_2I_2P to obtain pipeline runtimes.

Having a pipeline that automates the process of inferring PPIs interfaces, unifies all required tools, and provides an alternative to skip some prediction steps given available data, results in a cleaner set of interface data that can be obtained by researchers with or without expertise in the bioinformatics field.

Keywords: protein-protein interactions, Docker, bioinformatics, protein structure, bash scripting, in silico approach

Resumo

Interações proteína-proteína descrevem o contacto específico entre duas ou mais proteínas no meio celular. Tendo conhecimento da sua interface de interação, é possível entender as funções biológicas por detrás da sua estrutura quaternária, contribuindo assim para vários tópicos de investigação, tais como a identificação de novos alvos para fármacos no tratamento de diversas doenças.

Métodos *in vivo* e *in vitro* para a previsão e análise de interações proteína-proteína têm sido desenvolvidos e frequentemente usados. No entanto, embora capazes de prever interações de elevada qualidade, estes métodos são dispendiosos, demorados, e propensos a falsos positivos e ruído nos dados obtidos. Além disso, analisar comparativamente propriedades de interações proteína-proteína tais como as efetuadas em (Rocha et al. 2019) é trabalhoso e na maior parte dos casos o software usado não é intuitivo, especialmente para utilizadores sem conhecimentos de bioinformática.

Por estas razões, abordagens *in silico* podem ser usadas de modo a reduzir um conjunto original de interações para apenas aquelas que são mais passíveis de serem válidas, e de forma a poder analisar com facilidade interações proteína-proteína.

Estes motivos culminaram no desenvolvimento da P_2I_2P (Protein-Protein Interactions Interface Prediction), uma pipeline lançada em formato de imagem Docker, que infere interfaces de interações proteína-proteína. A pipeline encontra-se dividida em cinco passos principais que levam à obtenção de informação sobre a interface, tal como o número de resíduos de interface e área superficial acessível. Primeiramente, as estruturas 3D de ambas as proteínas intervenientes são obtidas usando o I-TASSER, depois os resíduos de interface são calculados com o CPORT. Em seguida, um modelo do docking da interação entre proteínas é inferido através do HADDOCK e, finalmente, o PDBePISA é utilizado para auferir dados sobre a interface proteica.

Dados provenientes de (Rocha et al. 2019) e da recentemente lançada base de dados AlphaFold Protein Structure Database (AlphaFold DB) foram analisados com a P_2I_2P para obter tempos de execução.

Ao existir uma pipeline que automatiza o processo de inferir interfaces de PPI, que une todas as ferramentas necessárias, e fornece a alternativa de passar à frente certos passos dessa previsão tendo em conta os dados disponíveis, leva a que os resultados obtidos sejam mais íntegros, e que possa ser usada por qualquer investigador que possua ou não conhecimentos na área da bioinformática.

Palavras-chave: interações proteína-proteína, Docker, bioinformática, estrutura proteica, bash scripting, técnica in silico

Agradecimentos

Aos meus orientadores Jorge e Cristina pela paciência, dedicação e sabedoria com que ao longo deste tempo todo me ensinaram e orientaram o trabalho. Pela segunda oportunidade com que me presentearam, oferecendo sempre a mesma simpatia e carinho para comigo, o meu profundo obrigado.

Ao meu orientador Hugo pela simpatia e prontidão com que sempre me ajudou. Obrigada por me mostrar que não há barreiras linguísticas à partilha de conhecimento científico.

Ao Professor Doutor Fernando Tavares, pela enorme compreensão, dedicação e preocupação que teve para comigo. Pela forma como encarou e procurou solucionar as minhas adversidades e aconselhar-me, o meu eterno obrigado.

À senhora Dona Maria Prazeres, Lucinda Pinto, Alice Santos, Sofia Santos, Alice Teixeira, César Guedes e os restantes trabalhadores das secções de pré- e pós-graduação, pela inesgotável paciência e extrema vontade em me ajudar a resolver os problemas e orientar o meu percurso académico por entre os diversos altos e baixos.

Aos meus pais Luís e Inês, por me darem não só a vida como também todas as ferramentas para eu poder chegar até aqui. Por sempre me apoiarem e me fazerem rir com as parvoíces de sempre, por me aturarem o mau feitio em momentos de stress e ansiedade e por estarem sempre cá para me dar consolo nos dias chuvosos. Espero que estejam sempre presentes e orgulhosos de mim, nesta e nas minhas próximas conquistas. A vocês, o maior obrigado.

Ao meu namorado Gabriel, por ter aparecido na minha vida quando eu mais precisava. Por me teres ajudado nas piores alturas e me ensinares a ver o mundo em tons de cor de rosinha. Por todas as nossas aventuras improvisadas e todos os calhaus que explorámos e que ainda temos por explorar. Obrigada pelos teus “apatás”.

“God, grant me the **serenity** to accept the things I cannot change,
courage to change the things I can,
and **wisdom** to know the difference.”

Reinhold Niebuhr

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AF – AlphaFold
AI – Artificial Intelligence
API – Application programming interface
ATXN1 – Ataxin 1
CASP - Critical Assessment of protein Structure Prediction
exp- – Expanded
LOMETS – Local Meta-threading Server
LXC – Linux containers
MC – Monte Carlo
MD – Molecular Dynamics
MSA – Multiple sequence alignment
NMR – Nuclear magnetic resonance
OS – Operating System
PDB – Protein Data Bank
polyQ – Polyglutamine
PPI – Protein-protein interactions
RMSD – Root-mean-square deviation
SCA1 – Spinocerebellar ataxia type 1
TAP – Tandem affinity purification
VM – Virtual Machine
wt- – Wild-type
Y2H – Yeast two-hybrid

Chapter 1

Introduction

1.1. Protein-protein Interactions and their Characteristics

Proteins are biomolecules consisting of one or more chains of amino acids, which sequence determines its unique 3D structure and cellular function. They constitute an important biological building block that fulfils their functions by interacting with each other (Alberts et al. 2002; Phizicky and Fields 1995; Tripathi et al. 2019).

Protein-protein interactions (PPI) describe the specific contact between two or more proteins within the cellular milieu. For an interaction to occur, there needs to be contact between residues of one protein with the residues of another (Nooren and Thornton 2003a; De Las Rivas and Fontanillo 2010). These participating residues constitute the interface of the PPI. Additionally, for an interaction to be formed, it is essential, not only for the interacting proteins to have a high degree of shape complementarity – directly causing a high specificity of PPIs –, but also for the contacting amino acids to have an acceptable degree of chemical complementarity, which facilitate the formation of hydrophobic, polar or charged interactions between them (Cohen et al. 2008; Bahadur et al. 2004; Tripathi and Varadarajan 2014; Nooren and Thornton 2003a).

Despite the high diversification shown by protein-protein interactions a few general rules can be deduced from studies on PPIs structural characteristics (Neuvirth, Raz, and Schreiber 2004a; Yan et al. 2008; Dai et al. 2016; Nooren and Thornton 2003a):

1) Interfaces of permanent PPIs are generally larger and more hydrophobic than transient complexes. And consequently, the surface of obligatory PPI interfaces is more hydrophobic that that of non-interacting surfaces (Cohen et al. 2008; Lo Conte, Chothia, and Janin 1999; Jones and Thornton 1996; Mintseris and Weng 2005).
2) Residues on the interface are evolutionarily more conserved than the rest of the protein, more particularly for obligatory interactions than transient ones (Levy 2010; Mintseris and Weng 2005). Additionally, these same residues have been shown to overlap quite well with interfacial hot spots (Keskin, Ma, and Nussinov 2005; Li et al. 1998).
3) Hot spots are a few key interface residues that significantly contribute with most of the free binding energy between proteins. These are further defined as residues that cause an increase in the binding energy of at least 2.0 kcal/mol, by alanine mutations (Thorn and Bogan 2001) and research has shown that their amino acidic composition consists fundamentally of tryptophan (21%), arginine (13.3%) and tyrosine (12.3%) (Lichtarge, Bourne, and Cohen 1996; Bogan and Thorn 1998), the latter being particularly abundant in the interface of antibodies (Fellouse, Wiesmann, and Sidhu 2004).
4) Protein-protein interactions whose interfaces have a surface area greater than ~1000 $Å^2$ are expected to undergo changes in conformation after complexation (Nooren and Thornton 2003b; Lo Conte, Chothia, and Janin 1999) and the triggers that result in these changes can be related to the stability of the complex (Nooren and Thornton 2003b), inducing stronger molecular changes the more powerful they are.

Even though the contact between proteins is necessary for an interaction to unfold, this does not mean that the interaction is static or permanent nor that they happen all the time (De Las Rivas and Fontanillo 2010). In fact, conditions such as the cell type, environmental conditions, protein phosphorylation and the presence of cofactors or other binding partners, affect the activity and PPI occurrence so that interactions do not happen by mere chance (Nooren and Thornton 2003a; De Las Rivas and Fontanillo 2010). Other factors like mutations can alter the interface which risks the stability of the protein complex, leading to cellular dysfunction and consequently triggering pathologies and

aiding their aggravation (Schreiber, Haran, and Zhou 2009; Ryan and Matthews 2005).

PPI interactions are of utmost importance for the organism's homeostatic balance, allowing for the building of macromolecular structures and enzymatic complexes which are the basis of just about every cellular function, going from transport activity and signal transduction to the catalysis of metabolic reactions, substrate binding or inhibition and biomolecular synthesis (Braun and Gingras 2012; Tripathi et al. 2019). Due to the importance of such regulatory processes in the proper functioning of the living being, any changes that might reverberate on the PPI's integrity are naturally of interest to the enrichment of the scientific knowledge of biological processes and the understanding of genotype-phenotype relationships and their involvement in the onset of complex diseases (Tripathi et al. 2019).

1.2. PPI Prediction and Detection methods – a brief overview

Proteins hardly ever act isolated during the completion of their functional roles *in vivo*. In fact, studies have shown that just about 80% of proteins operate in complexes in lieu of acting alone (Berggård, Linse, and James 2007; Yanagida 2002). The inference of protein functions within the cell depends highly on examining protein-protein interaction properties. For instance, by analysing the interaction of a protein whose function is already known with one whose role has not yet been made clear, it is possible to gather information on the ultimate function of that previously unidentified protein.

Analysing PPI interface information helps in the identification of novel drug targets for the treatment of various diseases (Pedamallu and Posfai 2010; Jubb, Blundell, and Ascher 2015; Wells and McClendon 2007; Murakami et al. 2017). PPI prediction also poses a crucial step for the construction of interaction networks for humans as well as other organisms (Kovács et al. 2019).

Protein-protein interaction detection methods are mainly grouped as *in vivo*, *in vitro* or *in silico* experiments (a summary of these methods is presented in (Rao et al. 2014), Table 1). *In vivo* and *in vitro* methods, such as yeast two-hybrid (Y2H), affinity purification and tandem affinity purification (TAP), have several limitations (von Mering et al. 2002; Rao et al. 2014). Not only are these prediction methods highly expensive, laborious and time-consuming, but they have many other drawbacks, namely: (1) the resulting data sets can be noisy and hugely prone to false positives, which in phylogenetic profile-based methods can originate from gene duplication or gene loss, and other genomic events during the course of evolution (Lin, Wu, and Chang 2013), (2) these methods depend on how adequately experimentation protocols are established in the applied target organisms, which ultimately can lead to them failing to predict certain types of proteins (Rao et al. 2014; Piehler 2005), (3) due to their unstable nature, many transient and weak interactions end up being excluded, due to the experimental methods' struggle to accurately predict them (Ngounou Wetie et al. 2013), and finally (4) large-scale methods have been shown to have a significant dissent with each other which can be in part due to false positives and false negatives (Huang and Bader 2009; Gingras et al. 2007; Serebriiskii and Golemis 2001; Berggård, Linse, and James 2007).

Because of these many drawbacks, it is important to validate protein-protein interactions using several methods. But where all these different procedures fail at predicting PPI 100% accurately, *in silico* methods can be of great value when applied to obtain a set of most-likely interactions from a wider set of interactions. The use of computational methods for the prediction of protein-protein interactions is not only cheaper than most large-scale detection assays, but it also allows to obtain data sets with higher quality and confidence, as well as being appreciably less labour-intensive. This will ultimately improve the confidence of PPI and corresponding interaction network by originating a starting point for upcoming lab experiments (Rao et al. 2014; Shatnawi 2015).

1.3. *In silico* comparison of interaction properties involving different forms of a protein

In (Rocha et al. 2019) two different forms of the Ataxin-1 (ATXN1) protein, which is responsible for causing spinocerebellar ataxia type 1 (SCA1), a neurodegenerative disorder that is caused by a polyglutamine (polyQ) expansion, were studied. Wild-type Ataxin-1 (wt-ATXN1) and expanded Ataxin-1 (exp-ATXN1) protein-protein interactions were analyzed with the aim of understanding how the expanded polyQ tract affects PPI and which ATXN1 regions explain the binding differences. The interfaces of wt-ATXN1 and exp-ATXN1 with 71 interacting partners were inferred using an *in silico* methodology to address whether the exp-ATXN1 binds with higher affinity to the reported interactor proteins and compared their results with those obtained by (Suter et al. 2013) and (Lim et al. 2006).

This *in silico* approach uses the three-dimensional protein structures predicted using I-TASSER (Zhang 2008), the predicted interfacing residues from CPORT (de Vries and Bonvin 2011), the protein-protein dockings using HADDOCK (van Zundert et al. 2016), and lastly, features about the interface of the protein clusters drawn from HADDOCK are acquired using PDBePISA (Krissinel and Henrick 2007). In (Rocha et al. 2019), Supplementary information, Figure S6) there is a flowchart of the methodology therein used.

The authors obtained insights into binding preferences caused by the expansion of the polyQ tract, binding surface prediction and possible functional significances of that polyQ expansion, using that *in silico* methodology. Nevertheless, it is important to note that the method has some limitations. The most relevant one is the size of the proteins considered, since I-TASSER is unable to infer the 3D structure of proteins larger than 1500 aa, and the fact that this method only takes into consideration binary interactions, that is, interactions of only a pair of proteins at a time, therefore being unable to infer interfaces of larger complexes. Furthermore, and because I-TASSER uses the threading templates from the PDB library with the highest significance, the polyQ region tract has less amino acids used as threading templates (Yang et al. 2015). This fact makes the structural prediction of these regions less reliable, therefore, this approach is expected to be more robust when no interactions are predicted with a polyQ expansion.

The *in silico* approach used in the above experiments can be employed for any biological system where the focus is the inference of PPI interfaces for any pair of interacting proteins. This was one of the main motivations for the development of the pipeline which will be described in the upcoming chapters.

1.4. Containerization with Docker images

Containerization is a means of running applications in isolated executables, called containers, all running under the same operating system (OS). Inside a container lies all dependencies, libraries, binaries, configuration files and other important resources that are required for an application to run. Each container is abstracted from the host OS only being able to access a minimal set of resources, which consequently allows the underlying OS to run several containers concurrently. It is commonly used for the packaging of the many components of an app that are designed to run independently.

Docker was launched to the public in 2013 and has since been the most prominent container technology. It has its basis in Linux containers (LXC) which is a feature for isolating and controlling resource usage within the Linux kernel (Waldspurger 2003), but later switched to libcontainer (now known as runC) which runs in the same OS as its host, thus allowing the sharing of a lot of the OS resources. A docker image is a template containing a set of instructions in order to create a container that has application code, libraries and all dependencies an application needs to run. These images are built in layers, each deriving from the previous one but always being different from it. On top of these, the “container layer” is a writable thin layer that is created once the container starts to run, which stores all changes to a container (creating, writing, deleting files, etc.) during its runtime. This way, different running containers can share the same underlying image while preserving their own individual state, and any user has an identical experience regardless of their computer environment

(Boettiger 2015). This multi-layered framework not only speeds up Docker builds, but also provides the portability that, for instance, a virtual machine (VM) lacks due to their intimate tie to the underlying OS (Bashari Rad, Bhatti, and Ahmadi 2017).

Therefore, Docker is ideal to provide ready-to-use pipelines when multiple applications are involved.

1.5. Motivation

Predicting PPIs and obtaining information about their interfaces is not always cost-effective, in the way that *in vivo* and *in vitro* methods not only waste a lot of time and lab resources, but in the end, they yield noisy data with the presence of false positives and false negatives. *In silico* methodologies certainly help with scaling down a wide set of original interactions to a group of more accurate and high-quality interactions and this was one of the main motivations to develop an automated pipeline for the study of PPI.

A second main motivation was providing a way of performing analyses related to those presented in (Rocha et al. 2019) effortlessly. These require obtaining the 3D structures of both interacting proteins, then predicting the interacting residues. Next, a model for the docking of the interaction is obtained and lastly, information about the PPI interface is inferred. All these stages are usually completed in a three-step process of submitting, waiting for the results, and downloading said results from the webserver of each integrated bioinformatics program or, in some cases, through the software's application programming interface (API) installed in the user's local computer environment.

Having to manually submit data in webservers of bioinformatics programs for the prediction of structural, interface, and docking information, is time consuming. Additionally, the researcher must manually download the results for all pairs of interactions which can single-handedly last an extra couple of days, time that could instead be used for other analyses and lab experiments, not to mention that this way it is harder to avoid data inconsistencies through errors in the data output.

Containerizing the process into a Docker image has several advantages such as:

1) It is made available to everyone, following the intent of respecting open science principles.
2) It abolishes the need to install several software programs. Hence saving time, computer storage and avoiding the headaches of going through hard-to-follow installation instructions, the nuisances of missing dependencies and compatibility issues.
3) It is easy to run and portable amongst OS. Docker does not require a vast informatics knowledge to be used, only requiring its installation which is very well documented, due to its popularity.

For these reasons, having a pipeline provided on a Docker image that automates the process of inferring PPIs interfaces, unifies all required tools, and provides an alternative to skip some prediction steps given available data, allows for the developed methodology to be less susceptible to errors and to reach laboratory researchers with or without expertise in the bioinformatics field.

1.6. Thesis outline

This thesis was organized into mainly five chapters, namely:

Chapter 1 – Introduction. Provides the reader with background on the topic, on its main motivations and previous work that served as basis for the development of the pipeline herein described.

Chapter 2 – Related work. A brief review of previous research on protein-protein interactions prediction, methodologies, and existing bioinformatics tools.

Chapter 3 – Building and running the Docker image. Overview and explanation of the pipeline's workflow, with details of the implementation of the pipeline and how to set for a specific analysis.

Chapter 4 – Testing the Docker image. Analysis of the Docker image runtimes, with data from (Rocha et al. 2019) and analysis of protein structures from the recently launched AlphaFold DB.

Chapter 5 – Conclusions. A summary of the work leading to the pipeline, along with its limitations and future work.

Chapter 2

Related work

In this section I will provide some insights into *in silico* methods for the prediction of PPI, databases where interaction data can be found, and for each method used in the pipeline's workflow I will briefly describe each one and present a few existing alternatives.

Protein-protein Interactions – Prediction methods

Several in silico methodologies have been developed throughout the years, showing the extreme importance that these bioinformatics tools pose in the prediction and validation of PPI. There are several approaches inside these computational approaches, namely: sequence-based approaches, structure-based approaches, chromosome proximity, gene fusion, in silico 2 hybrid, phylogenetic tree, phylogenetic profile, and gene expression (a summary of each type of approach is presented in (Rao et al. 2014), Table 1). In this chapter, I present a brief description of a few approaches

Inside structure-based approaches, the PrePPI[1] algorithm uses three-dimensional structural information to predict PPIs. It predicts interactions using a naive Bayesian network that combines structural, functional, evolutionary and expression information. It has shown accuracy levels comparable in terms of accuracy to high throughput methods and its effectiveness relies mainly on the use of homology models combined with the exploitation of both close and remote geometric relationships between proteins (Zhang et al. 2012).

