UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE BIOLOGIA ANIMAL

ESTUDO COMPUTACIONAL DAS INTERACÇÕES PROTEÍNA-PROTEÍNA

Dissertação orientada pelo Prof. Dr. António Ferreira (FCUL) e co-orientada pelo Prof. Dr. Ludwig Krippahl (FCT-UNL)

Sérgio Miguel Cardoso Marcelino dos Santos

Mestrado em Bioinformática e Biologia Computacional
Especialização em Bioinformática

Lisboa

2010

Agradecimentos

Gostaria de agradecer ao Prof. António Ferreira e ao Prof. Ludwig Krippahl pela orientação e pelo apoio no desenvolvimento deste trabalho.

Resumo

O reconhecimento molecular é um processo chave em sistemas biológicos. A replicação e transcrição do ADN, a adesão celular, as cascatas de sinalização e ciclos metabólicos são alguns dos processos que têm por base o reconhecimento molecular. A compreensão destes processos exige que se conheçam as interacções de proteínas que estão na base dos mesmos.

O modo como duas proteínas interagem pode ser difícil de prever, sobretudo se estas estabelecerem interacções transientes. O *Docking* é um método computacional que permite prever o modo de ligação entre duas moléculas e que tem potencial na previsão da interacção de complexos transientes.

Os métodos para prever interface de proteínas podem ser baseados unicamente nas propriedades geométricas, físico-químicas e estatísticas da superfície ou podem incorporar também informação evolucionária na forma de certas medidas de conservação derivadas de alinhamentos de múltiplas sequências (MSA). Ao longo do tempo ocorrem substituições de aminoácidos nas proteínas. Substituições que estabilizem a interface entre monómeros são favorecidas por selecção natural. Se uma mutação num monómero induz uma mutação noutro monómero do mesmo complexo, diz-se que as mutações estão correlacionadas. Estas mutações podem ser determinadas analisando as correlações entre alterações em pares de posições em MSA. Já foi demonstrado que pares de aminoácidos correlacionados estão significativamente mais perto uns dos outros do que pares não correlacionados e que estes podem ser usados para descriminar entre soluções correctas e incorrectas em métodos de *docking*.

Neste trabalho desenvolveu-se um sistema automatizado constituído por ferramentas em Python que integraram software disponível *online*, tal como o BLAST, o ClustalW e algoritmos de determinação de covariações, com o objectivo de determinar dados de coevolução que permitissem filtrar soluções de *docking* de complexos transientes.

Palavras-Chave: Coevolução; *Docking*; Mutações Correlacionadas; Interacção de Proteínas; MSA.

Abstract

Molecular recognition is a key process in biological systems. DNA replication and transcription, cellular adhesion, signaling cascades and metabolic cycles are some of the processes that underlie molecular recognition. In order to understand these processes it is of utmost importance to know the protein interactions that are on their origin.

The way in which two proteins interact might be difficult to predict, especially if they establish transient interactions. Docking is a computational method that allows the prediction of the binding mode between two molecules and has potential in predicting transient complexes.

Methods that predict protein interfaces can be based solely on geometric, statistical and physical-chemical properties of the surface or they can also incorporate evolutionary data related to amino acid conservation that is extracted from multiple sequence alignment (MSA). Throughout time amino acid substitutions occur. Substitutions that stabilize the interface between monomers are favored by natural selection. If a mutation within a monomer induces a mutation on another monomer of the same complex, it is considered that these mutations are correlated. These mutations can be determined by analysis of the correlations between a pair of amino acids in MSA. It has been demonstrated that pairs of amino acids that are correlated are significantly closer together in the structure when compared to pairs that are not correlated and correlated pairs can be used to distinguish right from wrong solutions in docking methods.

In this project, an automated system was developed that uses Python tools to integrate software available online such as BLAST, ClustalW and algorithms to determine co-variation with the objective of determining co-evolution data that allow the filtering of docking solutions of transient complexes.

Keywords: Co-evolution; Docking; Correlated Mutations; Protein Interactions; MSA.

Abreviaturas

MSA – *Multiple Sequence Aligments*

Pdb – *Protein Data Bank*

UniProt – *Universal Protein Resource*

PSI-BLAST – *Position-Specific Iterated BLAST*

ELSC – *Perturbation, Explicit Likelihood of Subset Covariation*

Índice

1 Objectivo

Automatizar a identificação de coevolução em proteínas visando o uso desta informação na filtração de soluções de *docking* aplicado a complexos transientes.

2 Introdução

Actualmente há uma grande quantidade informação sobre genes e proteínas. Extrair conhecimento do enorme volume de dados existente é uma tarefa extensa e complexa. Portanto, é necessário criar-se ferramentas informáticas que permitam lidar com este problema.

O reconhecimento molecular é um processo chave em sistemas biológicos. A replicação e transcrição do DNA, a adesão celular, as cascatas de sinalização e ciclos metabólicos são alguns dos múltiplos processos que têm por base o reconhecimento molecular.[1] A compreensão destes processos exige que se compreendam as interacções de proteínas que estão na base dos mesmos.

Muitas proteínas desempenham as suas funções metabólicas como componentes de complexos estáveis ou através de interacções transientes com outras proteínas. A informação experimental mais detalhada sobre estrutura de proteínas provém de estruturas raio-X de alta resolução. Todos os dias se conhecem mais estruturas de proteínas, com isto aumenta a necessidade de compreender e caracterizar o papel dessas proteínas e das vias metabólicas em que as mesmas participam[2]. Estas estruturas fornecem pistas do mecanismo através do qual os complexos desempenham as suas funções, permitem obter informação sobre princípios evolucionários das interacções proteína-proteína e podem ser usadas na previsão de interacções proteína-proteína utilizando *docking*.[3,4]

1.1 Docking

O *Docking* é um método computacional para prever o modo de ligação entre duas moléculas. O problema que o *docking* molecular tenta resolver pode ser definido da seguinte forma: Dadas as coordenadas dos átomos de duas moléculas, prever o seu modo correcto de associação. Esta forma geral não entra em consideração com quaisquer dados adicionais. Na prática, os programas de *docking* utilizam outras informações para melhor prever o modo de ligação entre as duas moléculas.[5]

Há três elementos chave no *docking*, que são a representação das estruturas, a procura no espaço conformacional e o ranking das possíveis soluções. O *docking* simula a interacção entre a superfície das moléculas, portanto, o primeiro problema é a

representação da superfície das mesmas. A superfície pode ser descrita através de modelos matemáticos, tais como descritores de formas geométricas, ou através de uma grelha. Pode também envolver tratamento estático ou dinâmico da estrutura da proteína, podendo ser rígido ou flexível.[5]

O *docking* ocorre em duas fases, na primeira é executado um procedimento de pesquisa e na segunda uma função de avaliação. Os dois factores críticos na fase de pesquisa são a velocidade e a eficiência em cobrir o espaço conformacional relevante. A função de avaliação deve ser rápida o suficiente para permitir a sua aplicação num grande número de potenciais soluções e deve ser capaz de discriminar entre conformações nativas e não nativas. Esta função deve ainda ter em conta todos os componentes energéticos. O ideal é combinar os melhores algoritmos de pesquisa com as melhores funções de avaliação. Os três aspectos do *docking* estão interrelacionados, na medida em que o sistema de representação da superfície afecta o tipo de algoritmo de pesquisa e a forma de ranking das potenciais soluções.[5]

A forma mais básica para descrever a superfície de uma proteína é a representação atómica dos resíduos expostos. Porém, uma representação deste tipo só é usada quando a função de avaliação se baseia em funções de energia potencial, como por exemplo, a função de energia do CHARMM. O programa de *docking* DARWIN é um exemplo de programa que usa este tipo de função de avaliação.[6] Na abordagem que usa uma grelha para descrever a superfície das moléculas, apenas os detalhes atómicos do local de ligação do ligando e do receptor são simulados explicitamente.[7]

Na maior parte dos casos, a superfície é representada pelas suas propriedades geométricas. A base da análise da geometria das proteínas foi desenvolvida por Connoly. A designada superfície de Connoly consiste em parte da superfície de van der Waals dos átomos, que está acessível a uma esfera sonda. A superfície no seu todo está conectada por uma rede de faces convexas, côncavas e em forma de sela, que suavizam as fendas os átomos.[8] Por exemplo, no programa de *docking* ESCHER, a área acessível ao solvente proveniente da análise Conolly é cortada em fatias de 1.5Å. De seguida, cada fatia é transformada num polígono que é usado para encontrar a correspondência entre as superfícies.[9]

Alinhar a superfície de duas moléculas de forma complementar é um processo que exige que seja feita a sobreposição das superfícies sem permitir que uma molécula se sobreponha a outra. Para construir hipóteses de tais transformações é necessário alinhar conjuntos de três pontos ordenados e não colineares das duas moléculas.[10] O programa MS-DOT, criado por Connolly, distribuía pontos discretos ao longo dos três tipos de superfícies que representam a forma da molécula e para cada face calculava um ponto de

interesse e uma normal. Cada ponto pertencia a uma face de forma côncava, convexa ou toroidal.[11]

1.1.1 Problemas em *Docking*

O problema mais simples em *docking* é conhecido como "*bound*" *docking*. Neste problema, separam-se as estruturas das moléculas constituintes do complexo, o receptor e o ligando, e utiliza-se *docking* para tentar reconstruir o complexo. No entanto, a principal aplicação do *docking* é a previsão de complexos cuja estrutura seja desconhecida – *"unbound*" *docking* ou *docking* de previsão. Para este tipo de problema podem ser usadas estruturas nativas, pseudo-nativas ou estruturas obtidas por modelação. Sendo que uma estrutura nativa é a estrutura da molécula livre em solução e uma estrutura pseudo-nativa é a estrutura da molécula num complexo com uma molécula diferente da usada no *docking*.[12]

O *docking* de previsão é bastante mais complexo que o *bound docking*. A complexidade adicional provém do facto de ocorrerem alterações conformacionais na formação dos complexos. Estas mudanças conformacionais podem ser de três tipos. Podem resultar de movimentos rápidos em pequena escala, movimentos lentos em larga escala ou desordem na proteína. No último caso, a molécula pode apresentar desordem local ou global, que estabiliza da estrutura da proteína, fazendo o equilíbrio deslocar-se nessa direcção. Em geral nestes casos, o estado nativo tem um núcleo hidrofóbico ou contém cargas descompensadas no seu interior.[13]

Os movimentos na zona da interface podem ser maiores do que noutras zonas expostas da proteína. O que é consistente com o facto das proteínas apresentarem frequentemente regiões de instabilidade à volta dos locais de ligação, sendo que os estes locais apresentam zonas rígidas e zonas flexíveis. É mais provável existirem diferenças conformacionais pronunciadas entre estruturas *bound* e *unbound* do que entre duas estruturas *unbound*.[14]

1.1.2 *Docking* Proteína-Proteína

O *docking* proteína-proteína tem imensas aplicações. É particularmente importante na previsão de vias metabólicas, interacções macromoleculares e *assemblies* macromoleculares. Devido à dificuldade em determinar *assemblies* macromoleculares experimentalmente, a previsão computacional dos possíveis modos de ligação é um dos principais objectivos deste tipo de *docking*.[15]

O *docking* proteína-proteína simula o reconhecimento molecular e é a tarefa mais complexa do *docking*. Isto porque o número de graus de liberdade é enorme, não sendo

uma possibilidade fazer uma pesquisa exaustiva do espaço conformacional. É por esta razão que muitos algoritmos de *docking* tratam as proteínas como corpos rígidos. Apesar destes algoritmos geralmente não conseguirem prever a verdadeira estrutura nativa dos complexos, em alguns casos funcionam relativamente bem. O principal problema é conseguir uma função de avaliação suficientemente rápida para avaliar um grande número de soluções.[15]

No caso dos algoritmos que tratam a proteína como um corpo rígido, a flexibilidade é tipicamente conseguida fazendo a superfície variar, até mesmo permitindo penetração atómica intermolecular. Após a previsão da forma de associação, seguem-se alguns procedimentos de optimização de interacções. No entanto, devido à dificuldade no ranking de soluções, em muitos casos tal procedimento não é prático. Grande parte dos estudos de *docking* rígido assume que o local de ligação é conhecido e são poucos os programas que fazem procura em toda a superfície das moléculas. Entre os que fazem, podem-se destacar os métodos FFT (*Fast Fourier Transform-based matching*)[16], o método *Geometric Hashing*[10] e o software BiGGER[17]. Mesmo entre estes, embora inicialmente seja pesquisado todo o espaço conformacional, as soluções são de seguida filtradas. Em termos de velocidade dos algoritmos, de uma forma geral o *Geometric Hashing* leva alguns minutos a executar os cálculos, o BIGGER leva algumas horas e o FFT/FTDOCK leva alguns dias. Estes valores servem apenas como modo de comparação, sendo que variam consoante o problema e a velocidade do CPU. Enquanto que o FTDOCK é um algoritmo baseado nas transformadas de Fourier, o BIGGER utiliza uma série de regras heurísticas que reduzem o espaço de pesquisa e, como consequência, o tempo de computação.[5]

Os resultados obtidos com algoritmos de *docking* proteína-proteína são geralmente satisfatórios quando se tenta reconstruir complexos conhecidos. Porém, quando aplicados a problemas de *unbound docking*, os resultados dependem da extensão das mudanças conformacionais que ocorrem no momento da ligação. A qualidade das previsões será tanto pior, quanto maiores forem os rearranjos na estrutura das proteínas para a formação do complexo. O movimento das cadeias laterais e átomos à superfície é tratado de forma implícita por alguns algoritmos, permitindo penetração molecular dos átomos das moléculas na zona da interface. Existem outras abordagens para a resolução deste problema, uma delas consiste em fazer o *docking* usando apenas os carbonos α da cadeia principal, é portanto um *docking* de baixa resolução.[18] Esta implementação tem em conta o movimento dos átomos das cadeias laterais à superfície, tratando-o também de forma implícita. Esta abordagem tem a vantagem de aumentar a performance do processo mas tem como resultado uma deterioração da qualidade das soluções, sendo que pode ser melhorada utilizando uma função de avaliação mais eficiente. Outro problema desta abordagem é que apenas lida com movimentos das cadeias laterais e ignora movimentos da cadeia principal.[5]

Existem também algoritmos que podem ser aplicados após a realização do *docking* por outros algoritmos, para melhorar as soluções. Entre eles, o algoritmo pseudo-Browniano[19], que tem facilidade em atravessar barreiras energéticas, e o método *Internal Coordinate Mechanics*[20]. Estes já mostraram ser bastante bem sucedidos quando optimizados, porém têm o mesmo problema que a primeira estratégia apresentada: não têm em conta possíveis movimentos da cadeia principal.

Há ainda um terceiro modo de tratar a flexibilidade da proteína, que permite movimentos tipo dobradiça.[21] Nesta abordagem, os ligandos podem sofrer movimentos de translação e rotação para melhor se ligarem ao receptor. Esta abordagem é rápida e, ao contrário das anteriores, permite movimentos da cadeia principal. Porém, há semelhança de métodos já mencionados, tem o problema de considerar receptor e ligando como corpos rígidos. Além disso, necessita que sejam definidos os locais que funcionam como "dobradiças".[22]

1.1.3 Funções de Avaliação

Um algoritmo de pesquisa pode produzir um enorme número de soluções, na ordem de 10^9, portanto é necessário usar uma função de avaliação para filtrar as soluções.[17] As simulações de energia livre são métodos eficientes para este tipo de problema, no entanto, não é prático usá-las numa pesquisa de *docking*. O objectivo das funções de avaliação é discernir entre soluções nativas correctas e as incorrectas, num tempo razoável.[5] Embora algumas funções de avaliação, em alguns casos, sejam capazes de colocar soluções correctas no top 100 ou até no top 10 do ranking, na maior parte dos casos as estruturas de maior ranking são falsos positivos.

Apesar de existirem funções de avaliação sofisticadas, ainda não existe nenhum método eficiente que discrimine entre soluções correctas e falsos positivos gerados pelos algoritmos de previsão. A falta de um método eficiente para rapidamente localizar soluções correctas, nomeadamente em casos em que o local de ligação é desconhecido, é o principal obstáculo ao uso do *docking* em aplicações práticas.[23]

1.2 Informação Evolucionária

Os métodos para prever interface de proteínas podem ser baseados unicamente nas propriedades geométricas, físico-químicas e estatísticas da superfície ou podem incorporar também informação evolucionária na forma de certas medidas de conservação derivadas de alinhamentos de múltiplas sequências (MSA – *Multiple Sequence Aligments*).[24]

Ao longo do tempo ocorrem substituições de aminoácidos nas proteínas. Substituições que estabilizem a interface entre monómoros são favorecidas por selecção natural. Se uma mutação num monómero induz uma mutação noutro monómero do mesmo complexo, diz-se que as mutações estão correlacionadas. Portanto, mutações correlacionadas são tendências de posições de aminoácidos mutarem coordenadamente. Esta tendência pode ser medida analisando as correlações entre alterações em pares de posições em MSA.[1] Quanto ao significado biológico das mutações compensatórias, há evidências experimentais que apoiam a ideia de que as mutações compensatórias têm um papel estabilizador.[25]

Já foi previamente demonstrado que os métodos de previsão de interacções proteína-proteína baseados nas sequências de aminoácidos são fiáveis. No mesmo estudo foi demonstrado que pares de aminoácidos correlacionados estão significativamente mais perto uns dos outros do que pares não correlacionados e que estes podem ser usados para descriminar entre soluções correctas e incorrectas em métodos de *docking*.[1] No entanto, o processo para o cálculo de covariações é complexo e apresenta várias dificuldades, como são exemplo, o modo de seleccionar as sequências, o alinhamento das mesmas, assim como, a integração de informação de várias fontes. Portanto, este trabalho teve como objectivo automatizar este processo e testar várias possibilidades em algumas das etapas do processo.