Under the sequence-based approaches, there are the domain-pairs-based prediction methods, which use the fact that domains are highly conserved throughout evolution to infer protein-protein interactions from the consistent and reliable interactions between domains (Memišević, Wallqvist, and Reifman 2013; Wojcik and Schächter 2001). One of these techniques is IDPP (Interacting Domain Profile Pairs), which clusters protein domains by sequence and connectivity similarities. It predicts a target organism's PPI map from the PPI map of a reference organism; therefore, it essentially transfers obtained interactions by homology (Wojcik, Boneca, and Legrain 2002). By choosing to use the sequence similarity between interacting domains instead of full length proteins, the authors show that their approach reduces the false-positive rate induced by multi-domain proteins. Furthermore, by combining sequence and interaction information, it allows for flexible sequence patterns correlated to physically interacting structures, which enhance prediction sensitivity, to be identified (Wojcik and Schächter 2001).

Several existing methods combine computational approaches with machine learning algorithms, which seek to improve the quality of the predicted outcome. An example of such is UNISPPI, a method developed by (Valente et al. 2013) that is based on primary sequence information for classifying protein pairs as interacting or noninteracting proteins. It is a machine learning-based approach which uses information on the primary sequences and a small set of features, for the prediction of interaction of protein pairs. Because it is based on experimentally validated data from diverse species and can be applied to predict proteins that interact or not in a large number of species, it truly becomes a universal predictor, that uses only the frequencies of amino acids in a protein pair. It surpasses limitations that some machine learning methods have, such as the small number of examples, building of negative datasets and the large number of attributes that code protein pairs (Valente et al. 2013). In (Macalino et al. 2018), there is a listing of the machine-learning-based predictors for the identification of protein-protein interactions, showing that there is a wide number and variety of these methods.

[1] https://bhapp.c2b2.columbia.edu/PrePPI/

With the uncovering of novel and unknown interactions, leveraged by methods such as those mentioned above for the prediction of PPI, emerges the need of keeping track of information involving their interfaces and biological function. Many useful databases that store this data regarding protein-protein exist, such as BioGRID[2] (Chatr-Aryamontri et al. 2017; Stark et al. 2006), STRING[3] (Snel et al. 2000; Szklarczyk et al. 2021), EvoPPI[4] (Vázquez et al. 2019b, 2019a), PANTHER[5] (Thomas et al. 2003; Mi et al. 2019), KEGG[6] (Kanehisa 2000; Kanehisa et al. 2021), MINT[7] (Licata et al. 2012; Orchard et al. 2014), and FlyBase[8] (Attrill et al. 2016). A more complete source of the primary databases and meta-databases for PPI information is present in (Macalino et al. 2018), Table 2). These databases can be therefore used as a means to obtain data to run with this pipeline.

Three-dimensional protein structure prediction

Predicting the 3D structure of proteins is a common problem, which has challenged molecular biologists, biochemists, computer scientists and other scientists for the past decades (Dorn et al. 2014; Baxevanis and Ouellette 2001). Existing methods can be categorized into template-based prediction and free modeling methods (also designated *ab initio* or *de novo* modeling), according to the availability of structural templates in the PDB. Template-based methods can be further divided into two categories: (1) Comparative Modeling, which aim to build the three-dimensional model of the query protein based on the sequence similarity with proteins with known structures (Martí-Renom et al. 2000), and (2) Threading methods, when a query protein sequence is matched with solved protein structures from different evolutionary origins to recognize similar folds, even if they do not show any evolutionary relationship (Bowie, Lüthy, and Eisenberg 1991). Template-free methods are employed when no structurally related proteins for a query are found in the PDB and therefore the structure must be built from scratch (and and Baker 2001).

Under *ab initio* methods, there are several developed algorithms such as AMBER (Weiner et al. 1984), CHARMM (Brooks et al. 1983) and OPLS (Jorgensen and Tirado-Rives 1988), which belong to the physics-based methods, combined with molecular dynamics (MD) conformational sampling, and QUARK[9] (Xu and Zhang 2012), ROSETTA[10] (Simons et al. 1997; Das et al. 2007), and I-TASSER[11] (Zhang 2008), inside knowledge-based methods, combined with Monte Carlo (MC) simulations.

Compared to template-based approaches, physics-based *ab initio* methods – like AMBER, CHARMM and OPLS – have been less successful in protein structure prediction, being generally limited to the structural prediction of small proteins (<120 residues). Although several efforts have been made to improve accuracy, physics-based folding is still far from routine for normal sized proteins mainly because of its excessive computing demand. However, it is possible to obtain more promising results in both structure prediction and refinement by combining fast search methods, such as MC simulations and genetic algorithms, with physics-based force-fields (Lee, Freddolino, and Zhang 2017).

The most well-known approach for efficient free-modeling was firstly described by (Bowie and Eisenberg 1994), which involves the generation of protein models by assembling small fragments cut from other proteins in the PDB library. This idea was the basis for the development of ROSETTA (Simons et al. 1997) which has shown great results in the free modeling targets of Critical Assessment of protein Structure Prediction (CASP) experiments and popularized the fragment assembly approach. This method acts in a two-phase manner, where firstly the authors generate models in a

[2] https://thebiogrid.org/
[3] https://string-db.org/
[4] http://evoppi.i3s.up.pt/
[5] http://www.pantherdb.org/
[6] https://www.genome.jp/kegg/
[7] https://mint.bio.uniroma2.it/
[8] http://flybase.org/
[9] https://zhanggroup.org/QUARK/
[10] http://www.robetta.org/
[11] https://zhanggroup.org/I-TASSER/

reduced form with conformations specified with heavy backbone and Cβ atoms. Then, MC simulations are performed with an all-atom physics-based potential, to refine the details of the low-resolution models.

Lastly, TASSER (Zhang and Skolnick 2004a), another successful free modeling method, constructs 3D models based on a purely knowledge-based approach. First the target sequence is threaded through a set of representative protein structures to search for possible folds. Contiguous fragments with various sizes are excised from the threaded aligned regions and used to reassemble protein structures, while unaligned regions are built by a lattice-based *ab initio* modeling. Recent versions of I-TASSSER (Roy, Kucukural, and Zhang 2010; Yang et al. 2015; Yang et al. 2016) refine TASSER cluster centroids by iterative MC simulations, and although the procedure uses structural fragments and spatial restraints from threading templates, it often constructs models of correct topology even when the topologies of individual templates are incorrect. I-TASSER has been consecutively ranked as one of the best automated protein structure prediction methods, from CASP7 to CASP14 experiments (Kinch et al. 2016; Battey et al. 2007).

Structural Alignment

Insights on the similarity between structures is extremely relevant, especially if proteins share any evolutionary history. Knowledge of these relationships can aid scientific advances in the field of structural biology, in branches such as protein fold classification (Hubbard et al. 1997), protein structure modeling (Moult et al. 2003) and structure-based protein function annotation (Skolnick, Fetrow, and Kolinski 2000; Baker and Sali 2001). Usually, a protein structure alignment method consists of two major components: a scoring function that measures similarity, and a search algorithm for optimization of the scoring function. Many computer programs for pairwise and multiple structural alignment have been developed in the past two decades. Some popular structure alignment methods include: DALI (Holm and Sander 1993), a distance matrix-based approach, that aligns two structures by matching their distance matrices; TM-align[12] (Zhang and Skolnick 2005), which aligns two protein structures by means of maximizing the TM-score (Zhang and Skolnick 2004b) and uses a similar search algorithm as STRUCTAL (Levitt and Gerstein 1998), a weight factor that weights the residue pairs at smaller distances relatively stronger than those at larger distances; MATT (Menke, Berger, and Cowen 2008), a method that aligns structures by concatenating the alignments of short fragmented structures, and measures similarity by calculating a p-value; Formatt (Daniels, Nadimpalli, and Cowen 2012) extends MATT's principles by taking into consideration primary sequence similarity in aligning two structures.

More recently some methods have been developed which show performances very close, or in some cases comparable to that of TM-align. It is the case of the Phenotypic Plasticity Method (PPM) (Csaba, Birzele, and Zimmer 2008), DeepAlign (Wang et al. 2013) and UniAlign (Zhao and Sacan 2015).

PPM is based on the observation that, despite conformational changes, protein structure families express a high flexibility on a smaller scale. This concept, phenotypic plasticity, comprises the changes in the actual 3D protein structure which are to be expected for the given genotype or genotype population, and is additionally based in evolutionary events that happen on the sequence level, namely mutations, insertions, and deletions. This method tries to model the evolutionary distance of two protein structures by explicitly measuring the cost of "morphing" one structure into the other one. It aligns protein structures by considering the phenotypic plasticity, while preserving the overall arrangement of the structures (Csaba, Birzele, and Zimmer 2008).

Another relevant method is DeepAlign, for automatic pairwise protein structure alignment. It aligns two protein structures using spatial proximity of equivalent residues, evolutionary relationships, and hydrogen-bonding similarity. The authors show that this method can produce alignments much more consistent with manually-curated alignments, compared to other automatic tools, especially when the proteins are remote homologs. These results suggest that evolutionary information and hydrogen-

[12] https://zhanggroup.org//TM-align/

bonding similarity are essential to the alignment of protein structures, in addition to geometric similarity (Wang et al. 2013).

The last method, UniAlign, incorporates different kinds of information namely evolutionary information captured in the form of sequence similarity, sequence profiles and residue conservation to achieve a more accurate alignment. The authors define a per-residue score (UniScore) as a weighted sum of features and develop an iterative optimization method to search for an alignment with the best overall UniScore. The resulting comparison of UniAlign with methods that only use the geometric relationship between proteins shows that it is capable of producing alignments that are in better agreement with hand-curated reference alignments. Although it shows robustness regarding sequence homology and increased accuracy, it suffers from an increased demand in computing time (Zhao and Sacan 2015).

Prediction of PPI Interface Residues

The prediction of interface residues of an interaction is essential for understanding how proteins interact with each other and is an extremely useful step for guiding protein docking. It aims to identify which residues on a protein surface interact with another protein and is based on the extraction and combination of distinguishing features from protein sequences and structures. PPI residues can be identified by experiments such as alanine scanning (Bradshaw et al. 2011; DeLano 2002), chemical modification of residues (Speck et al. 1981; Dhungana, Fessler, and Tomer 2009), hydrogen/deuterium exchange (Anand et al. 2003), disulfide cross-linking (Meenan et al. 2010), NMR (Bonvin, Boelens, and Kaptein 2005), and can furthermore be predicted by computational methods.

ProMate[13] is a naïve Bayesian program which is based on structural properties such as secondary structure, atom distribution, amino-acid pairing, and sequence conservation, which are specific to interfaces. By identifying these properties, it is possible for the algorithm to detect the residues involved in the interface of the unbound proteins. This method focuses solely on transient complexes due to their stability and functionality in both bound and unbound states. Despite the difficulty posed by the great diversity of PPI, the authors still managed to obtain a success rate of 70% in prediction of interface residues (Neuvirth, Raz, and Schreiber 2004b). Based on ProMate's infrastructure, ProMateus extends this program by enabling an easy exploration and incorporation of new features and databases by the user, ultimately providing an evaluation of the individual features' benefit and their combination within a set framework (Neuvirth et al. 2007).

Another relevant method for the prediction of interfacial residues is InterProSurf[14] which was developed based on statistical analyses of known protein complexes. This prediction method is solely based on the solvent accessible surface area of the protein and propensity of amino acid residues and does not include any information regarding the partner protein (Negi et al. 2006). By joining a patch analysis and cluster analysis method, this tool has obtained an overall accuracy of 70%, which can be increased by combining evolutionary information or correlated mutations across interfaces (Negi and Braun 2007).

In the recent years, PredUs 2.0[15] was published, which is based in PredUs (Zhang et al. 2011) and combines a novel patch-based score. It calculates two scores, one using PredUs and another using patch features, for each residue on a query protein surface, for which a Bayesian approach determines the likelihood ratio of both values. Then a final likelihood ratio is calculated, that reflects if a specific residue is interfacial (Hwang, Petrey, and Honig 2016). The authors show this method to be quite effective in predicting interfacial residues and outperforms both the original PredUs software and other known interface prediction methods, namely PINUP (Liang et al. 2006a) and cons-PPISP (Chen and Zhou 2005b).

Although these methods show promising levels of accuracy, CPORT (Consensus Prediction Of interface Residues in Transient complexes)[16] provides more stable and reliable interface predictions

[13] http://bioportal.weizmann.ac.il/promate/
[14] http://curie.utmb.edu/prosurf.html
[15] https://honiglab.c2b2.columbia.edu/PredUs/index_omega.html
[16] https://alcazar.science.uu.nl/services/CPORT/

by generating a consensus between each individual predictor that makes up this well-known method (de Vries and Bonvin 2011). CPORT combines the prediction scores of six predictors, namely WHISCY (de Vries, van Dijk, and Bonvin 2006), PIER (Kufareva et al. 2007), ProMate (Neuvirth, Raz, and Schreiber 2004b), cons-PPISP (Chen and Zhou 2005a), SPPIDER (Porollo and Meller 2007), and PINUP (Liang et al. 2006a), to reach a consensus interface residue prediction. Additionally, it has shown better performance than its best individual predictor, PINUP.

Protein Docking prediction

Protein docking aims to predict the 3D arrangement of a biologically relevant complex structure by using the coordinates of its component biomolecules. It is fundamental for the understanding and prediction of molecular recognition, by finding likely binding modes and predicting binding affinity, ultimately gaining attention due to its potential applications in protein engineering and drug design (Janin 2005). Nevertheless, the difficult nature of crystallizing protein-protein complexes, results in a scarce record of available structural information, when compared to individual proteins (Smith et al. 2005). Experimental studies are thus faced with greater technical difficulties aggravated by the small number of solved complexes deposited in the PDB library. Computational approaches can then guide new experiments and aid in the inferring of complex functional properties (Moreira, Fernandes, and Ramos 2010). There are several methods for protein docking due to the large diversity of characteristics they implement, namely the searching algorithm, the methods for scoring the results, the use of biological information, backbone flexibility and symmetry (Moreira, Fernandes, and Ramos 2010, Table 4).

ZDOCK[17] is a rigid body algorithm that uses the Fast Fourier Transform to allow for an efficient global docking search on a 3D grid. It combines shape complementarity, electrostatic cans statistical potential terms for scoring. This method shows 70% success in predictive accuracy on protein-protein docking benchmarks (Pierce, Hourai, and Weng 2011) and consistent success in the international Critical Assessment of Predicted Interactions (CAPRI) docking experiment (Hwang et al. 2010; Vreven et al. 2013; Wiehe et al. 2005).

Another approach worth mentioning in protein docking is RosettaDock[18]. This method uses a low-resolution rigid-body MC search and a real-space MC minimization to identify the lowest free energy of the docked protein complex. After the search, using a backbone-dependent rotamer packing algorithm, explicit side chains are added to the protein backbones. Once the proteins have side-chains, the rigid-body displacement is optimized. To simultaneously optimize the conformations of the side-chain and the rigid-body position, the sidechain packing and minimization operations are repeated 50 times. A large number of decoys is created with supercomputing clusters which are discriminated using a scoring function that uses van der Waals and solvation interactions, hydrogen bonding, residue-residue pair statistics, and rotamer probabilities. The final predictions are then selected from the clustered top ranked decoys (Gray et al. 2003; Lyskov and Gray 2008).

BonvinLab's HADDOCK (van Zundert et al. 2016; Dominguez, Boelens, and Bonvin 2003) is currently the most popular method for protein docking, showing an overall great performance in CAPRI experiments (Lensink and Wodak 2013; Janin 2005). This data-driven method starts with a randomization of orientations and rigid body energy minimization, followed by semirigid simulated annealing in torsion angle space, and final refinement in Cartesian space with explicit solvent. During the torsion angle space simulated annealing and the water refinement phases, the amino acids at the interface (side chains and backbone) are allowed to move to optimize the interface packing. It is able of integrating a diverse set of information from biochemical, biophysical or bioinformatics methods to reduce the conformational search space, or filter the solutions, or both. One of the most fundamental differences in comparison with other algorithms is that HADDOCK translates information about the interface into highly ambiguous intermolecular distance restraints used to directly drive the docking process (Dominguez, Boelens, and Bonvin 2003).

[17] https://zdock.umassmed.edu/
[18] https://rosie.graylab.jhu.edu/

Protein Structure analysis

The amount and diversity range of bioinformatics tools for the analysis of protein structures has been continuously increasing throughout the years, mainly due to the incessant growth of structural information, owning to the increase of fully sequenced genomes (greatly supported by next generation sequencing) and of protein structures submitted in the PDB (through advances in crystallography) (Paxman and Heras 2017).

For instance, PDBsum[19] offers a handful of tools to provide multiple structural analyses, such as analysis of all the grooves, pores and tunnels of a given protein structure, which are computed by Mole 2.0 (De Beer et al. 2014; Laskowski et al. 1997). An interesting fact is that recently, its server has included the possibility of using AlphaFold's predicted models for the available human proteome. Another similar web tool is eF-surf[20] from PDB Japan[21], that calculates the molecular surface of an uploaded structure file (Kinoshita, Murakami, and Nakamura 2007).

A more recent method, ProteinPlus[22] has been developed, although it is important to note that it focuses on protein-ligand interactions (Schöning-Stierand et al. 2020). It provides a collection of multi-purpose tools guided towards easy access for life scientists, that allows to perform structure search, quality assessment and pre-processing tasks such as hydrogen prediction (Protoss (Bietz et al. 2014)), binding site detection (DoGSiteScorer (Volkamer et al. 2012)), binding site ensemble generation (SIENA (Bietz and Rarey 2016)) and PPI classification (HyPPI (Schneider et al. 2013)).

Last but not least, PDBePISA[23] is the most well-known tool for the exploration and analysis of protein structures surfaces and quaternary structure prediction. This interactive tool allows for the automatic identification of probable quaternary structures, based on physical-chemical models of macromolecular interactions and chemical thermodynamics. Additionally, it provides features that are not available in other traditional bioinformatic techniques, namely: the results are delivered in conventional chemical notations, macromolecular interactions are detailed on residue level so that residue substitution effect may be easily modelled and interpreted, symmetry effects are taken into account, chemical stability of macromolecular complexes is estimated in Gibbs free energy terms along with suggestion of probable dissociation patterns (Krissinel and Henrick 2007).

[19] https://www.ebi.ac.uk/thornton-srv/databases/cgi-bin/pdbsum/GetPage.pl?pdbcode=index.html
[20] https://pdbj.org/eF-surf/top.do
[21] https://pdbj.org/
[22] https://proteins.plus/
[23] https://www.ebi.ac.uk/pdbe/pisa/

Chapter 3

Building and running the Docker Image

3.1. Introduction

The will to automate the methodology for inferring PPI interfaces and analyse their information as (Rocha et al. 2019) have employed, culminated in the building of P_2I_2P (Protein-Protein Interactions Interface Prediction), a bash script-based pipeline which consequently was turned into a Docker image for better portability and easiness of use.

Very broadly defined, this Docker image runs a multi-software bioinformatics pipeline that allows for the identification and characterization of protein-protein interactions interfaces and their properties. It is divided into four main steps. Starting from a set of Reference and Interactor proteins, it allows the user to:

- Obtain the 3D structure of all submitted proteins, with I-TASSER (Zhang 2008) and select the most approximate one using TM-align (Zhang and Skolnick 2005).
- Predict interface active and passive residues, with CPORT (de Vries and Bonvin 2011).
- Predict the structure of the interacting proteins complex, with HADDOCK (van Zundert et al. 2016).
- Obtain interface information, with PISA (Krissinel and Henrick 2007).

After this brief introduction to the underlying software of each step of the pipeline I will proceed to further elaborate on each one (section 3.2) then explain what the internal Configuration File is and how to fill it (section 3.3) before explaining the different ways the user can execute the pipeline (section 3.4) and where the final results are located (section 3.5).

It is worth noting that, throughout this thesis, I am assuming that the user is running a Linux-based OS, although the same procedures can be taken using any computer environment, as long as Docker is installed and this docker image is available in the host computer.

P_2I_2P is available locally[24] and will be later released as a Docker image. After downloading the zipped file, the user will have to run the command "*sudo docker load -i <filename.tar>*" to load the built image. After filling the configuration file (section 3.3) and providing the required files (section 3.4), to use the Docker image of the pipeline the user should run the following command: "*docker run --rm -v /var/run/docker.sock:/var/run/docker.sock -v $(pwd):$(pwd) pegi3s/pipeline_stable bash -c "/opt/main $(pwd)/user.config"* ".

3.2. Workflow overview

Each Docker image that makes up the basis for each step of P_2I_2P belong to a larger project called Bioinformatics Docker Images Project[25] (López-Fernández et al. 2022) which is a compendium of Docker images maintained by the Phenotypic Evolution Group (pegi3s) at the Instituto de Biologia Molecular e Celular (IBMC) / Instituto de Investigação e Inovação em Saúde da Universidade do

[24] https://www.dropbox.com/sh/aunlkeyrag8f3mt/AAAyxETsZwU93vkgMjdMLIeBa?dl=1
[25] https://pegi3s.github.io/dockerfiles/

Porto (i3S)[26].

The pipeline's workflow, as depicted in Figure 1, is mainly divided into two sides, based on the type of sequences that it works on (Reference and Interactors), and into five tasks, based on each program that the pipeline uses for each biological task. This is mainly because the execution of each step is fairly the same for both Reference and Interactors' proteins, with the exception of a few differences at the level of the data files and 3D model selection for the protein structure. In order to simplify its execution, the pipeline initially runs the References' side until the data regarding these proteins is completely available and ready, only then it begins the analysis of the interacting proteins.

Each of the five different tasks resort to data submission to the software's web server, being that each submission role has first been implemented into its own Docker image, which is deposited in the project's repository.

[26] https://www.i3s.up.pt/research-group.php?groupid=44

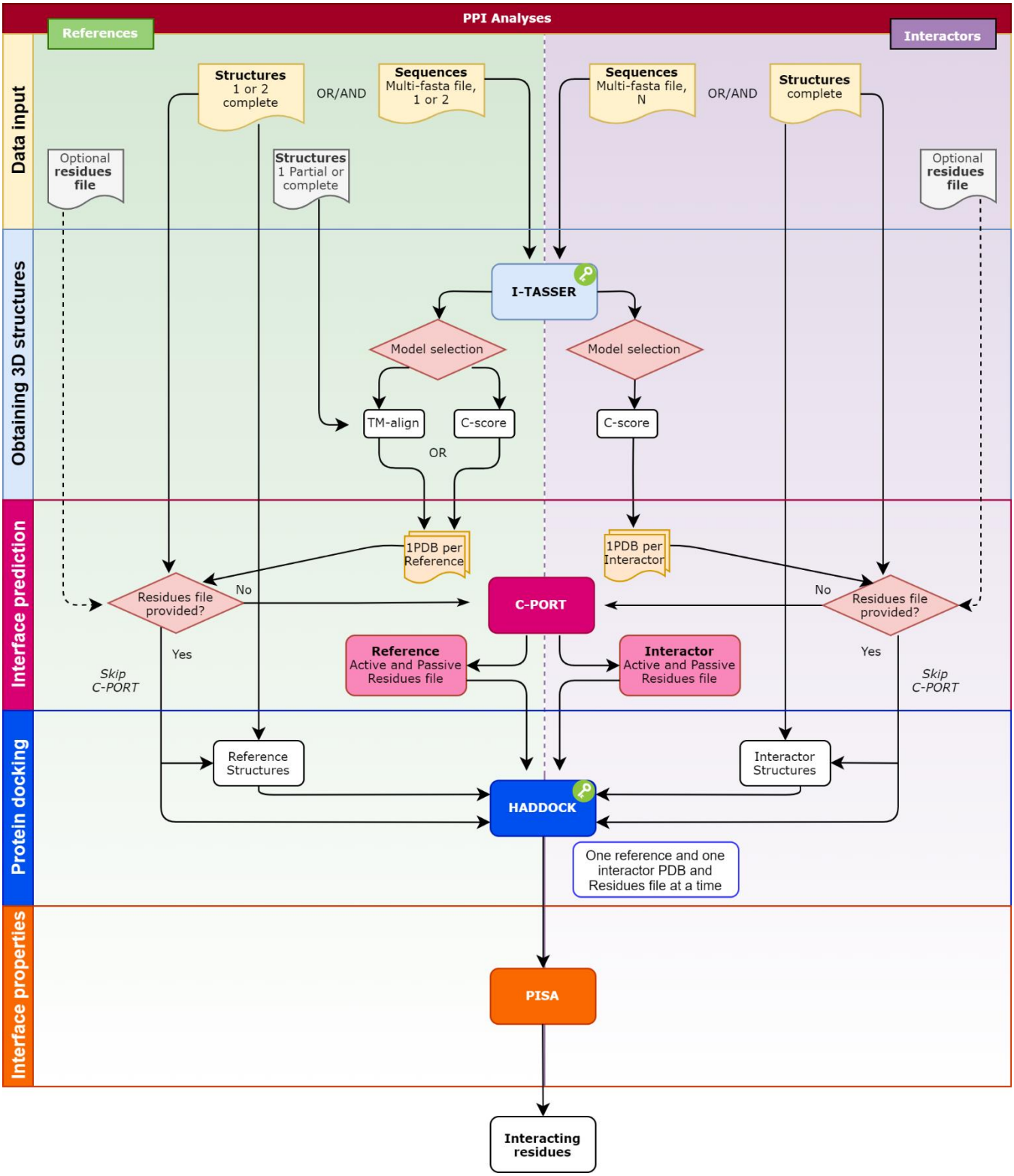

Figure 1 – Diagram of the pipeline's workflow. As seen above, the left side, in green, represents the References' execution and the right side, in purple, represents the Interactors'. Furthermore, the five main stages of the pipeline are each highlighted with it its own colour. Beige for the data input, baby-blue for the structure prediction, magenta for the interface prediction, blue for the protein docking and orange for the interface properties.

3.2.1. Getting started & Input files

To run the pipeline, it is mandatory that the user installs Docker on his OS. On the top of pegi3s Bioinformatics Docker Images Project's website[27], there is a small section explaining how to do so, although instructions can be easily found on the Docker guides[28] as well as all over the internet, given that Docker is so widely used.

Before the start of the analysis, there needs to be some organizing of the input files. Because the pipeline has a few steps that can be customized according to the user's intention, the input data can either be in the form of FASTA amino acid sequence files or Protein Data Bank (PDB) structure files. Details on where to put these files is going to be explained further down in section 3.4 due to it being linked to the customization of the pipeline.

Four preparation steps are set to run before every analysis. These are:

1) testWebServers – Tests if each webserver is up before the beginning of the analysis. If any server fails to respond or if the request times out, the user is warned and given the choice of aborting the start of the run or continuing at his own risk. This way, if a server is down at the moment the user is notified, and an unnecessary start of the pipeline is avoided.
2) checkConfigVariables – Checks if the variables given by the user in the *user.config* file are valid and if their values have been filled out correctly.
3) checkRequiredDirectories – Checks if the directories specified by the user truly exist and aborts the start of the run if they don't.
4) pullDockerImages – Lastly, but not least, this function pulls all the Docker images, from pegi3s, that are needed to run the pipeline. Namely, i-tasser_server[29], tm-align_server[30], cport_server[31], haddock_server[32] and pisa_server[33].

Since I-TASSER and HADDOCK are used, it is additionally required an account[34] in each software's web server.

3.2.2. Obtaining the three-dimensional structure

The prediction of the 3D structure of the submitted proteins is obtained with I-TASSER (Zhang 2008), which uses snippets of already known protein structures from LOMETS (Local Meta-Threading Server). These fragments are then assembled by replica-exchange Monte Carlo simulations to predict the whole structure of the protein. However, in cases where LOMETS cannot find an appropriate template, I-TASSER will build the structures by *ab initio* modelling. The model's quality is then assessed by the C-score, which is calculated based on the significance of threading template alignments and the convergence parameters of the structure assembly simulations (Zhang 2008). This scoring metric typically ranges from [-5, 2], being that a higher value denotes a model with a high confidence, which is obviously preferable.

Running I-TASSER generates five top models corresponding to the largest structure clusters. To select the model which will proceed further in the analysis, two approaches can be used, (1) the first model, which has the highest C-score and a better quality in most cases or (2) we compare each single one of the predicted five models with an external structure (partial or complete), that is provided by the user, and choose the model with the highest similarity to this structure using TM-align (Zhang and Skolnick 2005). TM-align is an algorithm for sequence independent protein structure comparisons

[27] https://pegi3s.github.io/dockerfiles/
[28] https://docs.docker.com/get-started/overview/
[29] https://hub.docker.com/r/pegi3s/i-tasser_server
[30] https://hub.docker.com/r/pegi3s/tm-align_server
[31] https://hub.docker.com/r/pegi3s/cport_server/
[32] https://hub.docker.com/r/pegi3s/haddock_server/
[33] https://hub.docker.com/r/pegi3s/pisa_server
[34] https://zhanggroup.org/I-TASSER/registration.html & https://wenmr.science.uu.nl/auth/register/, respectively

which relies on residue-to-residue alignment. It obtains an optimal superposition of the two aligned structures and returns a TM-score. This score ranges from [0,1], 1 being a perfect match between structures. This scale, contrarily to RMSD, is insensitive to the local modelling error (Zhang and Skolnick 2005). A score of >0.5 indicates a model of correct topology while <0.17 means a random similarity.

It is worth noting that this step can be fully skipped if the user provides, instead of the amino acid sequence, the whole protein structure in PDB format. More information about this is present in section 3.4.

3.2.3. Predicting the interface residues

With CPORT (de Vries and Bonvin 2011), we predicted the interface residues, both active and passive, from the three-dimensional structure of the proteins. CPORT is an algorithm that uses six interface prediction methods into a consensus predictor, that is PIER (Kufareva et al. 2007), ProMate (Neuvirth, Raz, and Schreiber 2004a), cons-PPISP (Chen and Zhou 2005a), SPPIDER (Porollo and Meller 2007), and PINUP (Liang et al. 2006b). The active and passive residues predicted by this tool can be further used to predict the docking of macromolecular complexes in HADDOCK (van Zundert et al. 2016).

Once again, it is worth noting that this step can be skipped in its entirety. If the user provides a file where the active and passive residues list for both Reference and Interactor proteins are specified, HADDOCK will use these as the interfacing residues for its prediction. Further explanation in section 3.4.

3.2.4. Docking of the proteins

The docking of the Reference proteins with their interacting partners is obtained with HADDOCK (van Zundert et al. 2016). HADDOCK is an information-driven flexible docking approach that accurately models the 3D structure of macromolecular complexes. It relies on the integration of protein interaction data obtained with a diversity of biochemical and biophysical bioinformatics methods, such as Nuclear Magnetic Resonance (NMR) spectroscopy, mutagenesis experiments, X-ray scattering and cryo-electron microscopy. The complex clusters resulting from all the possible predicted dockings are scored based on a statistic called the z-score. The z-score represents how many standard deviations the HADDOCK score of a given cluster is separated from the mean of all obtained clusters, meaning that the lower the z-score, the better the predictions are.

From all docking structures resulting from the HADDOCK run, all that have a negative z-score are selected to continue with the final segment of the analysis.

3.2.5. Obtaining the interface's properties

Interface properties of the PPI are finally obtained with PDBePISA (Krissinel and Henrick 2007). PDBePISA is an interactive tool for the exploration of macromolecular interfaces which interactively calculates results for the submitted structures, including chemical and structural properties of macromolecular surfaces and interfaces, as well as probable quaternary structures, their structural and chemical properties and probable dissociation pattern.

Among the obtained interface properties for all clusters from HADDOCK, the pipeline additionally selects the one with the highest number of interacting residues.

3.3. The configuration file

The user might have experimental or even theorical data available which could justify skipping some steps of the pipeline and use a certain type of file in place of another. In P_2I_2P, for instance, the user can either provide as input a FASTA with the protein sequence (s) or a PDB with the structure.

In order to control the flow of these steps that can be altered, I implemented a simple text file, called “user.config” (Figure 2), where the user specifies several configuration variables which are internally used by the pipeline. Each variable in this file belongs to one of five different sets of variables, some of which regulate how each individual program runs. Additionally, some variables are optional, that is, they can be omitted from the file. In this case, the pipeline will assume default values and proceed as usual.

In this section, I will explain in more detail what is the function of each of these variables.

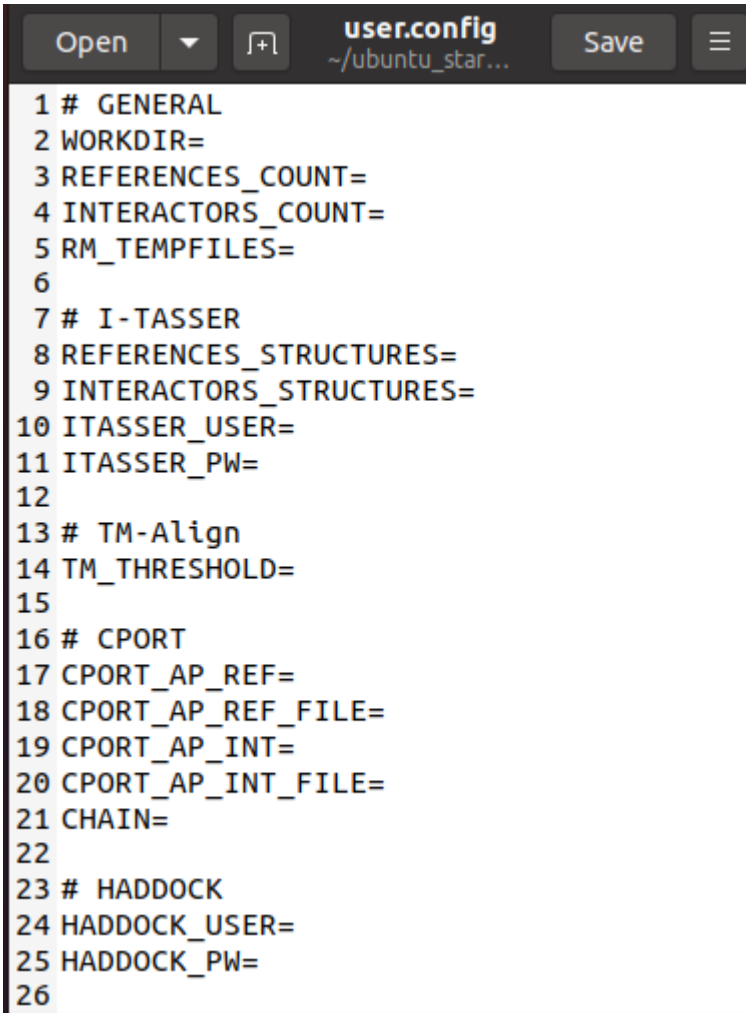

Figure 2 – Example of the configuration file. This is a simple text file where in each line a variable is declared or commented (starting with a “#”). This file must be named “user.config” and be placed inside the user’s working directory.

3.3.1. General variables

These are variables that provide general information, such as the working directory where the docker image will run and if the user wants to keep the temporary files that are generated during the execution of the pipeline.

- *WORKDIR* – denotes the absolute path to the directory where the pipeline will run with the input files provided by the user, and where all the results will be generated.
- *REFERENCES_COUNT* (optional) – the number of Reference proteins for which the pipeline will run the analysis (one or two). If you don't provide this value, the number of protein sequences will be extracted from the provided FASTA file.
- *INTERACTORS_COUNT* (optional) – the number of Interactor proteins for which the pipeline will run. If you don't provide this value, the number of protein sequences will be extracted from the provided FASTA file.
- *RM_TEMPFILES* (optional) – the option to choose whether to remove, at the end of the run, the temporary files generated or keep them (0 – no; 1 – yes).

3.3.2. I-TASSER variables

These variables control the course of the pipeline during the structure prediction step, using the I-TASSER tool.

- *REFERENCES_STRUCTURES* – determines which 3D model is to be used for the Reference proteins. It can take three different values:
  - Option "1" – uses, for each reference protein, the I-TASSER model with the highest C-score (that is, model 1).
  - Option "2" – compares all models obtained with I-TASSER with a partial or complete 3D structure, given by the user, and chooses the model with the highest structural similarity (using TM-align). This option requires that the user provides a PDB file with the partial/complete structure, in a folder named "References_Structures" inside *WORKDIR*.
  - Option "3" - skips I-TASSER altogether and uses as the final model a structure file provided by the user. This option requires that the user provides a PDB file, per Reference, in a folder named "References_Structures" inside *WORKDIR*.
- *INTERACTORS_STRUCTURES* – determines which 3D model is to be used for the Interactor proteins. It can take two different values:
  - Option "1" – chooses for each Interactor protein, the model from I-TASSER with the highest C-score (that it, the first model).
  - Option "2" – skips I-TASSER and uses as the final model a protein structure file provided by the user. This option requires that the user provides a PDB file, per Interactor, in a folder named "Interactors_Structures" inside *WORKDIR*.
- *I-TASSER_USER* – where the user should specify his I-TASSER account username (e-mail).
- *I-TASSER_PW* - where the user should specify his I-TASSER account password.

3.3.3. TM-align variables

- *TM_THRESHOLD* – serves to specify the value that the user wants to use as a threshold for the structural similarity between Reference proteins. This is because the user might want to analyse interactions with two Reference proteins that have a minimum structural similarity between them. The variable accepts a value between 0 and 1, but if the user does not fill it, the pipeline assumes a value of 0,5.

3.3.4. CPORT variables

These variables control the course of the pipeline in the interfacing residues prediction step, with the CPORT tool.

- *CPORT_AP_REF* – where the user specifies, for the Reference proteins, whether they want to provide a file with the active and passive residues for HADDOCK or not. If a file is provided, CPORT will be skipped for the References. If this field is not filled, the pipeline assumes it is necessary to predict the residues, therefore executing CPORT. It can take a value of 0 (zero) for “no” or 1 (one) for “yes”.
- *CPORT_AP_REF_FILE* – to specify the absolute path to the file which contains both active and passive residues for the Reference proteins. This field is mandatory if in the previous variable the user has selected option 1, otherwise no value should be declared. The file in question should only contain four lines, with the active and passive residues for the first Reference protein (lines 1 and 2) and after that, for the second one (lines 3 and 4).
- *CPORT_AP_INT* – where the user specifies, for the Interactor proteins, whether they want to provide a file with the active and passive residues for HADDOCK or not. If a file is provided, CPORT will be skipped for the Interactors. If this field is not filled, the pipeline assumes it is necessary to predict the residues, therefore executing CPORT. It can take a value of 0 (zero) for “no” or 1 (one) for “yes”.
- *CPORT_AP_INT_FILE* – to specify the absolute path to the file which contains both active and passive residues for the Interactor proteins. This field is mandatory if in the previous variable the user has selected option 1, otherwise no value should be declared. This file should only contain two lines with the active residues in line 1 and passive in line 2. In this way, the same active and passive residues are assumed for all interactor proteins.
- *CHAIN* – to specify which chain CPORT will use for prediction. If this field is not filled, the pipeline will use chain A as default.

3.3.5. HADDOCK variables

These variables provide information to the docking step of the pipeline, using the HADDOCK tool.

- *HADDOCK_USER* – where the user should specify his HADDOCK account username (e-mail).
- *HADDOCK_PW* – where the user should specify his HADDOCK account password.

3.3.6. Security Disclaimer

All account credentials provided in this file are only used inside the running docker image with the strict purpose of submitting the data files to the I-TASSER and HADDOCK web servers. It is also worth noting that this will enable both servers to send e-mail messages notifying the status of the data submitted and keep track of the overall analysis progress.

3.4. Customization of the analysis

Due to the set of variable options that the pipeline provides, setting up a run that matches the user's research aim is very straightforward. In this section, I will present the different available runs that the user will be able to execute with this image and a brief hands-on tutorial on how to replicate each one of them.