1.2.1 Fonte de Sequências – Base de Dados UniProt

As sequências são a base de muitos estudos em bioinformática, portanto é necessário assegurar a validade das mesmas para que não comprometam os resultados. Actualmente existem várias bases de dados de sequências de proteínas, uma delas é a UniProt. A UniProt é uma fonte fiável e centralizada de informação funcional e de sequências de proteínas. Esta base de dados foi criada juntando as bases de dados Swiss-Prot, a TrEMBL e a PIR, fornecendo três níveis de bases de dados: o UniParc (UniProt *Archive*), o UniProt (UniProt *Knowledgebase*) e o UniRef (UniProt *Reference*). A UniParc fornece um conjunto estável de sequências não redundantes, armazenando todas as sequências de proteínas de acesso público. A UniProt *Knowledgebase* é uma base de dados centralizada de sequências de proteínas e informação funcional com consistência e dados fiáveis devidamente validados. A UniRef disponibiliza um conjunto de dados não redundantes com base na UniParc e UniProt *Knowledgebase* de modo a abranger toda informação de sequências com várias resoluções.[26]

Apesar da maior parte da informação desta base de dados ter origem em sequências provenientes da associação DDBJ/EMBL/GenBank, também contém sequências submetidas

directamente à UniProt, sequências que aparecem em aplicações patenteadas ou sequências provenientes da base de dados PDB. A UniParc contém uma enorme quantidade de sequências provenientes não só das bases de dados já mencionadas mas também da Ensembl, International Protein Index (IPI), RefSeq, FlyBase e WormBase. Esta combinação de fontes faz da UniParc a base de dados de sequências de proteínas, não redundante e de livre acesso, mais completa. Há ainda a acrescentar que, ao contrário de outras bases de dados, a UniProt não contém diferentes entradas para a mesma proteína mencionada em diferentes fontes bibliográficas. Nesta base de dados, os dados são combinados de modo a minimizar a redundância.[26]

1.2.2 *Multiple Sequence Aligments*

Actualmente, a comparação e alinhamento de sequências de proteínas tem um papel preponderante em bioinformática. Os alinhamentos de múltiplas sequências permitem, entre outras coisas, detectar padrões conservados ao longo da evolução e determinar relações ancestrais entre diversos organismos. Podem-se realizar dois tipos de alinhamentos: globais, que têm em conta a sequência no seu todo, ou locais, que têm em conta apenas com segmentos em que a correspondência é melhor.[27]

Os programas mais utilizados para realizar MSA são o ClustalW, ClustalX, T-Coffee, MAFFT e MUSCLE. O Clustal foi lançado em 1980, sendo o mais antigo dos programas referidos[28], assim como, o mais utilizado para realizar alinhamentos globais.[27] Os programas da série Clustal actuais derivam todos do ClustalW[29] que incorporava um novo esquema de avaliação posição-específico e um sistema de penalização de grupos de sequências representados em excesso. Desde a data em que o Clustal foi lançado, os programas da série Clustal têm sido refinados, havendo inclusivamente uma interface gráfica designada ClustalX. Os últimos melhoramentos dos programas da série Clustal incluem ajustamentos nos algoritmos de alinhamento e utilização de um novo método UPGMA (**U**nweighted **P**air **G**roup **M**ethod with **A**rithmetic Mean) para gerar árvores guia. Estes aperfeiçoamentos aumentam a velocidade do alinhamento em problemas com dezenas de milhares de sequências. Houve também a adição de um novo método iterativo para realizar o alinhamento, o que veio aumentar a exactidão do mesmo.[28]

O T-Coffee foi lançado depois do Clustal e tinha como característica efectuar alinhamentos precisos de proteínas muito divergentes, mas tinha desvantagem ter um custo computacional elevado. O programa foi sendo optimizado e, actualmente, é um programa prático adequado a problemas de média dimensão. Mais recentemente, apareceram os programas MAFFT e MUSCLE. Estes demonstraram executar alinhamentos pelo menos tão precisos quanto o Clustal e com a vantagem de serem mais rápidos e capazes de alinhar

milhares de sequências. Apesar disso, os programas da série Clustal continuam ser muito usados, especialmente no suporte *web*.[28]

Para se realizar MSA é necessário haver um conjunto de sequências homólogas para alinhar. Estas sequências podem ser obtidas utilizando o software BLAST e a sequência de interesse como *query*.

1.2.3 BLAST

Actualmente, o BLAST[30] é a ferramenta standard de procura e alinhamento. O algoritmo do BLAST está em contínuo desenvolvimento e é um dos mais citados no em biologia. Muitos investigadores usam o BLAST para fazer uma pesquisa inicial usando sequências de dados experimentais, de forma terem uma ideia das sequências em que estão a trabalhar. Porém, as potencialidades do BLAST vão muito além disto. O BLAST está longe de ser um algoritmo básico. De facto é um algoritmo muito avançado que se tornou muito popular devido à sua disponibilidade, velocidade e fidelidade. Basicamente, o BLAST procura sequências homólogas à sequência de interesse em uma ou mais bases de dados da NCBI.[31]

O BLAST é um programa *open source*, qualquer pessoa pode fazer o *download* do mesmo e alterar o seu código. Isto originou o aparecimento de derivados do BLAST, sendo que o WU-BLAST é provavelmente o mais usado. O BLAST é altamente escalável e existe num grande número de plataformas, o que torna possível usá-lo em pequenos computadores pessoais, assim como, em grandes *clusters* de computadores.[32] O BLAST pode ser usado com diferentes objectivos. Por exemplo, quando se sequencia DNA de espécies desconhecidas, o BLAST pode ser usado como ponto de partida para identificar as espécies em questão ou espécies homólogas. Pode também servir para identificar domínios numa sequência, assim como, construir árvores filogenéticas. O BLAST é também usado identificar a que cromossoma pertence uma dada sequência de DNA. Outra das suas aplicações é mapear anotações de um organismo para outro ou procurar genes comuns em duas espécies diferentes.[31]

O BLAST identifica sequências homólogas usando um método heurístico que inicialmente encontra pequenas correspondências entre duas sequências, deste modo não tem toda a sequência em conta. Depois de encontrar a primeira correspondência, tenta iniciar alinhamento local destas correspondências. O que significa que o BLAST não garante um alinhamento óptimo. Para se obter um alinhamento óptimo, deve-se usar um algoritmo de alinhamento global.[32]

1.2.4 PSI-BLAST

O BLAST possui uma variante que se designa PSI-BLAST (*Position specific iterative* BLAST) que permite correr o blastp de forma iterativa. Em cada iteração do PSI-BLAST é construído um perfil a partir do alinhamento das sequências mais semelhantes à *query*. Apenas são utilizadas as sequências que têm uma pontuação superior a um determinado limite que pode ser ajustado. O perfil referido é basicamente uma matriz com uma pontuação específica para cada posição do alinhamento (*Position Specific Scoring Matrix* – PSSM). Posições muito conservadas recebem pontuações elevadas e posições pouco conservadas recebem pontuações próximas de zero. Este perfil é usado para realizar a pesquisa do BLAST seguinte e os resultados da mesma são usados para refinar o perfil anterior. Deste modo, a sensibilidade da pesquisa vai aumentando em cada iteração.[34] No que toca à detecção de similaridades entre sequências distantes, o PSI-BLAST é muito superior ao blastp porque combina a informação de conservação de várias sequências, numa única matriz. Seja como for, é preciso ter em mente que nem todas as sequências encontradas PSI-BLAST estão relacionadas com a *query*.[35]

1.2.5 Biopython

O projecto Biopython é uma associação internacional que desenvolve ferramentas em Python para biologia computacional. Estas ferramentas são de acesso livre e incluem módulos, *scripts* e *links* de páginas *web*, tornando mais simples o uso do Python em bioinformática.[36]