3.4.1. The "complete" run

In this execution mode, the user will obtain every single data object that there is to predict, meaning that, at every step of the pipeline a submission will be made to each web server (I-TASSER, CPORT, HADDOCK and PDBePISA) and the results will be afterwards obtained when ready, in order to move on to the next phase of the workflow. The biological data acquired in the end of this run will be the three-dimensional structure of the proteins (for both Reference and Interactor sets), interacting active and passive residues, protein docking and interface information.

Option settings:

Primarily, the user should set the value of *REFERENCES_STRUCTURES* to equal "1" and *INTERACTORS_STRUCTURES* to "1" on his *user.config* file (Figure 2). This will tell the pipeline that it is his intention to obtain the 3D structure of the Reference and Interactor proteins with I-TASSER and ultimately continue with the predicted model that possesses a higher confidence score (C-score), for each protein.

After that, it is mandatory that the user places a multi-FASTA file (with filename extension or not), for both Reference and Interactor proteins, with the amino acid sequences of each one of them in the directories "/Your_Workdir/References" and "/Your_Workdir/Interactors", respectively (Figure 3), where "Your_Workdir" holds the absolute path to the directory specified in the variable *WORKDIR* of the *user.config*.

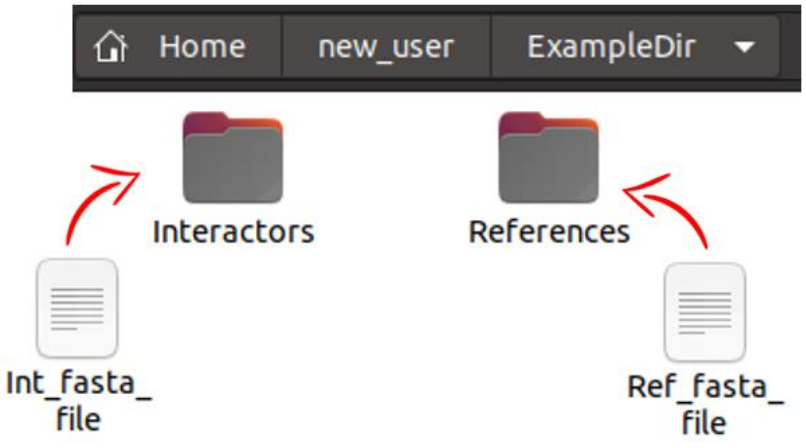

Figure 3 - File location schematic for the FASTA sequence files. Each file has multiple amino acid sequences for each Reference and Interactor proteins.

3.4.2. Complete run + TM-align – Choose the most similar model

This mode is in all ways equals to the previous one, except for the choosing criteria of the protein structures for the References. Whereas the previous method proceeds with the models that scored the highest C-score, this one allows the user to submit an additional structure (which can be partial, or complete) which will be compared with all models obtained from I-TASSER and the most similar one will be selected.

Option settings:

Firstly, the user should set the value of *REFERENCES_STRUCTURES* to equal "2" on his *user.config* file (Figure 2). This will tell the pipeline that it is his intention to obtain the 3D structure of the Reference proteins with I-TASSER but compare all models with the user submitted structure and ultimately proceed with the one possessing the closest structural similarity, that is, the highest TM-score.

Then, it is mandatory that the user places a multi-FASTA file for both Reference and Interactor proteins, with the amino acid sequences of each of them in the directories "/Your_Workdir/References" and "/Your_Workdir/Interactors", respectively (Figure 3), where "Your_Workdir" holds the absolute path to the directory specified in the variable *WORKDIR* of the *user.config*.

Additionally, the user should place his partial or complete structure file in a folder called "/Your_Workdir/References_Structures", as shown in Figure 4.

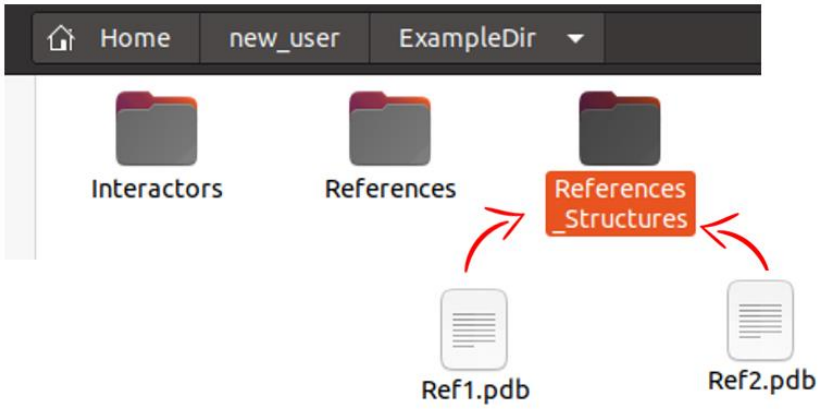

Figure 4 - File location schematic for the PDB structure files to be compared with the References with TM-align.

3.4.3. Skip I-TASSER – Use the given 3D structures

This approach allows the user to provide his own protein structures by skipping the prediction step with I-TASSER. The running of this step can be skipped for the References only, Interactors only or even for both types of proteins.

Option settings:

Depending on which kind of protein the user wants I-TASSER to be skipped on, he should set the following values on *user.config* accordingly. Setting *REFERENCES_STRUCTURES* to "3" will tell the pipeline to skip I-TASSER for the References and use the models provided and, similarly, setting *INTERACTORS_STRUCTURES* to "2" will instruct the same outcome on the Interactor proteins. Therefore, the user does not have to provide FASTA files. Instead, it is mandatory that in the *user.config* the user specifies the number of Reference and Interactor proteins that the analysis will run with. For this, the respective values should be put in *REFERENCES_COUNT* and *INTERACTORS_COUNT*. Additionally, it is necessary that the user places a PDB structure file for each Reference protein inside "/Your_Workdir/References_Structures" and similarly, for each Interactor protein, inside "/Your_Workdir/Interactors_Structures", where "Your_Workdir" holds the absolute path to the directory specified in the variable *WORKDIR* of the *user.config* (as shown in Figure 5).

It is recommended that the of the References' structure files are named "Ref1" and "Ref2" although they can be set to anything, as long as the file which comes first in alphabetical order corresponds to Reference number 1. The Interactors' files can have any name.

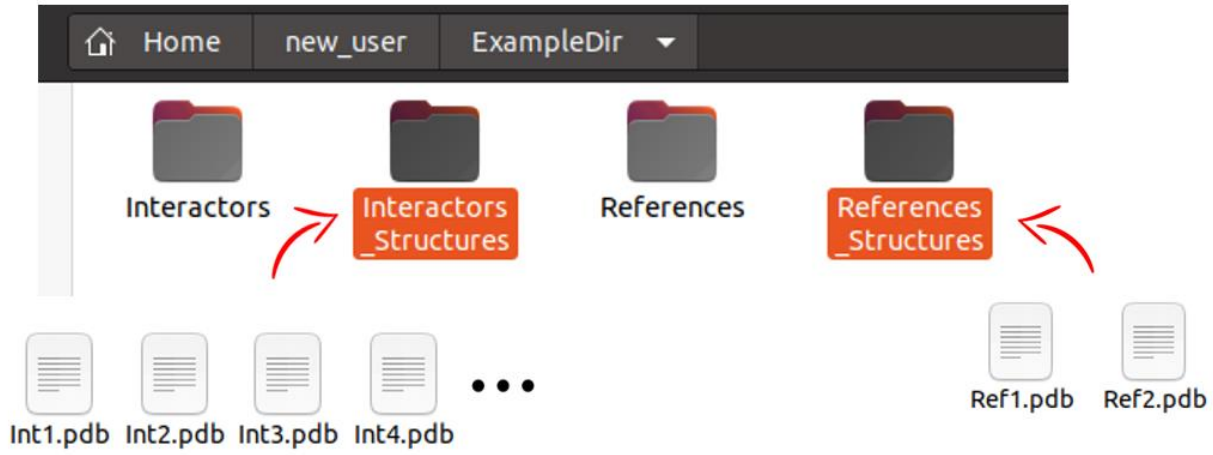

Figure 5 - Structure files location scheme. Each PDB file should be put inside its respective folder, for each protein type, in the working directory.

3.4.4. Skip CPORT – Use the given active and passive residues

This setting allows the user to provide his own residue files by skipping the prediction step with CPORT. The running of this step can be skipped for the References only, Interactors only or even for both types of proteins.

Option settings:

Depending on which kind of protein the user wants CPORT to be skipped on, he should set the following values on *user.config* accordingly. Setting *CPORT_AP_REF* to "1" will tell the pipeline to skip CPORT for the References and use as active and passive residues the ones provided in a simple text file, and, similarly, setting *CPORT_AP_INT* to "1" will instruct the same outcome on the Interactor proteins. The file for the Interactor proteins' residues should have two lines only, where in the first line are the active residues and in the second the passive ones, whereas the file for the Reference proteins' residues should have four lines (if you have two Reference proteins), where the first two lines correspond to the residues for the first Reference and the other two for the other protein, as shown in Figure 6.

In the case of the Interactors, the same active and passive sites are assumed for all interactor proteins. If the user wants to set different residue sites for each interactor protein, he will have to run each one individually.

Additionally, the user should specify the absolute paths to the residues' files, so the pipeline knows where this file is in order to use it. This should be done by filling the paths in *CPORT_AP_REF_FILE* and *CPORT_AP_INT_FILE*, for the References and Interactors, respectively.

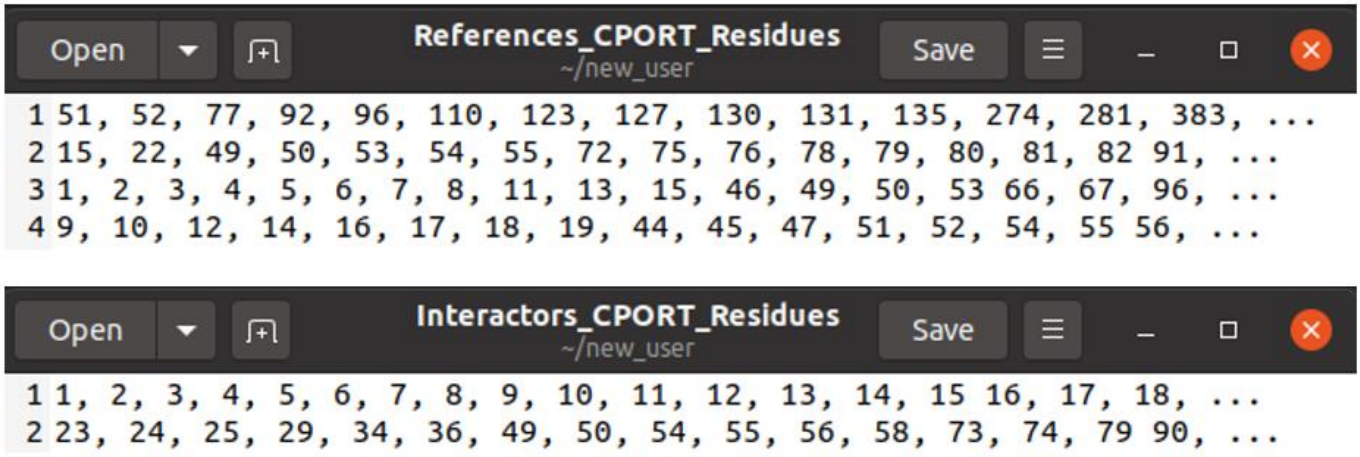

Figure 6 - Example of both residues file for the References and the Interactors proteins.

3.4.5. Skip I-TASSER and CPORT

This approach allows the user not only to provide his own protein structures, but also the residues files. This allows the analysis to skip both I-TASSER and CPORT and it can be skipped for the References only, Interactors only or even for both types of proteins.

Option settings:

Equally to modes 3.4.3 and 3.4.4, the user should set the values in the *user.config* accordingly to the pretended outcome.

To skip I-TASSER the user should set *REFERENCES_STRUCTURES* to “3” and *INTERACTORS_STRUCTURES* to “2”, then he should place the structure files in their respective folders (see Figure 5). Additionally, he should specify in *REFERENCES_COUNT* and *INTERACTORS_COUNT* how many sequences are there to analyse.

To skip CPORT the user should set *CPORT_AP_REF* and *CPORT_AP_INT* to “1”, then he should provide the absolute path to each residues file in *CPORT_AP_REF_FILE* and *CPORT_AP_INT_FILE* (see Figure 6).

3.5. Final results and their location

Regardless of which type of run the user sets, the final results are always saved in a folder named “Results” under the *WORKDIR* directory. This folder contains a subfolder for each pair Reference-Interactor, which in turn will contain two subfolders, “results” and “selection”. The “results” folder has for each predicted docking structure an XML file with a summary of the interface information generated using PDBePISA. Whereas the “selection” folder contains the selected pair of docking structures with the highest number of interface residues in the first structure. If there are two alternatives showing the same number of interface residues, the file selected is the one with the highest number for the second structure. Additionally, this folder contains a file called “structure1” where the interacting residues of the selected structure are shown. In Figure 7, a directory tree scheme showing the folders and files inside the “Results” folder is presented.

To see the results in the XML file it is recommended that a spreadsheet software like Excel or LibreOffice Calc is used.

```
Your_Workdir/
├── Reference_Structures/
├── Interactor_Structures/
└── Results/
    ├── XML_Extract_IntName_R1/
    └── XML_Extract_IntName_R2/
        ├── results/
        │   └── IntName_R2/
        │       ├── interface-0-XX-XX-XX-IntName_R2_cluster1_3.xml/
        │       │   └── summary
        │       └── interface-0-XX-XX-XX-IntName_R2_cluster4_2.xml/
        │           └── summary
        ├── selection/
        │   └── IntName_R2/
        │       └── interface-0-XX-XX-XX-IntName_R2_cluster4_2.xml/
        │           └── summary
        └── structure1
```

Figure 7 – User directory tree scheme showing the structure of the “Results” folder. Lines that end with a “/” represent directories while the others are data files.

Chapter 4

Testing the Docker image

4.1. Introduction

The hitherto described pipeline can be run for any experiments with the aim of inferring PPI interfaces and their properties, but here I am testing P2I2P using a protocol similar to that of (Rocha et al. 2019) for ATXN1. In this case, the References are wild-type ATXN1 and expanded ATXN1. The interactors were chosen in terms of size. Three interactor proteins – a small (EIF1B), medium (CHRNA7) and large-sized (DHX37) – were chosen to run each analysis, in order to compare with the data obtained in (Rocha et al. 2019), to calculate and compare runtimes for each prediction step, and assess if the protein sizes have a great influence on the running times and efficiency of the developed pipeline. The protein structures for these analyses were obtained from the results of the paper mentioned above, therefore there was no need to use I-TASSER, so this step was skipped for all runs. Moreover, a fourth analysis was conducted with a protein, also from (Rocha et al. 2019), of medium size, where instead its structure was downloaded from AlphaFold's database[35].

To select a protein from this database, the structural similarity between all 71 proteins, reported to interact with ATXN1 from (Rocha et al. 2019), and the corresponding 3D structures in AlphaFold DB was calculated with TM-align. The selection criteria for the fourth analysis was to choose a protein of medium size, that presented a TM-score of at least 0,8 (80% structural similarity).

A final analysis was run with the three proteins described above, but instead their 3D structures were obtained from AlphaFold DB in order to compare runtimes with the previous instances of the pipeline.

[35] https://alphafold.ebi.ac.uk

4.2. Proteins from (Rocha et al. 2019)

In order to assess the pipeline's prediction power, three proteins that interact with ATXN1 both in its wild type and expanded forms were chosen from the list of proteins that interact with both isoforms from (Rocha et al. 2019):

- **Small sized protein** – EIF1B (Uniport ID: O60739), Eukaryotic translation initiation factor 1B, length: 113 amino acids (aa).
- **Medium sized protein** – CHRNA7 (Uniport ID: P36544), Neuronal acetylcholine receptor subunit alpha-7, length: 502 aa.
- **Large sized protein** – DHX37 (Uniport ID: Q8IY37), Probable ATP-dependent RNA helicase 37, length: 1157 aa.

Since the protein structures were already available from previously published work, the prediction step with I-TASSER was skipped in its entirety, which meant that the analysis "started" from the interface prediction stage, using CPORT's algorithm. The *user.config* file's settings for the runs are shown in Figure 8Figure 8.

The runtime of each prediction step of the pipeline was calculated for each of the three analyses performed and the resulting values are presented in Table 1.

Open

user.config
~/paula/Docker/Pipeline_stable

```
# GENERAL
WORKDIR=/home/paula/Docker/Pipeline_stable/ATXN1_CHRNA7
REFERENCES_COUNT=
INTERACTORS_COUNT=
RM_TEMPFILES=

# I-TASSER
REFERENCES_STRUCTURES=3
INTERACTORS_STRUCTURES=1
ITASSER_USER=paula.gouveia@ibmc.up.pt
ITASSER_PW=******

# TM-Align
TM_THRESHOLD=0

# CPORT
CPORT_AP_REF=0
CPORT_AP_REF_FILE=
CPORT_AP_INT=0
CHAIN=A

# HADDOCK
HADDOCK_USER=paula.gouveia@ibmc.up.pt
HADDOCK_PW=******
```

Figure 8 – Settings inside user.config that weres used for each different run. The only variable that changed accordingly was WORKDIR. Some variables have empty values, such as RM_TEMPFILES, because the pipeline has default values that it sets for the cases where no option is given.

Table 1 - Runtimes for each different step of the pipeline, in seconds. Because the pre-analyses steps are common to all proteins and their runtime does not depend on the size of the protein nor on server queues, the duration time was only calculated once for each of the four preliminary steps. The numbers within brackets are approximate human-readable values which allow for a legible reading of the final runtimes.

Protein	Pipeline step		Runtime
All	Test servers		7.824
	Check config variables		0.584
	Check directories		0.573
	Docker images pull		40.423
EIF1B (113 aa)	CPORT	References	8200.7187 (2h 16min)
		Interactors	365.497 (6 min)
	HADDOCK		72638.276 (20h)
	PISA		1146.490 (19 min)
	XML extract	Pair 1	25.943
		Pair 2	63.236 (1 min)
CHRNA7 (502 aa)	CPORT	References	7214.592 (2h)
		Interactors	1446.512 (24 min)
	HADDOCK		278736.493 (3 days)
	PISA		1545.999 (26 min)
	XML extract	Pair 1	50.873
		Pair 2	79.179 (1 min)
DHX37 (1157 aa)	CPORT	References	7226.929 (2h)
		Interactors	8120.616 (2h 15min)
	HADDOCK		113954.668 (1 day)
	PISA		702.005 (12 min)
	XML extract	Pair 1	54.072
		Pair 2	42.804
ILVBL (632 aa)	CPORT	References	7575.406 (2h 6min)
		Interactors	1506.284 (25 min)
	HADDOCK		228370.263 (2 days)
	PISA		1279.000 (21 min)
	XML extract	Pair 1	70.162 (1 min)
		Pair 2	43.842

The runtimes for the four pre-analysis steps were, as expected, very brief in the order of seconds, and do not even run for a whole minute. Despite this, the runtime for these can vary depending on the quality of the user's internet connection, since the *Test servers* step requires a ping is sent to each webserver to inspect its integrity, and the *Docker images pull* step requires, as the name implies, the pulling of the Docker images from a registry to the user's local system.

Runtimes for the prediction of interacting residues with CPORT for both Reference proteins is shown to be consistent throughout all individual runs, lasting around 2 hours, which was expected since the sequences are the same every time. Moreover, for the Interactors the runtimes were increasingly longer from the smaller protein to the larger one, which was likewise expected.

Protein docking times were naturally the longest due to the complicated aspect of this prediction process. It was expected that the bigger the protein interacting with our reference sequences the longer it would take for HADDOCK to calculate the docking structures and their scores, which can vary between half an hour and several days (van Zundert et al. 2016). As can be seen in Table 1, HADDOCK for the small sized protein (EIF1B) ran for 20h whereas for the other proteins it ran for more than one day. Surprisingly, and contrarily to what was expected at first, the HADDOCK time for the bigger protein (DHX37) was lower than for the medium sized protein (CHRNA7), as well as for the protein obtained from AF (ILVBL), which lasted approximately for 1 day, 3 days and 2 days, respectively. This could be explained by the fact that the interface area of the interaction between CHRNA7 and both References is bigger and consists of more residues than the interfaces of DHX37 interactions, as can be seen in Table 2.

Interface information was obtained by PISA in less than half an hour for each, in general. The obtaining of PISA's results in XML format, and the extraction of readable data from the generated files took the longest for the medium-sized protein, 26 minutes and the shortest for the larger protein, 12 minutes.

Table 2 – Number of interfacing residues and interface area of PPI between the three proteins chosen and wt- and exp-ATXN1. From all the docking solutions obtained with a negative Z-score, the ones with the most residues were selected.

Protein	**Number of interface residues**		**Interface Area (Å)**	
	wt-ATXN1	exp-ATXN1	wt-ATXN1	exp-ATXN1
EIF1B	79	73	2230.61	1873.69
CHRNA7	98	109	3629.56	3547.03
DHX37	67	78	1709.47	2073.51

4.3. AlphaFold Protein Structure Database

4.3.1. Introduction

Recently, on July 15th, London-based DeepMind and The European Bioinformatics Institute (EMBL-EBI)[36] released to the public audience the AlphaFold 2 software (Jumper et al. 2021), fruit of a collaboration between these two renowned institutions. The latest version of this AI-based protein prediction method showed an astounding performance at the 14th Critical Assessment of protein Structure Prediction (CASP14)[37], outperforming other participating methods, and producing models for two-thirds of the targets with global distance test (GDT) scores above 90 (out of 100), with a median of 92.4 GDT across all targets, which ultimately shows its ability to predict protein structures to a level of accuracy comparable to that of laboratory experiments[38] which have been the working playground of many researches for the past decade in the field of predicting protein interactions. Moreover, I-TASSER's group (Zhang Server) performance in this assessment granted them the 9th position overall (but scoring 1st place within server-only groups), although being the top predictor until 2020. It is additionally worthy of noting that another developed similar system, RoseTTaFold which was inspired by AlphaFold and is quickly gaining popularity within the scientific community, performed almost as well as AlphaFold 2 in CASP14, and significantly better than other entries. By providing these tools in an open-source manner, researchers can employ their knowledge and resources oriented to building on advances to improve the state-of-the-art software and assist on the progress of structural bioinformatics.

A few key advances were brought by AlphaFold 2, namely how the artificial network makes use of information regarding proteins evolutionarily related to the prediction targets, and how the predicted structures of one part of a protein can affect how the network handles sequences from other parts of the protein. Moreover, AlphaFold 2 was also simplified so that the open-source version is 16 times faster and, depending on the protein size, it is able to generate structures in just minutes or hours, a speed comparable to that of RoseTTaFold (Jumper et al. 2021).

With the release of AlphaFold 2's source code, this cooperation additionally led to the launching of AlphaFold Protein Structure Database (AlphaFold DB)[39], a database of high-quality predictions of the complete human proteome and 20 other research significant organisms, totalling over 365.000 proteins, obtained with AlphaFold 2. With these protein structures ready to be used, researchers can save precious time they would otherwise spend predicting a structure and access its quality. By sharing these protein structures, DeepMind and EMBL-EBI provide the scientific community a valuable resource for the advance of proteomics and bioinformatics.

4.3.2. Structural comparison with I-TASSER predicted proteins

TM-align was used to measure the similarity between all structures obtained from that paper and the structures from AlphaFold, where the resulting values are presented in Supplementary Table 1.

From the structural comparison among all proteins (including both ATXN1 isoforms), we can see that just 29% (28.571) have a TM-score between 0,5 and 1, which corresponds to a higher structural similarity and about 51% (51.351) scored a TM-score between 0 and 0,3 suggesting that these protein structures were randomly chosen from the PDB and are unrelated.

[36] https://www.ebi.ac.uk/
[37] https://predictioncenter.org/casp14/index.cgi
[38] https://predictioncenter.org/casp14/doc/CASP14_press_release.html
[39] https://alphafold.ebi.ac.uk/

4.3.3. Runtimes with ILVBL

The protein structure for ILVBL (Uniprot ID: A1L0T0)[40], 2-hydroxyacyl-CoA lyase 2, was obtained from AlphaFold DB. The structural similarity of this structure and the one available from the results of (Rocha et al. 2019) was determined to be 0.80527.

Overall runtimes, at first glance, were not that much longer or shorter than the ones observed for the other proteins

[40] https://alphafold.ebi.ac.uk/entry/A1L0T0

Chapter 5

Conclusions

Despite being highly accurate in their final predictions, *in vivo* and *in vitro* methods are costly and slow. Moreover, the final data set is highly prone to noise, false positives, and false negatives, which ultimately decreases the confidence in inferred interactions. With *in silico* methods researchers can start from a set of high quality PPIs. To analyse these interactions, a structural prediction of its interface is needed. While the whole process of submitting data, waiting for results, downloading, and organizing the results is time consuming, this developed pipeline automates the whole process, allowing for easiness of use and time management.

While the pipeline was built to perform all the steps, from 3D structure and putative active and passive residue prediction to the docking of the interacting molecules, it has some level of flexibility since it is prepared to receive already existing protein structures and interface residues, which saves some time at the beginning of the analysis. Furthermore, this feature is very useful in the scenario where a power surge or other problems result in the aborting of the pipeline while it is running. By "recycling" the data obtained until a certain point, it saves time in the next iteration of the pipeline.

By uniting all tools and automating a protocol to infer PPI interfaces, the time spent submitting data and downloading it to/from webservers is significantly reduced. However, it does not reduce the overall waiting time for the results to be ready, given that this period only depends on the server's resources and job queue.

Additionally, by adapting each step and software tool to the Docker framework, we surpass the need of installing programs and deal with issues such as missing dependencies, hard-to-follow tutorials, and not to mention the fact that not every developer intends to share their work with the scientific community in an open source manner. Furthermore, by implementing these images from webservers, the user is notified whenever an important step of the processing is started or finishes. Namely because I-TASSER and HADDOCK require account authentication in their webservers, an e-mail is sent out each time their processing reaches a turning point, such as jobs going into the queue and when they finish running.

AlphaFold DB protein structures

Up until the developing of this project, I-TASSER was the top-notch algorithm for the prediction of 3D structures of proteins, being consistently ranked above 3rd place at every CASP since 2006[41]. In 2020's CASP14, AlphaFold 2 performed substantially better and ranked higher than Yang Zhang Lab's method, which showed promising prospects in the advance of structural bioinformatics. On July 15th, 2021, the source code of AlphaFold 2 was publicly released and later on the 22nd a database with structures predicted with the AI method, AlphaFold DB, was published.

Since AlphaFold has scored substantially higher than I-TASSER and other methods at CASP14, showing its superior predictive power, it is highly recommended the use of these structures in the structural prediction stage of this pipeline, as well as in any other experiments.

The TM-align scores obtained for both protein chains, as observed in Supplementary Tables 1 and 2, show the remarkable differences between both *in silico* approaches. Although 29% of the I-TASSER predicted structures score a TM-value above 0,5 (high similarity with AlphaFold), about half of the proteins (51%) are shown to have a low structural similarity, suggesting that I-TASSER's prediction power, for these particular proteins, falls short of an accurate result, in comparison to AlphaFold.

[41] https://www.predictioncenter.org/index.cgi

Limitations and Future work

Because of I-TASSER's inaptitude to infer the structures of proteins with lengths over 1500 aa, this was a major limitation of the methodology described in (Rocha et al. 2019). However, it does not pose a problem in the context of this pipeline due to the fact that it delivers an alternative way of obtaining a protein structure. Without recurring to Zhang's predictive method, the user can provide an already existing protein structure which he either fetched from a web repository or obtained from wet-lab experiments.

Moreover, the reliable predictions of AlphaFold 2 and the availableness of its related database, comes to add to this pipeline a whole new level of scientific relevance, since now these high quality protein structures can be directly used with the pipeline in detriment of running I-TASSER's step, ultimately improving its runtime, and increasing confidence in the interface predictions.

It is worth noting that while the above issue was easily surpassed, the other limitation of the methodology remains, which is the fact that this method only considers binary interactions.

Furthermore, regarding the prediction of the interfacing residues, if the user wants to skip this step for the interactor proteins, he must provide a file with two lines regarding the active and passive residues for the proteins. In this case, the same active and passive sites are assumed for all interactor proteins. Meaning that if the user wants to set different residue sites for each interactor protein, he will have to run each one individually with a different residue file each time. While this is limitative in terms of time efficiency, this was intentionally kept due to the fact that in the literature it is rare to obtain information regarding protein interface residues.

As the relevance of AlphaFold in the field of proteomics increases, advances in research will surely unravel. Keeping this in mind, and since the source code for AlphaFold v2.0 is openly available, we came up with the objective of implementing this extremely useful AI-method as a Docker image in the near future. Furthermore, to expand the pipeline's flexibility we will seek to augment the structural prediction step, by giving the user the third option of running the analysis with protein structures present in AlphaFold DB, instead of predicting them with I-TASSER.

An additional feature we aim to add to the pipeline is the possibility of providing the PDB docking file(s), which would allow for the skipping of HADDOCK and consequently starting the analysis on the last step which is the obtaining of information about the interaction interface.

Although at the time of writing of this thesis, the Docker image of this computational approach is not yet publicly available, we intend to release it in the near future.

Bibliography

Alberts, Bruce, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts, and Peter Walter. 2002. *Molecular Biology of The Cell*.

Anand, G. S., D. Law, J. G. Mandell, A. N. Snead, I. Tsigelny, S. S. Taylor, L. F. Ten Eyck, and E. A. Komives. 2003. 'Identification of the protein kinase A regulatory RIalpha-catalytic subunit interface by amide H/2H exchange and protein docking', *Proc Natl Acad Sci U S A*, 100: 13264-9.

and, Richard Bonneau, and David Baker. 2001. 'Ab Initio Protein Structure Prediction: Progress and Prospects', *Annual Review of Biophysics and Biomolecular Structure*, 30: 173-89.

Attrill, Helen, Kathleen Falls, Joshua L. Goodman, Gillian H. Millburn, Giulia Antonazzo, Alix J. Rey, and Steven J. Marygold. 2016. 'FlyBase: establishing a Gene Group resource forDrosophila melanogaster', *Nucleic acids research*, 44: D786-D92.

Bahadur, R. P., P. Chakrabarti, F. Rodier, and J. Janin. 2004. 'A dissection of specific and non-specific protein-protein interfaces', *J Mol Biol*, 336: 943-55.

Baker, D., and A. Sali. 2001. 'Protein structure prediction and structural genomics', *Science*, 294: 93-6.

Bashari Rad, Babak, Harrison Bhatti, and Mohammad Ahmadi. 2017. 'An Introduction to Docker and Analysis of its Performance', *IJCSNS International Journal of Computer Science and Network Security*, 173: 8.

Battey, James, Jürgen Kopp, Lorenza Bordoli, Randy Read, Neil Clarke, and Torsten Schwede. 2007. 'Automated server predictions in CASP7', *Proteins*, 69 Suppl 8: 68-82.

Baxevanis, A. D., and Francis Ouellette. 2001. *Bioinformatics: A practical guide to the analysis of genes and proteins*.

Berggård, Tord, Sara Linse, and Peter James. 2007. 'Methods for the detection and analysis of protein–protein interactions', *Proteomics*, 7: 2833-42.

Bietz, S., and M. Rarey. 2016. 'SIENA: Efficient Compilation of Selective Protein Binding Site Ensembles', *J Chem Inf Model*, 56: 248-59.

Bietz, Stefan, Sascha Urbaczek, Benjamin Schulz, and Matthias Rarey. 2014. 'Protoss: a holistic approach to predict tautomers and protonation states in protein-ligand complexes', *Journal of Cheminformatics*, 6: 12.

Boettiger, Carl. 2015. 'An introduction to Docker for reproducible research', *SIGOPS Oper. Syst. Rev.*, 49: 71–79.

Bogan, Andrew A., and Kurt S. Thorn. 1998. 'Anatomy of hot spots in protein interfaces11Edited by J. Wells', *Journal of Molecular Biology*, 280: 1-9.

Bonvin, A. M., R. Boelens, and R. Kaptein. 2005. 'NMR analysis of protein interactions', *Curr Opin Chem Biol*, 9: 501-8.

Bowie, J U, and D Eisenberg. 1994. 'An evolutionary approach to folding small alpha-helical proteins that uses sequence information and an empirical guiding fitness function', *Proceedings of the National Academy of Sciences*, 91: 4436-40.

Bowie, James U., Roland Lüthy, and David Eisenberg. 1991. 'A Method to Identify Protein Sequences That Fold into a Known Three-Dimensional Structure', *Science*, 253: 164-70.

Bradshaw, R. T., B. H. Patel, E. W. Tate, R. J. Leatherbarrow, and I. R. Gould. 2011. 'Comparing experimental and computational alanine scanning techniques for probing a prototypical protein-protein interaction', *Protein Eng Des Sel*, 24: 197-207.

Braun, P., and A. C. Gingras. 2012. 'History of protein-protein interactions: from egg-white to complex networks', *Proteomics*, 12: 1478-98.

Brooks, Bernard R., Robert E. Bruccoleri, Barry D. Olafson, David J. States, S. Swaminathan, and Martin Karplus. 1983. 'CHARMM: A program for macromolecular energy, minimization, and dynamics calculations', *Journal of Computational Chemistry*, 4: 187-217.

Chatr-Aryamontri, A., R. Oughtred, L. Boucher, J. Rust, C. Chang, N. K. Kolas, L. O'Donnell, S. Oster, C. Theesfeld, A. Sellam, C. Stark, B. J. Breitkreutz, K. Dolinski, and M. Tyers. 2017. 'The BioGRID interaction database: 2017 update', *Nucleic Acids Res*, 45: D369-d79.

Chen, H., and H. X. Zhou. 2005a. 'Prediction of interface residues in protein-protein complexes by a consensus neural network method: test against NMR data', *Proteins*, 61: 21-35.

Chen, Huiling, and Huan-Xiang Zhou. 2005b. 'Prediction of interface residues in protein–protein complexes by a consensus neural network method: Test against NMR data', *Proteins: Structure, Function, and Bioinformatics*, 61: 21-35.

Cohen, Mati, Dana Reichmann, Hani Neuvirth, and Gideon Schreiber. 2008. 'Similar chemistry, but different bond preferences in inter versus intra-protein interactions', *Proteins: Structure, Function, and Bioinformatics*, 72: 741-53.

Csaba, Gergely, Fabian Birzele, and Ralf Zimmer. 2008. 'Protein structure alignment considering phenotypic plasticity', *Bioinformatics*, 24: i98-i104.

Dai, Wentao, Aiping Wu, Liangxiao Ma, Yi-Xue Li, Taijiao Jiang, and Yuan-Yuan Li. 2016. 'A novel index of protein-protein interface propensity improves interface residue recognition', *BMC Systems Biology*, 10: 112.

Daniels, Noah M., Shilpa Nadimpalli, and Lenore J. Cowen. 2012. 'Formatt: Correcting protein multiple structural alignments by incorporating sequence alignment', *BMC Bioinformatics*, 13: 259.

Das, R., B. Qian, S. Raman, R. Vernon, J. Thompson, P. Bradley, S. Khare, M. D. Tyka, D. Bhat, D. Chivian, D. E. Kim, W. H. Sheffler, L. Malmström, A. M. Wollacott, C. Wang, I. Andre, and D. Baker. 2007. 'Structure prediction for CASP7 targets using extensive all-atom refinement with Rosetta@home', *Proteins*, 69 Suppl 8: 118-28.

De Beer, Tjaart A. P., Karel Berka, Janet M. Thornton, and Roman A. Laskowski. 2014. 'PDBsum additions', *Nucleic acids research*, 42: D292-D96.

De Las Rivas, Javier, and Celia Fontanillo. 2010. 'Protein–Protein Interactions Essentials: Key Concepts to Building and Analyzing Interactome Networks', *PLOS Computational Biology*, 6: e1000807.

de Vries, S. J., and A. M. Bonvin. 2011. 'CPORT: a consensus interface predictor and its performance in prediction-driven docking with HADDOCK', *PLoS One*, 6: e17695.

de Vries, Sjoerd J., Aalt D.J. van Dijk, and Alexandre M.J.J. Bonvin. 2006. 'WHISCY: What information does surface conservation yield? Application to data-driven docking', *Proteins: Structure, Function, and Bioinformatics*, 63: 479-89.

DeLano, W. L. 2002. 'Unraveling hot spots in binding interfaces: progress and challenges', *Curr Opin Struct Biol*, 12: 14-20.

Dhungana, S., M. B. Fessler, and K. B. Tomer. 2009. 'Epitope mapping by differential chemical modification of antigens', *Methods Mol Biol*, 524: 119-34.

Dominguez, Cyril, Rolf Boelens, and Alexandre M. J. J. Bonvin. 2003. 'HADDOCK: A Protein−Protein Docking Approach Based on Biochemical or Biophysical Information', *Journal of the American Chemical Society*, 125: 1731-37.

Dorn, Márcio, Mariel Barbachan e Silva, Luciana S. Buriol, and Luis C. Lamb. 2014. 'Three-dimensional protein structure prediction: Methods and computational strategies', *Computational Biology and Chemistry*, 53: 251-76.

Fellouse, Frederic A., Christian Wiesmann, and Sachdev S. Sidhu. 2004. 'Synthetic antibodies from a four-amino-acid code: A dominant role for tyrosine in antigen recognition', *Proceedings of the National Academy of Sciences of the United States of America*, 101: 12467-72.

Gingras, A. C., M. Gstaiger, B. Raught, and R. Aebersold. 2007. 'Analysis of protein complexes using mass spectrometry', *Nat Rev Mol Cell Biol*, 8: 645-54.

Gray, Jeffrey J., Stewart Moughon, Chu Wang, Ora Schueler-Furman, Brian Kuhlman, Carol A. Rohl, and David Baker. 2003. 'Protein–Protein Docking with Simultaneous Optimization of Rigid-body Displacement and Side-chain Conformations', *Journal of Molecular Biology*, 331: 281-99.

Holm, L., and C. Sander. 1993. 'Protein structure comparison by alignment of distance matrices', *J Mol Biol*, 233: 123-38.

Huang, H., and J. S. Bader. 2009. 'Precision and recall estimates for two-hybrid screens', *Bioinformatics*, 25: 372-8.

Hubbard, T. J., A. G. Murzin, S. E. Brenner, and C. Chothia. 1997. 'SCOP: a structural classification of proteins database', *Nucleic acids research*, 25: 236-39.

Hwang, Howook, Donald Petrey, and Barry Honig. 2016. 'A hybrid method for protein-protein interface prediction', *Protein Science*, 25: 159-65.

Hwang, Howook, Thom Vreven, Brian G. Pierce, Jui-Hung Hung, and Zhiping Weng. 2010. 'Performance of ZDOCK and ZRANK in CAPRI rounds 13-19', *Proteins: Structure, Function, and Bioinformatics*, 78: 3104-10.

Janin, J. 2005. 'Assessing predictions of protein-protein interaction: The CAPRI experiment', *Protein Science*, 14: 278-83.

Jones, S., and J. M. Thornton. 1996. 'Principles of protein-protein interactions', *Proc Natl Acad Sci U S A*, 93: 13-20.

Jorgensen, W. L., and J. Tirado-Rives. 1988. 'The OPLS [optimized potentials for liquid simulations] potential functions for proteins, energy minimizations for crystals of cyclic peptides and crambin', *J Am Chem Soc*, 110: 1657-66.