Entre as múltiplas funcionalidades do Biopython pode-se destacar a capacidade de extrair dados de diversos tipos de ficheiros, transformando-os em estruturas de dados utilizadas pelo Python. Sendo fácil depois fazer iterações sobre esses dados. O Biopython possui também interfaces para programas de bioinformática conhecidos, tais como, o BLAST, o ClustalW e o EMBOSS. Contém também classes para lidar com sequências e executar operações sobre as mesmas, tais como transcrição e tradução. O Biopython contém também implementações de algoritmos de classificação, como são exemplo o k-vizinhos mais próximos, *Naive Bayes* e *Support Vector Machines*. Existem também classes para trabalhar com alinhamentos, que incluem métodos para criar e usar matrizes de substituição. Para ser mais fácil utilizar todas estas ferramentas, o Biopython possui um tutorial e documentação *wiki* on-line muito completa e de fácil compreensão.[36,37]

1.2.6 Determinação de Correlações

A detecção de correlações dá-se em dois passos. O primeiro é o MSA da sequência da proteína e das sequências dos seus homólogos. O segundo é o cálculo de covariações para todos os pares de resíduos. Uma das maiores dificuldades deste processo é a escolha da função que determina as covariações, pois, existem em grande número e têm diferenças significantes. Outro problema é que as análises de coevolução podem não lidar bem com representações desiguais de sequências, divergência evolucionária insuficiente e presença de *gaps* no MSA. Para tentar resolver estes problemas foi desenvolvido pelo laboratório de Gerstein[38] um programa que integra uma série de ferramentas e algoritmos para fazer estudos de covariação. Este programa tem à disposição uma série de filtros de pré-processamento, algoritmos de coevolução e ferramentas para análise dos resultados.[38]

As opções de pré-processamento têm como objectivo filtrar os dados de modo aumentar a sensibilidade e especificidade das funções de covariação. É possível filtrar as sequências, removendo as que contêm demasiados *gaps* ou que são muito similares umas às outras, especificando-se para isso o limite de *gaps* e o limite de similaridade. Também pode ser especificado o número mínimo de sequências no MSA. É ainda possível utilizar um sistema de pesos baseado na topologia da árvore filogenética, caso esta seja fornecida, ou baseado no método *Markov random walk*. Ambos os métodos retiram peso a sequências muito semelhantes.[38]

Depois de filtrar as sequências, também é possível filtrar posições do alinhamento que contenham demasiadas *gaps* ou que sejam muito conservadas. Sendo que, no primeiro caso, as posições não contêm informação e, no segundo caso, as posições podem influenciar de forma artificial as pontuações de coevolução. Também se podem filtrar pares de posições, pois posições que estejam próximas na sequência podem levar à produção de resultados triviais que ocultem eventos de coevolução mais importantes. Estes pares de posições podem ser filtrados, especificando o mínimo de separação entre posições. Já foi demonstrado que inserções e deleções de múltiplos resíduos podem criar falsos positivos. Por isso, é possível filtrar pares de posições que participem nas mesmas *gaps* em muitas sequências.[38]

Existem outras opções disponíveis como agrupamento de resíduos similares num alfabeto de menor dimensão com o objectivo de aumentar a sensibilidade. Já foi também demonstrado que as *gaps* podem fornecer sinais evolucionários importantes, portanto, é possível tratar as *gaps* como ruído ou como um resíduo.[38]

Em algumas proteínas, resíduos correlacionados tendem a estar perto uns dos outros. O que sugere que uma mutação num resíduo pode criar instabilidade que é de certa forma compensada com uma mutação num resíduo próximo. Por isso, são disponibilizadas

funções para analisar as pontuações de coevolução em relação às distâncias entre resíduos, assim como, técnicas de aprendizagem automática para avaliar a eficiência de várias funções de coevolução. Também é fornecido um sistema para avaliar a significância das pontuações, mas apenas se o programa for utilizado localmente.[38]

1.2.7 Métodos de Determinação de Covariações

Existem muitos métodos para determinar covariações, alguns seguem estratégias muito diferentes e outros são apenas variações de métodos criados anteriormente. De seguida, são apresentados alguns deles, só são referidos aqueles que são disponibilizados pelo software referido na secção anterior.[38]

1.2.7.1 Correlação de Pearson

Este método procura co-ocorrência de mutações de similaridades comparáveis e baseia-se na definição matemática de coeficiente de correlação *r*. Este método utiliza uma matriz de similaridade que fornece uma pontuação para cada mutação. Esta pontuação indica quão radical ou conservada é a mutação em relação à alteração das propriedades físico-químicas dos aminoácidos. É esperado que uma mutação radical num resíduo seja acompanhada por uma mudança radical na posição complementar. Para cada posição *i*, todos os possíveis pares de sequências *k, l* são comparados, havendo portanto N^2 comparações para cada posição. A pontuação é atribuída a pares de posições de correlação, comparando as pontuações de similaridade da seguinte forma:[39]

$$CM_{ij} = r_{ij} = \frac{1}{N^2}\sum_{kl}\frac{W_{kl}(s_{ikl} - \langle s_i \rangle)(s_{jkl} - \langle s_j \rangle)}{\sigma_i \sigma_j} \qquad (1)$$

Na expressão anterior, $\langle S_i \rangle$ é a pontuação de similaridade média de N^2 comparações na posição *i*; S_{ikl} é a pontuação de similaridade entre o resíduo na posição *i* na sequência *k* e o resíduo *i* na sequência *l*; σ_i é o desvio padrão de N^2 similaridades na posição *i*; W_{kl} é a fracção de posições não idênticas do MSA nas sequências *k, l* normalizadas. O objectivo é atribuir menos peso a informação de sequências similares. Além disso, colunas totalmente conservadas com um desvio padrãode zero são removidas da análise.[39]

Existem algumas modificações ao método básico de correlação de Pearson. Uma delas é anular a utilização de pesos e outra é não comparar a identidade dos resíduos. [39]

O software já mencionado disponibiliza várias funções de correlação opcionais. Também disponibiliza várias matrizes de similaridade para diferentes propriedades dos aminoácidos, tais como, volume, pI e hidrofobicidade.[38]

1.2.7.2 *Observed Minus Expected Squared* – OMES

Este método foi criado com base no teste estatístico não paramétrico chi-quadrado. O que ele faz é comparar a co-ocorrência de cada dois resíduos em cada duas colunas com a sua co-ocorrência esperada [Eq.2 (B)]. A co-ocorrência esperada é baseada na ocorrência de cada resíduo na coluna, assumindo que não há dependência entre os dois resíduos distribuídos nas duas colunas [Eq.2 (A)].[39]

$$(\mathrm{A}) \quad N_{ex} = \frac{N_{\mathrm{ex}} N_{yj}}{N_{\mathrm{valid}}} \quad (\mathrm{B}) \quad r_{ij} = \sum_{l}^{L} \frac{(N_{\mathrm{obs}} - N_{\mathrm{ex}})^2}{N_{\mathrm{valid}}} \qquad (2)$$

Nas equações anteriores, *L* é o tamanho da lista de pares distintos de resíduos nas colunas *i* e *j*; N_{valid} é a sequência sem *gaps* nas colunas *i*, *j*; N_{obs} é o número de vezes que cada par distinto aparece; N_{ex} é o número de vezes que se espera que os resíduos *x* e *y* apareçam nas colunas *i* e *j*, respectivamente, dadas as suas ocorrências singulares nas colunas *i* e *j*; N_{xi} é o número de vezes que o resíduo *x* aparece na coluna *i*; N_{yj} é o número de vezes que o resíduo y aparece na coluna *j*.[39]

1.2.7.3 Informação Mútua

Neste método, cada posição é de facto uma coluna ordenada de resíduos. Para executar o cálculo para cada par é construída uma matriz triangular com dimensões *n* por *n*-1, sendo *n* o comprimento da sequência. De seguida, é atribuído um valor a cada célula da matriz e é apresentada uma lista dos valores como *output*.[40]

O método de informação mútua (MI – *Mutual Information*), muito utilizado em bioinformática, ganhou muita importância com o desenvolvimento da teoria da informação. Este mede a quantidade de informação que uma variável aleatória contém acerca de outra variável aleatória:[40]

$$MI_{ij} = \sum_{xy} P(x,y) \log \frac{P(x,y)}{P(x)P(y)}, \qquad (3)$$

P(x) é a probabilidade de encontrar o resíduo *x* na coluna *i*; *P(y)* é a probabilidade de encontrar o resíduo *y* na coluna *j* e *P(x,y)* é a probabilidade de encontrar os resíduos *x,y* nas colunas *i,j* respectivamente. As probabilidades são calculadas a partir da distribuição de aminoácidos nas respectivas colunas do MSA.[39]

A estatística MI escala até um máximo de 1. O valor de informação atinge 1 quando se verificam 3 condições. Em primeiro lugar, tem de haver uma covariação perfeita entre os resíduos presentes nas duas posições. Em segundo lugar, todos os 20 aminoácidos têm que aparecer nas duas posições. E em terceiro lugar, todos os aminoácidos têm que estar presentes na mesma frequência. Destas propriedades, apenas a primeira é desejável para medir coevolução, as outras duas podem levar ao aparecimento de falsos positivos.[40]

São disponibilizadas várias opções de normalização que permitem lidar melhor com efeitos de amostras de tamanho reduzido e efeitos de influência filogenética.[39]

1.2.7.4 *Quartets*

Este método usa o conceito base da correlação de Pearson mas não utiliza a matriz de similaridade das propriedades físico-químicas. Uma das razões para negligenciar o uso desta matriz é o facto de em algumas famílias de proteínas, a alteração de aminoácidos similares em certas posições, induzir alterações dramáticas noutras posições.[39]

O método *quartets* tem este nome porque utiliza conjuntos de quatro resíduos compostos por dois pares de aminoácidos de duas colunas (X_{ik}, Y_{il}, X'_{jk}, Y'_{jl}). Para cada par de colunas é construída uma matriz 20 x 20 e é somada a contribuição de todos os conjuntos de resíduos. Um conjunto contribui 1 se cumprir todas as condições da equação seguinte, caso contrário contribui 0:[39]

$$\left\{ \begin{array}{l} [(P_{ix}{*}P_{jy} \gg P_{iy}{*}P_{jx}) \text{ and } ((P_{ix} > D_{\min}) \text{ or } (P_{jy} > D_{min}))] \text{ or} \\ [(P_{ix}{*}P_{jy} \ll P_{iy}{*}P_{jx}) \text{ and } ((P_{iy} > D_{\min}) \text{ or } (P_{jx} > D_{min}))] \end{array} \right\}$$
$$\text{and} \left\{ \left(\frac{P_{ix}{*}P_{jy}}{P_{iy}{*}P_{jx}} > D^{Q}_{\min} \right) \text{ or } \left(\frac{P_{iy}{*}P_{jx}}{P_{ix}{*}P_{jy}} > D^{Q}_{\min} \right) \right\} \quad (4)$$

$D^{Q}_{\min}$ é o número de sequências que evidenciam correlação de um par de aminoácidos, dividido pelo número de sequências evidenciam o contrário e D_{min} é o número absoluto de sequências que evidenciam correlação.[39]

1.2.7.5 *Statistical Coupling Analysis* – SCA

O método SCA é baseado na perturbação de um MSA. O resíduo mais presente na coluna *j* define um sub-alinhamento. O sub-alinhamento é composto apenas por sequências

em que esse resíduo aparece na coluna *j*. A pontuação da mutação correlacionada é expressada por um termo energético, $\Delta\Delta G_{i,j}$, que expressa a diferença entre o parâmetro ΔG_j do alinhamento completo e do sub-alinhamento.[39]

$$(\mathrm{A}) \quad \Delta G_i^{\mathrm{stat}} = kT * \sqrt{\sum_x \left(\ln \frac{P_i^x}{P_{\mathrm{MSA}}^x} \right)^2} \qquad (5)$$

$$(\mathrm{B}) \Delta\Delta G_{ij}^{\mathrm{stat}} = kT * \sqrt{\sum_x \left(\ln \frac{P_{i|\delta j}^x}{P_{\mathrm{MSA|\delta j}}^x} - \ln \frac{P_i^x}{P_{\mathrm{MSA}}^x} \right)^2}$$

$$(\mathrm{C}) \quad \Delta\Delta G_{ij} = \sqrt{\sum_x (\ln P_{i|\delta j}^x - P_i^x)^2}$$

P^x_i é a probabilidade de encontrar um aminoácido *x* na coluna *i*; P^x_{MSA} é a probabilidade de encontrar o aminoácido *x* no MSA; $P^x_{i|\delta j}$ é a probabilidade de encontrar o aminoácido *x* na coluna *i* do sub-alinhamento perturbado em relação à coluna *j*.[39]

O método SCA foi introduzido na forma da equação (A), sendo que a constante *kT* só era utilizada para a equação aparecer como termo energético e não tinha qualquer efeito no cálculo da correlação. Além disso, como a ligação entre a pontuação SCA e o termo energético foi contestada, não é apropriado utilizar este termo. Depois disso, o termo P_{MSA} foi também removido, utilizando-se agora na forma da equação (C).[39]

1.2.7.6 *Perturbation*, *Explicit Likelihood of Subset Covariation* – ELSC

O método de perturbação em essência é similar ao SCA. A principal diferença entre os dois é o modo de medir os desvios da composição aminoacídica entre o sub-alinhamento e o alinhamento completo. ELSC mede quantas possibilidades de sub-alinhamentos de tamanho *n* teriam a composição encontrada no alinhamento *j*.[39]

$$\prod_{x=1}^{20} \frac{\binom{N_{xj}}{n_{xj}}}{\binom{N_{xj}}{m_{xj}}} \qquad (6)$$

$N_{x,j}$ é o número de resíduos do tipo *x* na posição *j* no MSA não perturbado; $n_{x,j}$ é o número de resíduos do tipo *x* na posição *j* no sub-alinhamento definido pela perturbação na coluna *i*; m_{xj} é o número de resíduos do tipo *x* na posição *j* no sub-alinhamento idealizado.[39,41]

2 Materiais e Métodos

Este trabalho teve como finalidade utilizar informação de coevolução para determinar pontos de contacto ou zonas de contacto em complexos proteicos transientes. Esta informação pode ser útil para resolver o problema do *docking* molecular, permitindo filtrar soluções geradas pelo algoritmo de pesquisa ou restringindo as soluções geradas pelo mesmo.

O procedimento deste trabalho foi concebido com base em outros trabalhos que mostraram ser eficientes.[40,42] O trabalho foi desenvolvido de forma a ser o mais automatizado possível para que o utilizador interviesse apenas quando fosse imperativo e para que o mesmo não necessitasse de ter muitos conhecimentos de informática. No entanto, devido à complexidade do problema, não foi conseguido automatizar todas as etapas. Neste trabalho foi utilizado o Biopython sempre que possível por facilitar a leitura de ficheiros fasta, a execução do psi-blast e do ClustalW, entre outros.

Este método parte das sequências das subunidades dos complexos e termina com a obtenção de informação de coevolução. Em traços gerais, estes são os passos:

- a sequência de cada subunidade é utilizada como *query* no blastp ou psi-blast;
- o resultado do psi-blast é um conjunto de sequências homólogas à *query*;
- cada conjunto de sequências é alinhado utilizando o ClustalW;
- junta-se o alinhamento das subunidades do mesmo complexo;
- executam-se os cálculos de covariação para cada complexo.

Foi criado um módulo separado para cada fase do etapa, sendo possível utilizar cada um individualmente ou em conjunto de modo sequencial, executando um *script batch*. Neste trabalho, as ferramentas foram criadas de modo a só utilizarem informação local sendo unicamente necessário ter o devido software instalado (Python, Biopython, BLAST, etc.), um ficheiro fasta que funciona como “base de dados”, um ficheiro com os identificadores dos complexos que se pretendem estudar e um ficheiro que contenha as sequências das proteínas que compõem os complexos. O ficheiro que funciona como “base de dados” pode conter sequências retiradas de uma ou mais bases de dados da *web* e pode ser concebido pelo utilizador com sequências de interesse para o estudo que está a realizar. Além destes ficheiros, existe outro onde o utilizador especifica que tipo de pesquisa pretende usar, blastp ou psi-blast.

À medida que o procedimento foi construído, foi sendo testado num conjunto de sequências de complexos cujas estruturas são conhecidas. Portanto, a primeira fase foi encontrar e seleccionar estes complexos proteicos.

2.1 Criação do Conjunto de Teste

Pesquisar numa base de dados complexos proteicos que possam ser utilizados nesta situação é uma tarefa complexa. Em estudos já realizados, a pesquisa e filtração dos dados teve de ser feita manualmente.[42] No estudo referenciado foi usado um conjunto de dados que já tinha sido construído pelos autores num trabalho anterior, no qual tinham separado manualmente complexos transientes e complexos estáveis.[2] Este conjunto foi filtrado de forma a não ter informação redundante e foi constituído por 212 complexos transientes e 115 complexos estáveis ou obrigatórios. Dois complexos foram considerados não redundantes se os domínios em contacto tivessem uma classificação estrutural diferente. É de notar que a mesma proteína pode estar representada múltiplas vezes como parte de diferentes complexos não redundantes.[42] Decidiu-se portanto utilizar este conjunto de dados no trabalho realizado por ter várias vantagens. Em primeiro lugar, evitou a construção de um conjunto de dados de início, poupando tempo. Em segundo lugar, a classificação dos complexos em transientes e obrigatórios já foi validada, permitindo fazer estudos comparativos. Como o objectivo era aplicar o método a complexos transientes, foram seleccionados alguns destes complexos. Na primeira fase da selecção foram seleccionados apenas os complexos diméricos, pondo de lado todos os complexos com mais de duas subunidades. Posteriormente, este conjunto de complexos foi filtrado de novo.