Jubb, Harry, Tom L. Blundell, and David B. Ascher. 2015. 'Flexibility and small pockets at protein–protein interfaces: New insights into druggability', *Progress in Biophysics and Molecular Biology*, 119: 2-9.

Jumper, John, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. 2021. 'Highly accurate protein structure prediction with AlphaFold', *Nature*, 596: 583-89.

Kanehisa, M. 2000. 'KEGG: Kyoto Encyclopedia of Genes and Genomes', *Nucleic acids research*, 28: 27-30.

Kanehisa, Minoru, Miho Furumichi, Yoko Sato, Mari Ishiguro-Watanabe, and Mao Tanabe. 2021. 'KEGG: integrating viruses and cellular organisms', *Nucleic acids research*, 49: D545-D51.

Keskin, O., B. Ma, and R. Nussinov. 2005. 'Hot regions in protein--protein interactions: the organization and contribution of structurally conserved hot spot residues', *J Mol Biol*, 345: 1281-94.

Kinch, L. N., W. Li, B. Monastyrskyy, A. Kryshtafovych, and N. V. Grishin. 2016. 'Evaluation of free modeling targets in CASP11 and ROLL', *Proteins*, 84 Suppl 1: 51-66.

Kinoshita, K., Y. Murakami, and H. Nakamura. 2007. 'eF-seek: prediction of the functional sites of proteins by searching for similar electrostatic potential and molecular surface shape', *Nucleic acids research*, 35: W398-W402.

Kovács, István A., Katja Luck, Kerstin Spirohn, Yang Wang, Carl Pollis, Sadie Schlabach, Wenting Bian, Dae-Kyum Kim, Nishka Kishore, Tong Hao, Michael A. Calderwood, Marc Vidal, and Albert-László Barabási. 2019. 'Network-based prediction of protein interactions', *Nature Communications*, 10: 1240.

Krissinel, Evgeny, and Kim Henrick. 2007. 'Inference of Macromolecular Assemblies from Crystalline State', *Journal of Molecular Biology*, 372: 774-97.

Kufareva, Irina, Levon Budagyan, Eugene Raush, Maxim Totrov, and Ruben Abagyan. 2007. 'PIER: Protein interface recognition for structural proteomics', *Proteins: Structure, Function, and Bioinformatics*, 67: 400-17.

Laskowski, R. A., E. G. Hutchinson, A. D. Michie, A. C. Wallace, M. L. Jones, and J. M. Thornton. 1997. 'PDBsum: a Web-based database of summaries and analyses of all PDB structures', *Trends Biochem Sci*, 22: 488-90.

Lee, Jooyoung, Peter L. Freddolino, and Yang Zhang. 2017. 'Ab Initio Protein Structure Prediction.' in Daniel J. Rigden (ed.), *From Protein Structure to Function with Bioinformatics* (Springer Netherlands: Dordrecht).

Lensink, Marc F., and Shoshana J. Wodak. 2013. 'Docking, scoring, and affinity prediction in CAPRI', *Proteins: Structure, Function, and Bioinformatics*, 81: 2082-95.

Levitt, M., and M. Gerstein. 1998. 'A unified statistical framework for sequence comparison and structure comparison', *Proceedings of the National Academy of Sciences*, 95: 5913-20.

Levy, E. D. 2010. 'A simple definition of structural regions in proteins and its use in analyzing interface

evolution', *J Mol Biol*, 403: 660-70.
Li, W., S. J. Hamill, A. M. Hemmings, G. R. Moore, R. James, and C. Kleanthous. 1998. 'Dual recognition and the role of specificity-determining residues in colicin E9 DNase-immunity protein interactions', *Biochemistry*, 37: 11771-9.
Liang, Shide, Chi Zhang, Song Liu, and Yaoqi Zhou. 2006a. 'Protein binding site prediction using an empirical scoring function', *Nucleic acids research*, 34: 3698-707.
———. 2006b. 'Protein Binding Site Prediction Using an Empirical Scoring Function', *Nucleic acids research*, 34: 3698-707.
Licata, Luana, Leonardo Briganti, Daniele Peluso, Livia Perfetto, Marta Iannuccelli, Eugenia Galeota, Francesca Sacco, Anita Palma, Aurelio Pio Nardozza, Elena Santonico, Luisa Castagnoli, and Gianni Cesareni. 2012. 'MINT, the molecular interaction database: 2012 update', *Nucleic acids research*, 40: D857-D61.
Lichtarge, O., H. R. Bourne, and F. E. Cohen. 1996. 'An evolutionary trace method defines binding surfaces common to protein families', *J Mol Biol*, 257: 342-58.
Lim, J., T. Hao, C. Shaw, A. J. Patel, G. Szabó, J. F. Rual, C. J. Fisk, N. Li, A. Smolyar, D. E. Hill, A. L. Barabási, M. Vidal, and H. Y. Zoghbi. 2006. 'A protein-protein interaction network for human inherited ataxias and disorders of Purkinje cell degeneration', *Cell*, 125: 801-14.
Lin, Tzu-Wen, Jian-Wei Wu, and Darby Tien-Hao Chang. 2013. 'Combining Phylogenetic Profiling-Based and Machine Learning-Based Techniques to Predict Functional Related Proteins', *PLoS One*, 8: e75940.
Lo Conte, L., C. Chothia, and J. Janin. 1999. 'The atomic structure of protein-protein recognition sites', *J Mol Biol*, 285: 2177-98.
López-Fernández, Hugo, Pedro Ferreira, Miguel Reboiro-Jato, Cristina P. Vieira, and Jorge Vieira. 2022. "The pegi3s Bioinformatics Docker Images Project." In, 31-40. Cham: Springer International Publishing.
Lyskov, S., and J. J. Gray. 2008. 'The RosettaDock server for local protein-protein docking', *Nucleic acids research*, 36: W233-W38.
Macalino, Stephani Joy Y., Shaherin Basith, Nina Abigail B. Clavio, Hyerim Chang, Soosung Kang, and Sun Choi. 2018. 'Evolution of In Silico Strategies for Protein-Protein Interaction Drug Discovery', *Molecules*, 23: 1963.
Martí-Renom, Marc A., Ashley C. Stuart, András Fiser, Roberto Sánchez, Francisco Melo and, and Andrej Šali. 2000. 'Comparative Protein Structure Modeling of Genes and Genomes', *Annual Review of Biophysics and Biomolecular Structure*, 29: 291-325.
Meenan, N. A., A. Sharma, S. J. Fleishman, C. J. Macdonald, B. Morel, R. Boetzel, G. R. Moore, D. Baker, and C. Kleanthous. 2010. 'The structural and energetic basis for high selectivity in a high-affinity protein-protein interaction', *Proc Natl Acad Sci U S A*, 107: 10080-5.
Memišević, Vesna, Anders Wallqvist, and Jaques Reifman. 2013. 'Reconstituting protein interaction networks using parameter-dependent domain-domain interactions', *BMC Bioinformatics*, 14: 154.
Menke, M., B. Berger, and L. Cowen. 2008. 'Matt: local flexibility aids protein multiple structure alignment', *PLoS Comput Biol*, 4: e10.
Mi, Huaiyu, Anushya Muruganujan, Xiaosong Huang, Dustin Ebert, Caitlin Mills, Xinyu Guo, and Paul D. Thomas. 2019. 'Protocol Update for large-scale genome and gene function analysis with the PANTHER classification system (v.14.0)', *Nature Protocols*, 14: 703-21.
Mintseris, Julian, and Zhiping Weng. 2005. 'Structure, function, and evolution of transient and obligate protein–protein interactions', *Proceedings of the National Academy of Sciences of the United States of America*, 102: 10930-35.
Moreira, Irina S., Pedro A. Fernandes, and Maria J. Ramos. 2010. 'Protein–protein docking dealing with the unknown', *Journal of Computational Chemistry*, 31: 317-42.
Moult, J., K. Fidelis, A. Zemla, and T. Hubbard. 2003. 'Critical assessment of methods of protein structure prediction (CASP)-round V', *Proteins*, 53 Suppl 6: 334-9.
Murakami, Yoichi, Lokesh P. Tripathi, Philip Prathipati, and Kenji Mizuguchi. 2017. 'Network analysis and in silico prediction of protein–protein interactions with applications in drug discovery', *Current Opinion in Structural Biology*, 44: 134-42.
Negi, Surendra S., and Werner Braun. 2007. 'Statistical analysis of physical-chemical properties and

prediction of protein-protein interfaces', *Journal of Molecular Modeling*, 13: 1157-67.

Negi, Surendra S., Andrey A. Kolokoltsov, Catherine H. Schein, Robert A. Davey, and Werner Braun. 2006. 'Determining functionally important amino acid residues of the E1 protein of Venezuelan equine encephalitis virus', *Journal of Molecular Modeling*, 12: 921-29.

Neuvirth, H., R. Raz, and G. Schreiber. 2004a. 'ProMate: a structure based prediction program to identify the location of protein-protein binding sites', *J Mol Biol*, 338: 181-99.

Neuvirth, Hani, Uri Heinemann, David Birnbaum, Naftali Tishby, and Gideon Schreiber. 2007. 'ProMateus—an open research approach to protein-binding sites analysis', *Nucleic acids research*, 35: W543-W48.

Neuvirth, Hani, Ran Raz, and Gideon Schreiber. 2004b. 'ProMate: A Structure Based Prediction Program to Identify the Location of Protein–Protein Binding Sites', *Journal of Molecular Biology*, 338: 181-99.

Ngounou Wetie, A. G., I. Sokolowska, A. G. Woods, U. Roy, J. A. Loo, and C. C. Darie. 2013. 'Investigation of stable and transient protein-protein interactions: Past, present, and future', *Proteomics*, 13: 538-57.

Nooren, I. M., and J. M. Thornton. 2003a. 'Diversity of protein-protein interactions', *Embo j*, 22: 3486-92.

———. 2003b. 'Structural characterisation and functional significance of transient protein-protein interactions', *J Mol Biol*, 325: 991-1018.

Orchard, Sandra, Mais Ammari, Bruno Aranda, Lionel Breuza, Leonardo Briganti, Fiona Broackes-Carter, Nancy H. Campbell, Gayatri Chavali, Carol Chen, Noemi Del-Toro, Margaret Duesbury, Marine Dumousseau, Eugenia Galeota, Ursula Hinz, Marta Iannuccelli, Sruthi Jagannathan, Rafael Jimenez, Jyoti Khadake, Astrid Lagreid, Luana Licata, Ruth C. Lovering, Birgit Meldal, Anna N. Melidoni, Mila Milagros, Daniele Peluso, Livia Perfetto, Pablo Porras, Arathi Raghunath, Sylvie Ricard-Blum, Bernd Roechert, Andre Stutz, Michael Tognolli, Kim Van Roey, Gianni Cesareni, and Henning Hermjakob. 2014. 'The MIntAct project—IntAct as a common curation platform for 11 molecular interaction databases', *Nucleic acids research*, 42: D358-D63.

Paxman, Jason J., and Begoña Heras. 2017. 'Bioinformatics Tools and Resources for Analyzing Protein Structures.' in Shivakumar Keerthikumar and Suresh Mathivanan (eds.), *Proteome Bioinformatics* (Springer New York: New York, NY).

Pedamallu, C. S., and J. Posfai. 2010. 'Open source tool for prediction of genome wide protein-protein interaction network based on ortholog information', *Source Code Biol Med*, 5: 8.

Phizicky, E. M., and S. Fields. 1995. 'Protein-protein interactions: methods for detection and analysis', *Microbiol Rev*, 59: 94-123.

Piehler, J. 2005. 'New methodologies for measuring protein interactions in vivo and in vitro', *Curr Opin Struct Biol*, 15: 4-14.

Pierce, Brian G., Yuichiro Hourai, and Zhiping Weng. 2011. 'Accelerating Protein Docking in ZDOCK Using an Advanced 3D Convolution Library', *PLoS One*, 6: e24657.

Porollo, Aleksey, and Jaroslaw Meller. 2007. 'Prediction-Based Fingerprints of Protein – Protein Interactions', *Proteins-Structure Function and Bioinformatics*, 66: 630-45.

Rao, V. S., K. Srinivas, G. N. Sujini, and G. N. Kumar. 2014. 'Protein-protein interaction detection: methods and analysis', *Int J Proteomics*, 2014: 147648.

Rocha, Sara, Jorge Vieira, Noé Vázquez, Hugo López-Fernández, Florentino Fdez-Riverola, Miguel Reboiro-Jato, André D. Sousa, and Cristina P. Vieira. 2019. 'ATXN1 N-terminal region explains the binding differences of wild-type and expanded forms', *BMC Medical Genomics*, 12: 145.

Roy, Ambrish, Alper Kucukural, and Yang Zhang. 2010. 'I-TASSER: a unified platform for automated protein structure and function prediction', *Nature Protocols*, 5: 725-38.

Ryan, Daniel P., and Jacqueline M. Matthews. 2005. 'Protein–protein interactions in human disease', *Current Opinion in Structural Biology*, 15: 441-46.

Schneider, Nadine, Gudrun Lange, Sally Hindle, Robert Klein, and Matthias Rarey. 2013. 'A consistent description of HYdrogen bond and DEhydration energies in protein–ligand complexes: methods behind the HYDE scoring function', *Journal of Computer-Aided Molecular Design*, 27: 15-29.

Schöning-Stierand, Katrin, Konrad Diedrich, Rainer Fährrolfes, Florian Flachsenberg, Agnes Meyder, Eva Nittinger, Ruben Steinegger, and Matthias Rarey. 2020. 'ProteinsPlus: interactive analysis of protein–ligand binding interfaces', *Nucleic acids research*, 48: W48-W53.

Schreiber, G., G. Haran, and H. X. Zhou. 2009. 'Fundamental Aspects of Protein−Protein Association Kinetics', *Chemical Reviews*, 109: 839-60.

Serebriiskii, I. G., and E. A. Golemis. 2001. 'Two-hybrid system and false positives. Approaches to detection and elimination', *Methods Mol Biol*, 177: 123-34.

Shatnawi, Maad. 2015. 'Chapter 6 - Review of Recent Protein-Protein Interaction Techniques.' in Quoc Nam Tran and Hamid Arabnia (eds.), *Emerging Trends in Computational Biology, Bioinformatics, and Systems Biology* (Morgan Kaufmann: Boston).

Simons, K. T., C. Kooperberg, E. Huang, and D. Baker. 1997. 'Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions', *J Mol Biol*, 268: 209-25.

Skolnick, J., J. S. Fetrow, and A. Kolinski. 2000. 'Structural genomics and its importance for gene function analysis', *Nat Biotechnol*, 18: 283-7.

Smith, G. R., P. W. Fitzjohn, C. S. Page, and P. A. Bates. 2005. 'Incorporation of flexibility into rigid-body docking: applications in rounds 3-5 of CAPRI', *Proteins*, 60: 263-8.

Snel, B., G. Lehmann, P. Bork, and M. A. Huynen. 2000. 'STRING: a web-server to retrieve and display the repeatedly occurring neighbourhood of a gene', *Nucleic Acids Res*, 28: 3442-4.

Speck, S. H., W. H. Koppenol, J. K. Dethmers, N. Osheroff, E. Margoliash, and K. V. Rajagopalan. 1981. 'Definition of cytochrome c binding domains by chemical modification. Interaction of horse cytochrome c with beef sulfite oxidase and analysis of steady state kinetics', *J Biol Chem*, 256: 7394-400.

Stark, C., B. J. Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, and M. Tyers. 2006. 'BioGRID: a general repository for interaction datasets', *Nucleic Acids Res*, 34: D535-9.

Suter, B., J. F. Fontaine, R. Yildirimman, T. Raskó, M. H. Schaefer, A. Rasche, P. Porras, B. M. Vázquez-Álvarez, J. Russ, K. Rau, R. Foulle, M. Zenkner, K. Saar, R. Herwig, M. A. Andrade-Navarro, and E. E. Wanker. 2013. 'Development and application of a DNA microarray-based yeast two-hybrid system', *Nucleic Acids Res*, 41: 1496-507.

Szklarczyk, D., A. L. Gable, K. C. Nastou, D. Lyon, R. Kirsch, S. Pyysalo, N. T. Doncheva, M. Legeay, T. Fang, P. Bork, L. J. Jensen, and C. von Mering. 2021. 'The STRING database in 2021: customizable protein-protein networks, and functional characterization of user-uploaded gene/measurement sets', *Nucleic Acids Res*, 49: D605-d12.

Thomas, Paul, Michael Campbell, Anish Kejariwal, Huaiyu Mi, Brian Karlak, Robin Daverman, Karen Diemer, Anushya Muruganujan, and Apurva Narechania. 2003. 'PANTHER: A Library of Protein Families and Subfamilies Indexed by Function', *Genome research*, 13: 2129-41.

Thorn, K. S., and A. A. Bogan. 2001. 'ASEdb: a database of alanine mutations and their effects on the free energy of binding in protein interactions', *Bioinformatics*, 17: 284-5.

Tripathi, A., and R. Varadarajan. 2014. 'Residue specific contributions to stability and activity inferred from saturation mutagenesis and deep sequencing', *Curr Opin Struct Biol*, 24: 63-71.

Tripathi, Lokesh P., Yi-An Chen, Kenji Mizuguchi, and Yoichi Murakami. 2019. 'Network-Based Analysis for Biological Discovery.' in Shoba Ranganathan, Michael Gribskov, Kenta Nakai and Christian Schönbach (eds.), *Encyclopedia of Bioinformatics and Computational Biology* (Academic Press: Oxford).

Valente, Guilherme T., Marcio L. Acencio, Cesar Martins, and Ney Lemke. 2013. 'The Development of a Universal In Silico Predictor of Protein-Protein Interactions', *PLoS One*, 8: e65587.

van Zundert, G. C. P., J. P. G. L. M. Rodrigues, M. Trellet, C. Schmitz, P. L. Kastritis, E. Karaca, A. S. J. Melquiond, M. van Dijk, S. J. de Vries, and A. M. J. J. Bonvin. 2016. 'The HADDOCK2.2 Web Server: User-Friendly Integrative Modeling of Biomolecular Complexes', *Journal of Molecular Biology*, 428: 720-25.

Vázquez, Noé, Sara Rocha, Hugo López-Fernández, André Torres, Rui Camacho, Florentino Fdez-Riverola, Jorge Vieira, Cristina P. Vieira, and Miguel Reboiro-Jato. 2019a. 'EvoPPI 1.0: a Web Platform for Within- and Between-Species Multiple Interactome Comparisons and Application to Nine PolyQ Proteins Determining Neurodegenerative Diseases', *Interdisciplinary Sciences: Computational Life Sciences*, 11: 45-56.

———. 2019b. "EvoPPI: A Web Application to Compare Protein-Protein Interactions (PPIs) from Different Databases and Species." In, 149-56. Cham: Springer International Publishing.

Volkamer, A., D. Kuhn, T. Grombacher, F. Rippmann, and M. Rarey. 2012. 'Combining global and local measures for structure-based druggability predictions', *J Chem Inf Model*, 52: 360-72.

von Mering, Christian, Roland Krause, Berend Snel, Michael Cornell, Stephen G. Oliver, Stanley Fields, and Peer Bork. 2002. 'Comparative assessment of large-scale data sets of protein–protein interactions', *Nature*, 417: 399-403.

Vreven, Thom, Brian G. Pierce, Howook Hwang, and Zhiping Weng. 2013. 'Performance of ZDOCK in CAPRI rounds 20-26', *Proteins*, 81: 2175-82.

Waldspurger, Carl A. 2003. 'Memory resource management in VMware ESX server', *SIGOPS Oper. Syst. Rev.*, 36: 181–94.

Wang, Sheng, Jianzhu Ma, Jian Peng, and Jinbo Xu. 2013. 'Protein structure alignment beyond spatial proximity', *Scientific Reports*, 3.

Weiner, Scott J., Peter A. Kollman, David A. Case, U. Chandra Singh, Caterina Ghio, Guliano Alagona, Salvatore Profeta, and Paul Weiner. 1984. 'A new force field for molecular mechanical simulation of nucleic acids and proteins', *Journal of the American Chemical Society*, 106: 765-84.