2.2 Pesquisa de Sequências

Para se obterem as sequências para realizar os MSAs foi utilizado o psi-blast. Este foi utilizado para procurar sequências homólogas às sequências de cada uma das subunidades dos complexos seleccionados. A base de dados utilizada para procurar as sequências foi a UniProt[26]. Na pesquisa das sequências foram usadas duas iterações e os parâmetro *default* do blastp com excepção do *e-value* que foi alterado para $1x10^{-6}$. Foi usado este valor de *e-value* porque, para o problema em questão, as sequências a utilizar têm de ser semelhantes, mas por outro lado têm de ser diferentes o suficiente para apresentarem covariações. Após a pesquisa pelo psi-blast, obteve-se um conjunto de sequências para cada sequência inicial. Nesta fase houve outra filtragem, sendo que foram postos de lado complexos cujas subunidades possuíssem conjuntos de sequências pequenos, inutilizáveis em estudos de coevolução. Portanto, foram eliminados todos os complexos que tivessem pelo menos uma subunidade cujo conjunto de sequências tivesse dimensão inferior a dez. Dos complexos que restaram, foram seleccionados 8 complexos aleatoriamente, ou seja, 16 conjuntos de sequências, para utilizar no restante trabalho.

Tabela 1 Complexos proteicos do conjunto de teste.

PDB ID	Descrição
1buh[43]	CDK2 kinase complex with cell cycle-regulatory protein CKSHS1
1euv[44]	ULP1 protease domain in complex with SMT3
1grn[45]	CDC42/CDC42GAP/ALF3 complex
1h59[46]	Complex of IGFBP-5 with IGF-I
1itb[47]	Type-1 interleukin-1 receptor complexed with interleukin-1 beta
1m10[48]	Complex of glycoprotein Ib alpha and the von Willebrand Factor A1 Domain
1ycs[49]	P53-53BP2 complex
1zbd[50]	Small G Protein RAB3A complexed with the efector domain of Rabphilin-3A

Estes conjuntos de sequências foram verificados manualmente e constatou-se que algumas sequências se encontravam em duplicado, portanto as sequências foram filtradas de modo a eliminar a redundância.

Portanto, existe um módulo que executa o que foi referenciado nesta secção. Além disso, como o *output* do blastp e psi-blast é gerado em formato xml, este módulo também o converte o para formato fasta.

2.3 Selecção das Sequências

O que se segue à obtenção dos conjuntos de sequências é a realização dos MSAs, no entanto, nem todas as sequências do conjunto devem ser utilizadas. Portanto, antes de alinhar as sequências foi necessário seleccioná-las. Este processo de selecção foi feito por outro módulo que lê as sequências do ficheiro fasta gerado anteriormente, filtra-as e guarda-as noutro ficheiro. Uma das razões que torna necessária a selecção das sequências é que, para realizar estes cálculos de covariações, as sequências das subunidades dos complexos têm de estar organizadas de acordo com a espécie. Ou seja, cada sequência no conjunto de sequências da subunidade A tem que emparelhar com uma sequência da mesma espécie do conjunto de sequências da subunidade B. O que acontece é que nem sempre os conjuntos de sequências de A e B possuem sequências das mesmas espécies, o que significa que há sequências que não têm par, logo, não podem ser usadas em estudos deste tipo. Portanto, estas sequências tiveram de ser identificadas e eliminadas.

Para executar a tarefa de selecção e emparelhamento de sequências foi criada uma classe para armazenar a informação relativa às sequências. Até esta etapa, a informação relativa às sequências tinha sido tratada, utilizando a classe SeqRecord do Biopython. Porém, esta classe não permite aceder directamente a parte da informação necessária para resolver o problema referido. A informação sobre a espécie, e até sobre o nome da proteína, está contida numa *string* que é o atributo *description* da classe SeqRecord. Para se aceder directamente a esta informação e para se conseguir filtrar as sequências foi necessário implementar uma nova classe, semelhante à SeqRecord mas que armazenasse mais informação. Foi criada a classe SeqPlus que possui os seguintes atributos:

Tabela 2 Atributos da classe SeqPlus.

Atributo	Descrição
seqRec	O objecto SeqRecord de onde é extraída a informação
description	O atributo *description* do objecto SeqRecord original
id	O atributo *id* do objecto SeqRecord original
theSeq	A sequência da proteína
specie	A espécie a que pertence a proteína
proteinName	O nome da proteína
proteinId	O identificador da proteína na base de dados UniProt
subunit	A subunidade da proteína

A informação contida nestes novos atributos é extraída do atributo *description* do objecto SeqRecord original através de métodos desenvolvidos para isso. Só é garantido o correcto funcionamento destes métodos para atributos *description* de sequências provenientes da base de dados UniProt.

Os atributos desta nova classe permitem aceder directamente a informação importante que pode ter várias aplicações. Neste caso particular, o facto de se poder aceder directamente à espécie é importante. Foi também criada outra classe designada Subunit que guarda todas as sequências de uma subunidade, num dicionário cujas chaves são os nomes das espécies. Assim, é possível procurar sequências da mesma espécie em subunidades diferentes com facilidade.

O número de sequências nos conjuntos varia e pode ser muito diferente em subunidades do mesmo complexo. Por exemplo, o complexo 1buh é o da cinase CDK2 com a proteína CKsHs1. O que acontece neste caso e em casos semelhante é que para a CKsHs1 existem poucas sequências por espécie. No entanto, para no conjunto de sequências da CDK2 existem muitas sequências por espécie, pois há outras cinases que partilham motivos com esta. Aqui a questão foi como seleccionar as sequências, do conjunto

de sequências da CDK2, para emparelhar com as sequências da CKsHs1. Existem vários critérios que podem ser usados para seleccionar as sequências. Podem ser seleccionadas todas as sequências da proteína, pode ser seleccionada a sequência melhor de cada espécie, ou uma sequência ao acaso de cada espécie, entre outras. Qualquer que seja o critério, há sempre um compromisso. Um critério muito alargado pode incluir sequências que vão ocultar a covariação e um critério muito rigoroso pode reduzir demasiado o número de sequências, que é importante no cálculo de correlações. Neste caso particular, optou-se por testar dois tipos de selecção, seleccionar o máximo de sequências do conjunto de CDK2 e seleccionar apenas uma sequência da CDK2 de cada espécie. A primeira forma de selecção mencionada implica que se repitam sequências do conjunto de sequências da CKsHs1, pois este é de menor dimensão. Esta opção tem o problema de poder alterar a frequência relativa dos aminoácidos e de no fundo se estar a assumir que certas proteínas formam complexos que podem não existir na realidade, pois se está a utilizar sequências que são de outras proteínas. O segundo modo de selecção tem o problema de implicar que se comparem nomes de proteínas. Apesar da classe já mencionada permitir aceder directamente ao nome das proteínas, este nome é representado por uma *string* que não pode ser comparada directamente porque sequências da mesma proteína mas de espécies diferentes podem ter nomenclaturas ligeiramente diferentes, mesmo se sequências forem provenientes da mesma base de dados. Portanto, para contornar este problema utilizou-se um sistema que utiliza uma palavra-chave. Caso a palavra-chave esteja presente no nome da proteína, esta é seleccionada, caso contrário, não é. Neste caso, definindo a palavra passe como “CDK2” permite seleccionar apenas sequências de proteínas CDK2. A escolha da palavra-chave tem que ser definida pelo utilizador e é específica de cada conjunto.

2.4 Alinhamento dos Conjuntos de Sequências

Após ter sido feito o emparelhamento das sequências, obteve-se um conjunto de sequências para cada complexo. Cada sequência neste conjunto corresponde à junção das sequências das duas subunidades. Nesta fase, cada conjunto de sequências foi alinhado utilizando o ClustalW com os parâmetros *default*. Existe um módulo que executa os alinhamentos e converte o *output* do ClustalW para formato fasta.

2.5 Cálculos de Determinação de Covariações

Os cálculos de covariações foram executados utilizando o programa já mencionado na secção *1.2.5*. Este programa, embora seja fácil de utilizar no suporte *web*, quando utilizado localmente é de utilização um pouco mais complexa, sendo que, a sua

documentação não é pormenorizada e para que possa ser utilizado tem que ser instalado outro software denominado *apache ant*.[51] Este software serve para compilar o *source code* do programa anterior e utiliza a linguagem de programação Java. Depois de instalado o programa, o problema foi determinar em que forma específica tinham de estar os ficheiros de *input*. A pouca documentação disponível não contém qualquer informação sobre este aspecto. O problema foi solucionado submetendo um cálculo na interface *web* e utilizando o ficheiro de *input* que é fornecido no final dos cálculos. Este ficheiro foi depois modificado para ser utilizado localmente com diversos algoritmos para determinar de covariações. A execução deste programa implica ainda que seja utilizada a linha de comandos e que sejam definidas três variáveis locais antes de ser executado, portanto, foi feito um *script batch* que realiza estas operações. Desta forma, é possível utilizar o programa sem recorrer directamente à linha de comandos, o que pode ser vantajoso para alguns utilizadores.

Foram feitos testes utilizando vários algoritmos implementados no programa: informação mútua, ELSC, *quartets* e método de correlação de Pearson. No que toca a opções de filtração de sequências foram utilizados os seguintes valores:

Tabela 3 Filtros utilizados na selecção de sequências para calcular covarições.

Filtro	Valor
Máximo de similaridade entre sequências	0.9
Nº mínimo de sequências	10
Nº máximo de *gaps* por posição	0.1
Máximo carácter mais frequente	0.9
Separação mínima entre posições	3

Foi criado um módulo que cria ficheiros específicos para que cada um dos conjuntos de sequências possa ser utilizado pelo programa que determina covariações. São criados dois ficheiros para cada conjunto de sequências e além disso é copiado o ficheiro fasta com o MSA para o directório apropriado. Seguidamente o mesmo módulo executa os cálculos de covariação para todos os conjuntos de sequências.

2.6 Extracção da Informação de Covarições

Após a realização dos cálculos de covarições é gerado um ficheiro de *output* que contem um valor para cada par de aminoácidos no complexo. O valor varia de 0 a 1 e quanto maior for, maior é a probabilidade de esses aminoácidos terem sofrido coevolução. Para se conseguir extrair alguma informação do ficheiro de *output* foram desenvolvidos alguns métodos que extraem as soluções com maior valor de covariação. Neste caso,

começou-se por analisar as dez soluções com maiores valores de covariação. Além disso, foram filtradas as soluções que correspondiam a relações entre aminoácidos da mesma subunidade para ficarem apenas as soluções que correspondiam a relações entre aminoácidos de subunidades diferentes, que eram as que interessavam para o problema em questão. De seguida, as posições dos aminoácidos foram corrigidas. Em muitos casos as proteínas nos ficheiros pdb não se encontram completas, portanto foi necessário fazer a conversão das posições dos aminoácidos do *output* do programa que determina as covariações, para as posições dos mesmos no ficherio pdb. Para tal, as duas sequências foram alinhadas utilizando uma vez mais o ClustalW.

De seguida, foi calculada a distância entre resíduos para todos os pares dos mesmos. Por fim, foi criado um ficheiro contendo o símbolo e posição de cada par de resíduos, assim como, a sua pontuação de covariação e a distância entre os mesmos.

2.7 Visualização

Após a extracção das soluções foi utilizado o software DS Visualizer[52] para visualizar os aminoácidos na proteína e verificar se estes se encontravam próximos ou em contacto um com outro. Caso se encontrassem, significaria que este método permitiria prever pontos de contacto e zonas de interface.

3 Resultados e Discussão

Foram realizados vários testes sobre as proteínas dos 16 conjuntos de sequências, correspondentes aos oito complexos. Recorreu-se vários algoritmos para determinar as covariações, com a finalidade de verificar se os resultados dos algoritmos seriam concordantes e avaliar a eficiência dos mesmos. No que toca à determinação de correlações entre subunidades de um complexo, os testes não chegaram a ser completados porque surgiram alguns problemas logo de início.

Começou-se por utilizar o complexo 1buh, mencionado anteriormente, e o algoritmo de informação mútua para determinar as covariações. Depois, foram extraídas as 10 melhores soluções e foi utilizado o software já referido para observar os aminoácidos em questão na estrutura da proteína. Aí verificou-se que as soluções não correspondiam a aminoácidos próximos. Foram testados os dois métodos de selecção de sequências e nenhum deles detectou um único ponto de contacto. Posteriormente, testaram-se outros complexos e verificou-se o mesmo: não existia um único par de resíduos que estivessem próximos. Quando se testaram outros algoritmos para determinar covariações, tais como o *quartets* e método de correlação de Pearson, o melhor que se obteve foi um par de resíduos

em contacto no top 10. Portanto, o problema podia ser causado por baixa qualidade dos MSAs, assim como do processo de selecção de sequências. O problema também poderia ser resultado dos cálculos para determinar covariações estarem a ser feitos com as sequências completas das duas subunidades, em vez de se utilizar apenas os aminoácidos à superfície da proteína. Além disso, este processo calcula relações não só entre aminoácidos de subunidades diferentes, mas também, entre aminoácidos da mesma subunidade. Estes dois últimos factores contribuem para que correlações entre aminoácidos à superfície das duas subunidades apareçam pouco evidenciadas.

A maior parte dos estudos de coevolução realizados são de determinação de correlações intramoleculares. O facto de se estar a determinar correlações entre duas subunidades, torna o problema mais complexo. Portanto, nesta fase decidiu-se simplificar o problema, utilizando apenas uma subunidade para executar os cálculos de covariação. Deste modo, efectuaram-se cálculos para determinar apenas correlações intramoleculares, com o objectivo de eliminar hipóteses que pudessem estar na origem dos resultados acima referidos. Além disso, o top 10 das soluções foi aumentado para top 20 e o método de selecção de sequências também foi alterado. O método de selecção anterior requeria a utilização de uma palavra-chave escolhida pelo utilizador, o que não era prático e se mostrou pouco eficiente. Portanto, adoptou-se outro método mais simples e automático que consiste simplesmente na escolha na sequência mais semelhante de cada espécie.

Nesta nova fase de testes foram utilizadas as proteínas pertencentes aos oito complexos seleccionados anteriormente. O que significa que se utilizaram 16 proteínas. Fizeram-se os mesmos testes, o psi-blast para efectuar a procura inicial de sequências e foram utilizados os quatro métodos para determinar covariações já mencionados anteriormente. O conjunto de proteínas, apesar de inicialmente ser composto por 16 proteínas, ficou reduzido a 12. Devido à filtração de sequências que é feita pelo programa que calcula as covariações, alguns dos conjuntos de sequências ficaram reduzidos a menos de 10 sequências, portanto não foram usados nos cálculos.

Após a realização dos cálculos de covariação, verificou-se que a simplificação do problema, assim como as alterações efectuadas permitiram detectar mais pontos de contacto. Os resultados apresentados são bem mais consistentes do que os obtidos anteriormente, em que no máximo se obtia um ponto de contacto. Em baixo está uma tabela com o número de contactos no top 20, por método de determinação de covariações para cada complexo.

Tabela 4 Número de aminoácidos em contacto por método de determinação de correlações.

Proteínas	MI	C. Pearson	ELSC	*Quartets*
1BUH_A	0	0	0	0
1BUH_B	2	2	0	2
1GRN_A	3	2	2	1
1GRN_B	0	2	1	0
1H59_A	1	1	0	0
1ITB_A	1	2	0	1
1ITB_B	1	3	1	1
1M10_A	2	5	0	1
1M10_B	0	15	3	0
1YCS_A	2	0	1	0
1YCS_B	0	3	3	0
1ZBD_A	0	1	2	0

Não há nenhum método que detecte contactos em todos os casos mas o método de correlação de Pearson detecta contactos em dez das vinte proteínas. Além disso, de uma maneira geral, é o método que detecta mais contactos. Apesar de existirem muitos falsos positivos em quase todos os casos, no caso particular da proteína 1M10_B, o método de correlação de Pearson apresentou uma grande eficiência, apresentando apenas 5 falsos positivos.

Em alguns casos, como são exemplo a proteína 1BUH_A e a 1H59_A, foram detectados poucos contactos ou até mesmo nenhuns. Isto deve-se ao facto de terem sido seleccionadas demasiadas sequências, algumas das quais já são muito diferentes da proteína e tornam impossível obter dados de covariação. Portanto, o principal ponto a melhorar é a selecção das sequências.

Este procedimento apesar de se encontrar numa fase preliminar, foi seguidamente testado em alguns complexos, para tentar determinar covariações intermoleculares, que era o objectivo do trabalho. Foram usados apenas os complexos cujas subunidades estivessem ambas presentes nos resultados finais dos testes anteriores. Ou seja, foram utilizados os complexos 1BUH, 1GRN, 1ITB, 1M10 e 1YCS. Os resultados foram filtrados de forma a se obterem apenas valores de covariação para pares de resíduos pertencentes a aminoácidos de subunidades diferentes. Os resultados obtidos foram os seguintes:

Tabela 5 Número de aminoácidos em contacto por método de determinação de correlações.