Wells, James A., and Christopher L. McClendon. 2007. 'Reaching for high-hanging fruit in drug discovery at protein–protein interfaces', *Nature*, 450: 1001-09.

Wiehe, Kevin, Brian Pierce, Julian Mintseris, Wei Wei Tong, Robert Anderson, Rong Chen, and Zhiping Weng. 2005. 'ZDOCK and RDOCK performance in CAPRI rounds 3, 4, and 5', *Proteins: Structure, Function, and Bioinformatics*, 60: 207-13.

Wojcik, Jérôme, Ivo G. Boneca, and Pierre Legrain. 2002. 'Prediction, Assessment and Validation of Protein Interaction Maps in Bacteria', *Journal of Molecular Biology*, 323: 763-70.

Wojcik, Jérôme, and Vincent Schächter. 2001. 'Protein-protein interaction map inference using interacting domain profile pairs', *Bioinformatics*, 17: S296-S305.

Xu, D., and Y. Zhang. 2012. 'Ab initio protein structure assembly using continuous structure fragments and optimized knowledge-based force field', *Proteins*, 80: 1715-35.

Yan, C., F. Wu, R. L. Jernigan, D. Dobbs, and V. Honavar. 2008. 'Characterization of protein-protein interfaces', *Protein J*, 27: 59-70.

Yanagida, M. 2002. 'Functional proteomics; current achievements', *J Chromatogr B Analyt Technol Biomed Life Sci*, 771: 89-106.

Yang, J., W. Zhang, B. He, S. E. Walker, H. Zhang, B. Govindarajoo, J. Virtanen, Z. Xue, H. B. Shen, and Y. Zhang. 2016. 'Template-based protein structure prediction in CASP11 and retrospect of I-TASSER in the last decade', *Proteins*, 84 Suppl 1: 233-46.

Yang, Jianyi, Renxiang Yan, Ambrish Roy, Dong Xu, Jonathan Poisson, and Yang Zhang. 2015. 'The I-TASSER Suite: protein structure and function prediction', *Nature Methods*, 12: 7-8.

Zhang, Q. C., L. Deng, M. Fisher, J. Guan, B. Honig, and D. Petrey. 2011. 'PredUs: a web server for predicting protein interfaces using structural neighbors', *Nucleic Acids Res*, 39: W283-7.

Zhang, Qiangfeng Cliff, Donald Petrey, Lei Deng, Li Qiang, Yu Shi, Chan Aye Thu, Brygida Bisikirska, Celine Lefebvre, Domenico Accili, Tony Hunter, Tom Maniatis, Andrea Califano, and Barry Honig. 2012. 'Structure-based prediction of protein–protein interactions on a genome-wide scale', *Nature*, 490: 556-60.

Zhang, Y., and J. Skolnick. 2004a. 'Automated structure prediction of weakly homologous proteins on a genomic scale', *Proc Natl Acad Sci U S A*, 101: 7594-9.

———. 2004b. 'Scoring function for automated assessment of protein structure template quality', *Proteins*, 57: 702-10.

———. 2005. 'TM-align: a protein structure alignment algorithm based on the TM-score', *Nucleic Acids Res*, 33: 2302-9.

Zhang, Yang. 2008. 'I-TASSER server for protein 3D structure prediction', *BMC Bioinformatics*, 9: 40.

Zhao, Chunyu, and Ahmet Sacan. 2015. 'UniAlign: protein structure alignment meets evolution', *Bioinformatics*, 31: 3139-46.

Supplementary

Supplementary Table 1 - Structural comparison with TM- align between the interactor proteins from (Rocha et al. 2019) and their equivalents from AlphaFold DB.

Interactors	Uniprot ID	AlphaFold link	TM-score (chain1)	TM-score (chain2)
ADD3	Q9UEY8	https://alphafold.ebi.ac.uk/files/AF-Q9UEY8-F1-model_v1.pdb	0.38356	0.38356
ARID5A	Q03989	https://alphafold.ebi.ac.uk/files/AF-Q03989-F1-model_v1.pdb	0.14074	0.14074
ASNS	P08243	https://alphafold.ebi.ac.uk/files/AF-P08243-F1-model_v1.pdb	0.87133	0.87133
BAALC	Q8WXS3	https://alphafold.ebi.ac.uk/files/AF-Q8WXS3-F1-model_v1.pdb	0.15165	0.17762
BASP1	P80723	https://alphafold.ebi.ac.uk/files/AF-P80723-F1-model_v1.pdb	0.12229	0.12229
C16orf5	Q9H305	https://alphafold.ebi.ac.uk/files/AF-Q9H305-F1-model_v1.pdb	0.19034	0.19034
C2orf27B	Q580R0	https://alphafold.ebi.ac.uk/files/AF-Q580R0-F1-model_v1.pdb	0.21757	0.2132
CAMK2B	Q13554	https://alphafold.ebi.ac.uk/files/AF-Q13554-F1-model_v1.pdb	0.63187	0.63187
CHRNA7	P36544	https://alphafold.ebi.ac.uk/files/AF-P36544-F1-model_v1.pdb	0.67725	0.67725
CREM	Q03060	https://alphafold.ebi.ac.uk/files/AF-Q03060-F1-model_v1.pdb	0.15715	0.16234
CRIP2	P52943	https://alphafold.ebi.ac.uk/files/AF-P52943-F1-model_v1.pdb	0.27783	0.27783
CRK	P46108	https://alphafold.ebi.ac.uk/files/AF-P46108-F1-model_v1.pdb	0.32392	0.32392
CRY2	Q49AN0	https://alphafold.ebi.ac.uk/files/AF-Q49AN0-F1-model_v1.pdb	0.80283	0.80283
CXorf27	O75409	https://alphafold.ebi.ac.uk/files/AF-O75409-F1-model_v1.pdb	0.28031	0.28031
DHRSX	Q8N5I4	https://alphafold.ebi.ac.uk/files/AF-Q8N5I4-F1-model_v1.pdb	0.82073	0.82073
DHX37	Q8IY37	https://alphafold.ebi.ac.uk/files/AF-Q8IY37-F1-model_v1.pdb	0.56811	0.56811
DIXDC1	Q155Q3	https://alphafold.ebi.ac.uk/files/AF-Q155Q3-F1-model_v1.pdb	0.17837	0.17837
EIF1B	O60739	https://alphafold.ebi.ac.uk/files/AF-O60739-F1-model_v1.pdb	0.76998	0.76998
EIF3F	O00303	https://alphafold.ebi.ac.uk/files/AF-O00303-F1-model_v1.pdb	0.22441	0.19832
ESRRA	P11474	https://alphafold.ebi.ac.uk/files/AF-P11474-F1-model_v1.pdb	0.51667	0.51667
ETV4	P43268	https://alphafold.ebi.ac.uk/files/AF-P43268-F1-model_v1.pdb	0.15181	0.15181
FAM46A	Q96IP4	https://alphafold.ebi.ac.uk/files/AF-Q96IP4-F1-model_v1.pdb	0.32818	0.32818

FAM46B	Q96A09	https://alphafold.ebi.ac.uk/files/AF-Q96A09-F1-model_v1.pdb	0.24574	0.24574
FAR1	Q8WVX9	https://alphafold.ebi.ac.uk/files/AF-Q8WVX9-F1-model_v1.pdb	0.67016	0.67016
FOSL1	P15407	https://alphafold.ebi.ac.uk/files/AF-P15407-F1-model_v1.pdb	0.17059	0.17059
GATAD1	Q8WUU5	https://alphafold.ebi.ac.uk/files/AF-Q8WUU5-F1-model_v1.pdb	0.27873	0.27873
GGA2	Q9UJY4	https://alphafold.ebi.ac.uk/files/AF-Q9UJY4-F1-model_v1.pdb	0.2869	0.2869
HEY2	Q9UBP5	https://alphafold.ebi.ac.uk/files/AF-Q9UBP5-F1-model_v1.pdb	0.26409	0.26409
HEYL	Q9NQ87	https://alphafold.ebi.ac.uk/files/AF-Q9NQ87-F1-model_v1.pdb	0.19942	0.19942
HNRPLL	Q8WVV9	https://alphafold.ebi.ac.uk/files/AF-Q8WVV9-F1-model_v1.pdb	0.24156	0.24156
ILVBL	A1L0T0	https://alphafold.ebi.ac.uk/files/AF-A1L0T0-F1-model_v1.pdb	0.80527	0.80527
IMMT	Q16891	https://alphafold.ebi.ac.uk/files/AF-Q16891-F1-model_v1.pdb	0.17549	0.17549
KCTD15	Q96SI1	https://alphafold.ebi.ac.uk/files/AF-Q96SI1-F1-model_v1.pdb	0.39129	0.39129
KIF22	Q14807	https://alphafold.ebi.ac.uk/files/AF-Q14807-F1-model_v1.pdb	0.46554	0.46554
LASP1	Q14847	https://alphafold.ebi.ac.uk/files/AF-Q14847-F1-model_v1.pdb	0.2523	0.2523
LITAF	Q99732	https://alphafold.ebi.ac.uk/files/AF-Q99732-F1-model_v1.pdb	0.21974	0.21974
LPAR2	Q9HBW0	https://alphafold.ebi.ac.uk/files/AF-Q9HBW0-F1-model_v1.pdb	0.83259	0.83259
MAGEB18	Q96M61	https://alphafold.ebi.ac.uk/files/AF-Q96M61-F1-model_v1.pdb	0.48369	0.48369
MAGEB2	O15479	https://alphafold.ebi.ac.uk/files/AF-O15479-F1-model_v1.pdb	0.40424	0.40424
MAGEB6	Q8N7X4	https://alphafold.ebi.ac.uk/files/AF-Q8N7X4-F1-model_v1.pdb	0.18966	0.18966
MCART1	Q9H1U9	https://alphafold.ebi.ac.uk/files/AF-Q9H1U9-F1-model_v1.pdb	0.77359	0.77359
MLST8	Q9BVC4	https://alphafold.ebi.ac.uk/files/AF-Q9BVC4-F1-model_v1.pdb	0.92421	0.92421
MSX2	P35548	https://alphafold.ebi.ac.uk/files/AF-P35548-F1-model_v1.pdb	0.20151	0.20151
NCAM1	P13591	https://alphafold.ebi.ac.uk/files/AF-P13591-F1-model_v1.pdb	0.25372	0.25372
OTX2	P32243	https://alphafold.ebi.ac.uk/files/AF-P32243-F1-model_v1.pdb	0.26305	0.26305
PIAS1	O75925	https://alphafold.ebi.ac.uk/files/AF-O75925-F1-model_v1.pdb	0.44587	0.44587
PLEKHB1	Q9UF11	https://alphafold.ebi.ac.uk/files/AF-Q9UF11-F1-model_v1.pdb	0.49592	0.49592
PPAT	Q06203	https://alphafold.ebi.ac.uk/files/AF-Q06203-F1-model_v1.pdb	0.86027	0.86027

PSPH	P78330	https://alphafold.ebi.ac.uk/files/AF-P78330-F1-model_v1.pdb	0.9666	0.9666
QKI	Q96PU8	https://alphafold.ebi.ac.uk/files/AF-Q96PU8-F1-model_v1.pdb	0.4706	0.4706
RAI2	Q9Y5P3	https://alphafold.ebi.ac.uk/files/AF-Q9Y5P3-F1-model_v1.pdb	0.11809	0.11809
RAPGEF1	Q13905	https://alphafold.ebi.ac.uk/files/AF-Q13905-F1-model_v1.pdb	0.37063	0.37063
RBFOX2	O43251	https://alphafold.ebi.ac.uk/files/AF-O43251-F1-model_v1.pdb	0.20787	0.20787
RBM26	Q5T8P6	https://alphafold.ebi.ac.uk/files/AF-Q5T8P6-F1-model_v1.pdb	0.13183	0.13183
SEMA4G	Q9NTN9	https://alphafold.ebi.ac.uk/files/AF-Q9NTN9-F1-model_v1.pdb	0.52806	0.52806
SF1	Q15637	https://alphafold.ebi.ac.uk/files/AF-Q15637-F1-model_v1.pdb	0.1941	0.1941
SLC6A13	Q9NSD5	https://alphafold.ebi.ac.uk/files/AF-Q9NSD5-F1-model_v1.pdb	0.89642	0.89642
STAM2	O75886	https://alphafold.ebi.ac.uk/files/AF-O75886-F1-model_v1.pdb	0.18664	0.18664
SV2A	Q7L0J3	https://alphafold.ebi.ac.uk/files/AF-Q7L0J3-F1-model_v1.pdb	0.20249	0.20249
TCTA	P57738	https://alphafold.ebi.ac.uk/files/AF-P57738-F1-model_v1.pdb	0.29469	0.29469
TMX2	Q9Y320	https://alphafold.ebi.ac.uk/files/AF-Q9Y320-F1-model_v1.pdb	0.45459	0.45459
TOMM20	Q15388	https://alphafold.ebi.ac.uk/files/AF-Q15388-F1-model_v1.pdb	0.43261	0.43261
TP53I11	O14683	https://alphafold.ebi.ac.uk/files/AF-O14683-F1-model_v1.pdb	0.31848	0.31848
TRIM38	O00635	https://alphafold.ebi.ac.uk/files/AF-O00635-F1-model_v1.pdb	0.37281	0.37281
TSC1	Q92574	https://alphafold.ebi.ac.uk/files/AF-Q92574-F1-model_v1.pdb	0.25223	0.25223
TTRAP	O95551	https://alphafold.ebi.ac.uk/files/AF-O95551-F1-model_v1.pdb	0.67217	0.67217
UHRF2	Q96PU4	https://alphafold.ebi.ac.uk/files/AF-Q96PU4-F1-model_v1.pdb	0.22005	0.22005
WBSCR16	Q96I51	https://alphafold.ebi.ac.uk/files/AF-Q96I51-F1-model_v1.pdb	0.8451	0.8451
YWHAE	P62258	https://alphafold.ebi.ac.uk/files/AF-P62258-F1-model_v1.pdb	0.92746	0.92746
ZC3H10	Q96K80	https://alphafold.ebi.ac.uk/files/AF-Q96K80-F1-model_v1.pdb	0.16791	0.16791
ZSCAN1	Q8NBB4	https://alphafold.ebi.ac.uk/files/AF-Q8NBB4-F1-model_v1.pdb	0.21964	0.21964

P2I2P Pipeline Manual

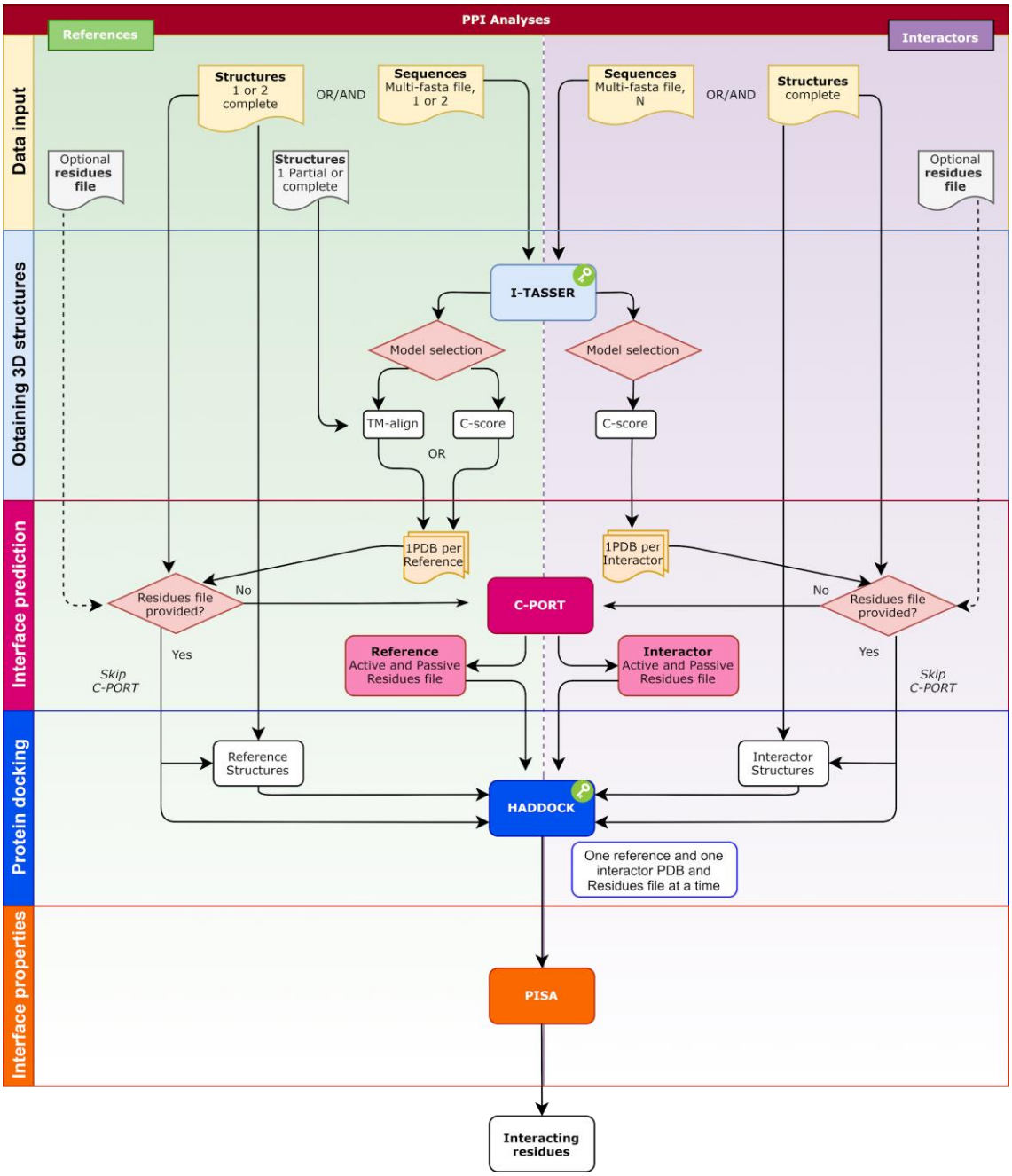

Figure 1 – P2I2P pipeline workflow.

Contents

1. Introduction

This docker image runs a multi-software bioinformatics pipeline that allows for the identification of protein-protein interactions interfaces and their properties.

Starting from a set of Reference and Interactor proteins, it allows the user to:

- Obtain the 3D structure of all submitted proteins with *I-TASSER*.
- select the most accurate model using C-score, or TM-align (for the cases where two different protein forms are analyzed);
- Predict interface active and passive residues, with *CPORT*;
- Predict the structure of the interacting proteins complex, with *HADDOCK*;
- Obtain interface information, with *PDBePISA*.

The containerization of this tool allows it to run on several computer environments and different operating systems. Therefore, to use it, you only need a version of Docker installed on your machine.

In this manual we explain the internal Configuration File (1.1) and how to fill in (what you need before initializing a simple run on your test data; 2). We also describe further customization options for other possible runs (3), which can be adapted to your personal preferences and experimental data.

1.1. Getting started

This manual is written under the assumption you understand basic concepts of protein-protein interactions and are familiar with handling Docker images. If you're new to any of these subjects, we recommend you to read (Schreiber, 2020) and pay a visit to the Docker guides.

This image belongs to a larger project called Bioinformatics Docker Images Project. On the beginning of this web page, you will find instructions on how to install Docker, if you haven't done so already.

Use `docker pull pegi3s/p2i2p` to obtain the image. Then follow one of these instructions to run the pipeline:

- In unix distributions, you should adapt and run the following command: `docker run --rm -v /var/run/docker.sock:/var/run/docker.sock -v /your/data/dir:/your/data/dir pegi3s/p2i2p bash -c "/opt/run"`
- In Windows, you should adapt and run the following command: `docker run --rm -v /var/run/docker.sock:/var/run/docker.sock -v "/c/Users/User_name/dir/":"/c/Users/User_name/dir/" pegi3s/p2i2p bash -c "/opt/run"`; Please note that data must be under the same drive than the Docker Toolbox installation (usually C:) and in a folder with write permissions (e.g. C:/Users/User_name/).

Please note that the original software licenses still apply.

2. Running the Pipeline

2.1. What you'll need

To run the image, you will essentially need two things:

- A Configuration File, which is a simple text file with variables for customizing your run (detailed in the next section).
- Your input data, in the form of Fasta files and/or PDB structure files (more information about these in section 2.3.).

Additionally, for the *I-TASSER* and *HADDOCK* steps of the pipeline, you will need to have an account in each web server and take note of your credentials. If you haven't done so already, you can register here and here, respectively.

2.2. The Configuration File

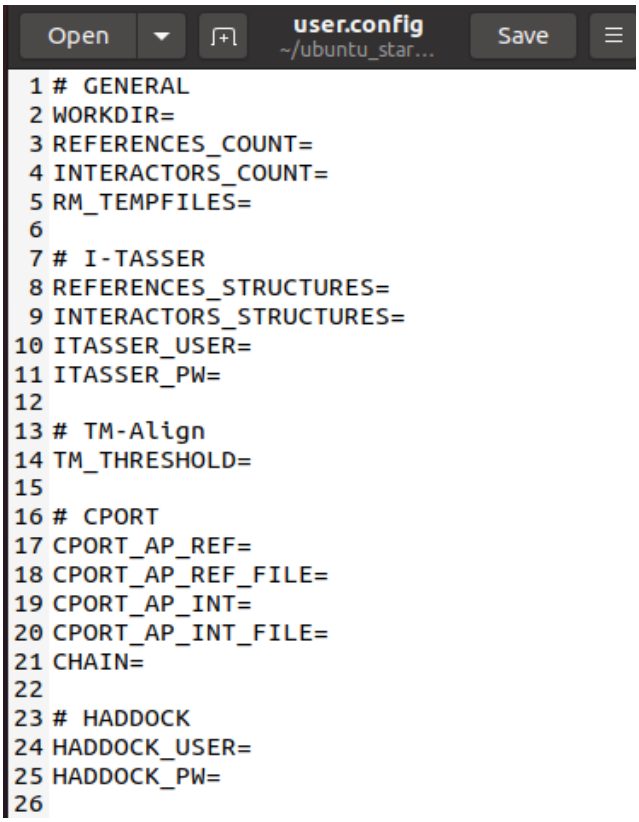

Figure 2 - A template of the user.config file.

The configuration file (Fig. 2) contains variables used internally by the pipeline, most of which are intended for the customization of your run. It is a simple text file, with one variable per line. Each variable belongs to one of 5 different sets of variables, some of which regulate how each individual program runs (see bellow).

This file must be named `user.config` and must be placed inside your working directory.

General variables

- `WORKDIR` – the absolute path to the directory where the pipeline will run, with the input files you provide, and where all the results will be generated;
- `REFERENCES_COUNT` (optional) – 1 or 2 – this is the number of Reference proteins you're using. If you don't provide this value, it counts the number of protein sequences from your FASTA file;
- `INTERACTORS_COUNT` (optional) – this is the number of Interactor proteins you're using. If you don't provide this value, it counts the number of interactor sequences from your FASTA file;
- `RM_TEMPFILES` (optional) – 0 (no) or 1 (yes) – choose whether to remove the temporary files generated by this tool, at the end of the run, or keep them;

I-TASSER variables

- `REFERENCES_STRUCTURES` – 1, 2 or 3 – determines which 3D model is to be used for your Reference protein(s);
  - ❖ Option 1: uses, for each reference protein, the *I-TASSER* model with the highest C-Score;
  - ❖ Option 2: compares all models obtained with *I-TASSER* with a partial (or complete) 3D structure, given by the user, and chooses the model with the highest structural similarity (using *TM-align*). This option requires that you provide a PDB file, with the partial/complete structure, in a folder named **`References_Structures`** inside `WORKDIR`;
  - ❖ Option 3: skips *I-TASSER* altogether and uses as the final model a structure file provided by the user. This option requires that you provide a PDB file, per reference, in a folder named **`References_Structures`** inside `WORKDIR`**;**
- `INTERACTORS_STRUCTURES` – 1 or 2 – determines which 3D model is to be used for your Interactor proteins
  - ❖ Option 1: uses, for each interactor protein, the *I-TASSER* model with the highest C-Score.
  - ❖ Option 2: skips *I-TASSER* and uses as the final model a structure file provided by the user. This option requires that you provide a PDB file, per interactor, in a folder named **`Interactors_Structures`** inside `WORKDIR`;
- `I-TASSER_USER` – to specify your *I-TASSER* account username (you can register here);
- `I-TASSER_PW` – to specify your *I-TASSER* account password (you can register here);

TM-align variables

- `TM_THRESHOLD` – range [0-1] – to specify the value you want to use as a threshold for the structure similarity between Reference proteins. Accepts a value between 0 and 1, and if you don't fill this field, it assumes a value of 0.5.

CPORT variables

- `CPORT_AP_REF` – 0 (no) or 1 (yes) – where you specify, for the Reference protein(s), whether you provide a file with the active and passive residues for *HADDOCK*. If you provide a file, *C-PORT* will be skipped for the Reference(s). If you don't fill this field, it assumes you want to predict the residues, therefore it runs *C-PORT*;
- `CPORT_AP_REF_FILE` – to specify the absolute path to the file which contains both active and passive residues for the Reference protein(s). This field only makes sense if in the previous variable you selected (1), otherwise leave it empty. This file should contain 2 lines only;
- `CPORT_AP_INT` – 0 (no) or 1 (yes) – where you specify, for the Interactor proteins, whether you provide a file with the active and passive residues for *HADDOCK*. If you provide a file, *C-PORT* will be skipped for the Interactors. If you don't fill this field, it assumes you want to predict the residues, therefore it runs *C-PORT*;
- `CPORT_AP_INT_FILE` – to specify the absolute path to the file which contains both active and passive residues for the Interactor proteins. This field only makes sense if in the previous variable you selected (1), otherwise leave it empty. This file should contain 2 lines only;
- `CHAIN` – to specify which chain you want *C-PORT* to use for prediction. If you don't fill this field, it will use chain A as default.

HADDOCK variables

- `HADDOCK_USER` – to specify your *HADDOCK* account username (you can register here);
- `HADDOCK_PW` – to specify your *HADDOCK* account password (you can register here).

2.3. Files to provide

For both References and Interactor proteins, you should provide at least one of these files:

- FASTA file with the amino acid sequences of your protein, and/or
- PDB file of your protein structure.

Depending on your configuration values, for either Reference or Interactor proteins, you might need to provide:

- Residues file – a simple text file with the interface Active and Passive residues for *HADDOCK*. This should be a file with only 2 lines, where the Active residues are on line 1 and Passive residues on line 2.

Similarly, for the Reference(s) protein(s), you might need to provide:

- Partial or complete PDB structure file – to compare with your Reference(s) and choose the model with the highest similarity (using *TM-align*).

3. Customizing your run

Due to the set of variable options that this pipeline gives you, setting up a run that matches the aim of your research is very straightforward.

In this section, we present the available runs you can execute with this tool and a tutorial on how to replicate them yourself. An explanation of each variable herein described is present in chapter 2, section 2.2.

3.1. "Complete" run

For this option you should first set `REFERENCES_STRUCTURES=1` and `INTERACTORS_STRUCTURES=1` on your user.config, which will tell the pipeline to obtain the 3D structure of your Reference and Interactor protein(s) with *I-TASSER* and choose the model(s) with the highest C-score.

Then, for both Reference(s) and Interactors, you should put a multi-FASTA file (with extension or not) with the amino acid sequences of each of your proteins in `/Your_Workdir/References` and `/Your_Workdir/Interactors` – as exemplified in Figure 3.

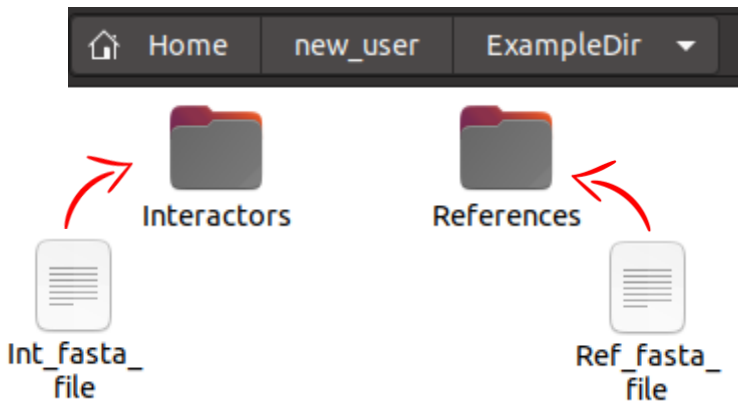

Figure 3 - FASTA file location scheme. Each FASTA file should be put inside its respective folder, in the working directory.

3.2. Simple run + TM-align – Choose the most similar model

For this option you should set `REFERENCES_STRUCTURES=2` on your user.config, which will tell the pipeline to obtain the 3D structure of your Reference protein(s) with *I-TASSER*, but compare all resulting models with a (partial or complete) structure you provide and choose the model with the highest structural similarity – with TM-align. The choice is made according to the highest TM-score value, which measures the structural similarity between two protein structures.

Then, for both Reference(s) and Interactors, you should put a multi-FASTA file (with extension or not) with the amino acid sequences of each of your proteins in `/Your_Workdir/References` and `/Your_Workdir/Interactors` – as exemplified in figure 3.

Additionally, to compare with your Reference protein(s), you should put a PDB structure (or two PDBs, in case you have two reference proteins) inside `/Your_Workdir/Reference_Structures`, as shown in Figure 4.

It is recommended that the References' structure files are named "Ref1" and "Ref2" although they can be set to anything, as long as the file which comes first in alphabetical order corresponds to Reference number 1. The Interactors' files can have any name.

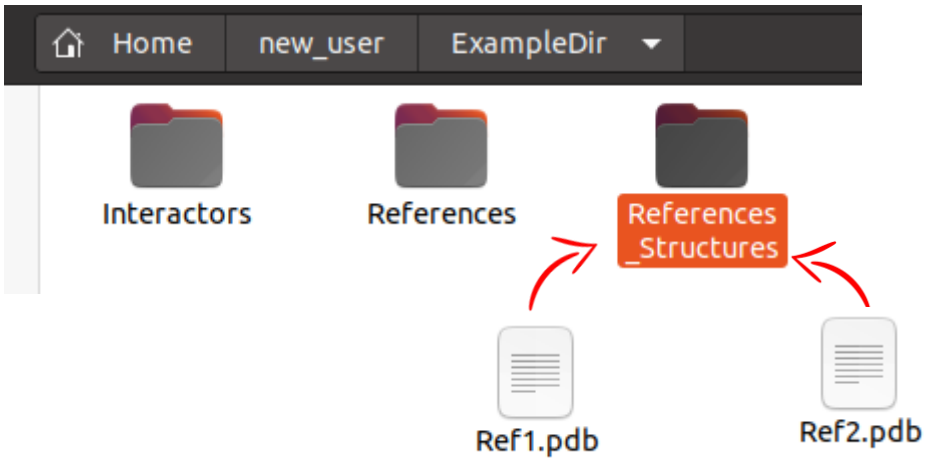

Figure 4 - Structure file location scheme. Each PDB file for the Reference sequences should be only put inside this folder, in the working directory.

3.3. Skip *I-TASSER* – Use the given 3D structures

You have the option to run the pipeline with your own protein structures, therefore *I-TASSER* will not run and structural models will not be predicted. You can skip *I-TASSER* for the Reference(s), for the Interactors or even for both of them.

For this option, you should set `REFERENCES_STRUCTURES=3` on your user.config, which will tell the pipeline to skip the *I-TASSER* step and use as the Reference(s) model(s) the structures you provide. Similarly for the Interactors, you should set `INTERACTORS_STRUCTURES=2`, and so the structures you provide will be used as the 3D models for the Interactors.

Therefore, you do not have to provide the FASTA files with the protein sequences, for the pipeline to run. You must instead write in the user.config how many Reference(s) and Interactors will be analyzed. For this, specify their number with `REFERENCES_COUNT` and `INTERACTORS_COUNT`.

For the files, you should place a PDB structure file for each Reference protein inside `/Your_Workdir/References_Structures` and similarly, for each Interactor protein, inside `/Your_Workdir/Interactors_Structures` – as shown in figure 5.

It is recommended that the References' structure files are named "Ref1" and "Ref2" although they can be set to anything, as long as the file which comes first in alphabetical order corresponds to Reference number 1 if active/passive residues files are provided. The Interactors' files can have any name.

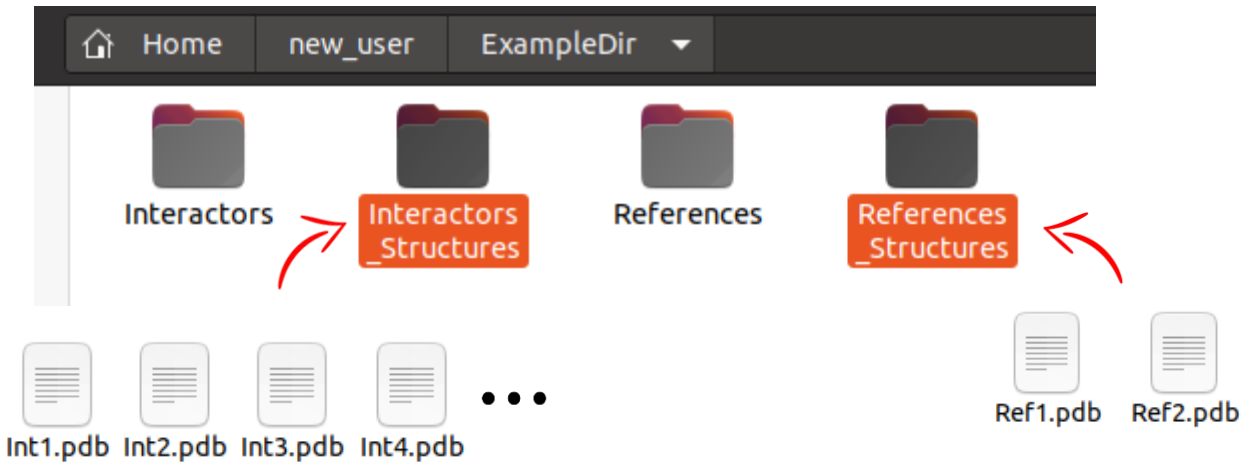

Figure 5 - Structure files location scheme. Each PDB file should be put inside its respective folder, in the working directory.

3.4. Skip CPORT – Use the given active and passive residues

You have the option to run the pipeline with a residues file provided by you, therefore *CPORT* will not run and the interface active and passive residues will not be predicted. You can skip *CPORT* for the Reference(s), for the Interactors or even for both of them.

For this option you should set `CPORT_AP_REF=1` on your user.config, which will tell the pipeline to skip the *CPORT* step and use as the Reference(s) active and passive residues the ones you provide in an external file. The absolute path to this file should be provided in `CPORT_AP_REF_FILE`.

Similarly, for the Interactors, you should set `CPORT_AP_INT=1`, which will tell the pipeline to skip the *CPORT* step and use as the active and passive residues the ones you provide in an external file. The absolute path to this file should be provided in `CPORT_AP_INT_FILE`.

The file for the Interactor proteins' residues should have 2 lines only, where in the first line are the active residues and in the second the passive ones, whereas the file for the Reference proteins' residues should have 4 lines (if you have two Reference proteins), where the first 2 lines correspond to the residues for the first Reference and the other 2 for the other protein – as shown in figure 6. The same active and passive interactor sites are assumed for all interactor proteins.

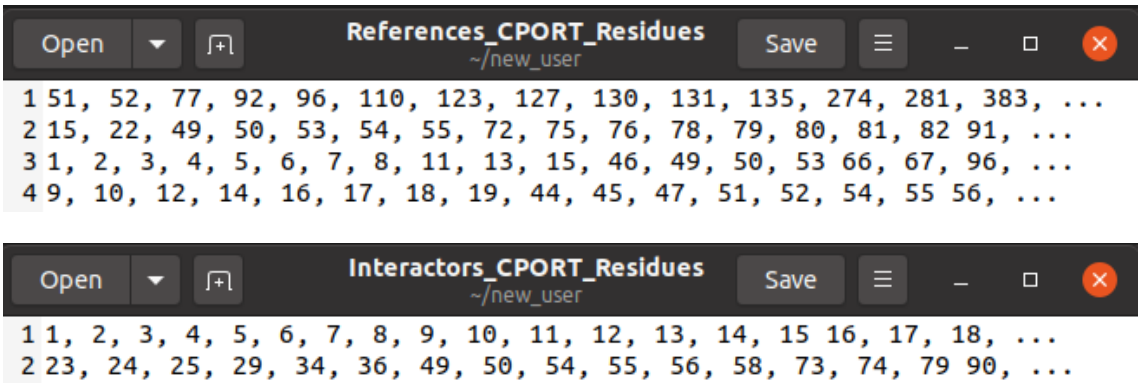

Figure 6 - Example of the residues files.

3.5. Skip *I-TASSER* and *CPORT*

With this docker image, you have the option to skip both of these tools which means you can use your own protein 3D structures and interface residues, meaning you will also reduce the time and computational effort you spend in your analysis.

For this option, you should set `REFERENCES_STRUCTURES=3` on your user.config, which will tell the pipeline to skip the *I-TASSER* step and use as the Reference(s) model(s) the structures you provide. Similarly for the Interactors, you should set `INTERACTORS_STRUCTURES=2`, and so the structures you provide will be used as the 3D models for the Interactors.

Therefore, you do not have to provide the FASTA files with the protein sequences, for the pipeline to run you must instead write in the user.config how many Reference(s) and Interactors will be analyzed. For this, specify their number with `REFERENCES_COUNT` and `INTERACTORS_COUNT`.

For the files, you should place a PDB structure file for each Reference protein inside `/Your_Workdir/References_Structures` and similarly, for each Interactor protein, inside `/Your_Workdir/Interactors_Structures` – as shown in figure 5.

Additionally, you should set `CPORT_AP_REF=1` on your user.config, which will tell the pipeline to skip the *CPORT* step and use as the Reference(s) active and passive residues the ones you provide in an external file. The absolute path to this file should be provided in `CPORT_AP_REF_FILE`.

Similarly, for the Interactors, you should set `CPORT_AP_INT=1`, which will tell the pipeline to skip the *CPORT* step and use as the active and passive residues the ones you provide in an external file. The absolute path to this file should be provided in `CPORT_AP_INT_FILE`.

Each of the files you provide containing the interface active and passive residues should have 2 lines only, where in the first line are the active residues and in the second the passive ones – as shown in figure 6.

4. Warnings and final remarks

1*.* At the beginning of the pipeline, each web server will be tested and if there is at least one that does not appear to be online, it will warn you, so you can abort the analysis or continue at your own risk.

2. After obtaining the 3D structures for your Reference proteins with *I-TASSER*, in the next step *CPORT* can fail for one of them (only if you choose to run the pipeline with 2 Reference proteins). In that case, the program will ask if you want to abort the pipeline or alternatively continue the remaining analysis with the other one.

3*.* If you chose to run with two Reference proteins, the pipeline will compare both of them structurally using *TM-align*. If your proteins have a structural similarity that is lower than the threshold you chose in `TM_THRESHOLD`, the pipeline will abort. In case you don't require them to share some similarity with each other you must set this variable to **0 (zero)**.

4. At the end of the run, the pipeline will move the results from *PISA* to the working dir. However, the remaining results from previous steps (the protein structures from *I-TASSER*, predicted residues from *CPORT*, HADDOCK results) will be inside a temporary folder (/your/working/dir/tempFolder). If these results are also important to you, do not forget to set `RM_TEMPFILES` to **0 (zero)**.