Proteínas	MI	C. Pearson	ELSC	*Quartets*
1BUH	1	0	0	1
1GRN	2	2	1	0
1ITB	1	2	0	1
1M10	1	3	1	0
1YCS	0	1	1	0

Nestes testes, o método de correlação de Pearson mostrou ser o mais eficiente na maior parte dos casos mas mesmo assim, o número de contactos foi em todos os casos bastante baixo, variando geralmente entre 0 e 2. Apenas num dos casos foram detectados 3 pontos de contacto. No entanto, há que ter em conta que foram considerados todos os pares de resíduos entre as duas subunidades. Para se tentar obter melhores resultados pode-se considerar apenas resíduos à superfície das subunidades. Desta forma, apareceram mais correlações de aminoácidos à superfície no top 20 de soluções pois é filtrado um grande número de falsos positivos.

Apesar dos pontos de contacto variarem entre 0 e 2 na maioria dos casos, como o objectivo deste trabalho é aplicar este procedimento no *docking* para diminuir o espaço de soluções a explorar, este método tem potencialidades. Mesmo que em 20 soluções, uma ou duas sejam verdadeiras, já é possível diminuir muito o número de soluções geradas pelo algoritmo de pesquisa, assim como filtrar soluções já geradas pelo mesmo.

Para se poder passar à aplicação deste tipo de procedimento à detecção de pontos de contacto para utilizar em *docking*, é necessário refinar os métodos, testando-os primeiro em casos simples, como os apresentados anteriormente. A fase mais crítica deste tipo de método é a selecção de sequências, assim como, os MSAs, portanto tentar melhorar ambos é um bom ponto de partida. Além disso, o conjunto de teste utilizado foi perdendo dimensão à medida que os vários passos foram executados, portanto, este conjunto deve ser alargado para testar a robustez do procedimento. Podem-se ainda variar os parâmetros utilizados para filtrar as sequências para os cálculos de covariações, assim como, fazer variar o valor limite do *e-value* e outros parâmetros na pesquisa inicial de sequências para tornar a pesquisa mais rigorosa.

Apesar deste método mostrar ter potencialidades, é precoce utilizar os dados produzidos neste trabalho para filtrar soluções de *docking* ou restringir as soluções geradas pelo mesmo porque ainda há vários aspectos a melhorar.

4 Conclusão

Durante a fase de construção do conjunto de teste ficou evidente que métodos deste tipo não são generalistas, não podem ser aplicados a qualquer proteína, só são aplicáveis em situações em que existam sequências homólogas suficientes para realizar os cálculos de covariações. No que toca ao desenvolvimento do método, há ainda a referir que o Biopython foi uma ferramenta base na sua concepção e que facilita a utilização dos algoritmos de pesquisa e alinhamento de sequências. Esta ferramenta, embora não tenha permitido responder aos problemas mais complexos e específicos do trabalho, foi um bom ponto de partida.

O método utilizado ainda está em fase de desenvolvimento e ainda tem de ser melhorado para se aplicar à determinação coevolução entre proteínas e se obter resultados robustos. Apesar do trabalho mostrar que tem potencialidades, a selecção das sequências para realizar o MSA tem de ser melhorada. Noutros estudos que mostraram ter bons resultados, cada sequência foi analisada e seleccionada manual e individualmente. O que significa que a complexidade do problema não permite que esta selecção seja baseada em critérios tão simples como os utilizados.

Apesar do método na globalidade ainda ter de ser aperfeiçoado, devido à forma como foi planeado e concebido, permite que módulos individuais e independentes possam ser utilizados. Por exemplo, o utilizador pode facilmente executar o BLAST sobre uma base de dados diferente das disponibilizadas no suporte *web*. Além disso, pode executar várias pesquisas consecutivas do BLAST para várias proteínas, bastando para isso colocar o identificador das sequências no ficheiro designado para isso. Da mesma maneira, o utilizador pode executar vários MSAs consecutivos utilizando o ClustalW. Outro módulo que pode ser usado independentemente é o que lê e filtra os resultados do programa que determina covariações. O utilizador pode executar um cálculo localmente, ou utilizando o suporte *web*, e utilizar este módulo para ler o ficheiro e lhe retornar os N melhores resultados. Além dos módulos independentes, pode ainda ser usada a classe SeqPlus individual, ou juntamente com a classe Subunit. Apesar destas classes terem sido criadas de forma a facilitar a resolução deste problema particular, elas têm mais potencialidades e, à semelhança da classe SeqRecord do Biopython, podem ser utilizadas na análise e extracção de informação de conjuntos de sequências, desde que estes provenham da base de dados UniProt ou cujas descrições sigam as mesmas regras desta base de dados.

5 Bibliografia

1. Pazos F, Helmer-Citterich M, Ausiello G, Valencia A. Correlated mutations contain information about protein-protein interaction. *Journal of molecular biology*. 1997;271(4):511-23.

2. Mintseris J, Weng Z. Atomic contact vectors in protein-protein recognition. *Proteins*. 2003;53(3):629-39.

3. Bordner AJ, Gorin AA. Comprehensive inventory of protein complexes in the Protein Data Bank from consistent classification of interfaces. *BMC bioinformatics*. 2008;9:234.

4. Kundrotas PJ, Alexov E. PROTCOM: searchable database of protein complexes enhanced with domain–domain structures. *Database*. 2007;35(October 2006):575-579.

5. Halperin I, Ma B, Wolfson H, Nussinov R. Principles of docking: An overview of search algorithms and a guide to scoring functions. *Proteins*. 2002;47(4):409-43.

6. Taylor JS, Burnett RM. DARWIN: a program for docking flexible molecules. *Proteins*. 2000;41(2):173-91. 7. Luty BA, Wasserman ZR, Stouten PF, et al. A molecular mechanics/grid method for evaluation of ligand-receptor interactions. *Journal of Computational Chemistry*. 1995;16(4):454-464.

8. Connolly ML. Analytical molecular surface calculation. *Journal of Applied Crystallography*. 1983;16(5):548-558. 9. Ausiello G, Cesareni G, Helmer-Citterich M. ESCHER: a new docking procedure applied to the reconstruction of protein tertiary structure. *Proteins*. 1997;28(4):556-67.

10. Norel R, Lin SL, Wolfson HJ, Nussinov R. Shape complementarity at protein-protein interfaces. *Biopolymers*. 1994;34(7):933-40. 11. Connolly ML. Solvent-accessible surfaces of proteins and nucleic acids. *Science (New York, N.Y.)*. 1983;221(4612):709-13. Available at: http://www.ncbi.nlm.nih.gov/pubmed/6879170.

12. Schafferhans A, Klebe G. Docking ligands onto binding site representations derived from proteins built by homology modelling. *Journal of molecular biology*. 2001;307(1):407-27.

13. Tsai CJ, Ma B, Sham YY, Kumar S, Nussinov R. Structured disorder and conformational selection. *Proteins*. 2001;44(4):418-27.

14. Todd MJ, Semo N, Freire E. The structural stability of the HIV-1 protease. *Journal of molecular biology*. 1998;283(2):475-88.

15. Kuntz ID, Blaney JM, Oatley SJ, Langridge R, Ferrin TE. A geometric approach to macromolecule-ligand interactions. *Journal of molecular biology*. 1982;161(2):269-88.

16. Katchalski-Katzir E, Shariv I, Eisenstein M, et al. Molecular surface recognition: determination of geometric fit between proteins and their ligands by correlation techniques. *Proceedings of the National Academy of Sciences of the United States of America*. 1992;89(6):2195-9.

17. Palma PN, Krippahl L, Wampler JE, Moura JJ. BiGGER: a new (soft) docking algorithm for predicting protein interactions. *Proteins*. 2000;39(4):372-84.

18. Vakser IA, Matar OG, Lam CF. A systematic study of low-resolution recognition in protein--protein complexes. *Proceedings of the National Academy of Sciences of the United States of America*. 1999;96(15):8477-82.

19. Li Z, Scheraga Ha. Monte Carlo-minimization approach to the multiple-minima problem in protein folding. *Proceedings of the National Academy of Sciences of the United States of America*. 1987;84(19):6611-5.

20. Abagyan R, Totrov M, Kuznetsov D. ICM: A new method for protein modeling and design: Applications to docking and structure prediction from the distorted native conformation. *Journal of Computational Chemistry*. 1994;15(5):488-506.

21. Sandak B, Nussinov R, Wolfson HJ. An automated computer vision and roboticsbased technique for 3-D flexible biomolecular docking and matching. *Bioinformatics*. 1995;11(1):87-99.

22. Hayward S, Kitao A, Berendsen HJ. Model-free methods of analyzing domain motions in proteins from simulation: a comparison of normal mode analysis and molecular dynamics simulation of lysozyme. *Proteins*. 1997;27(3):425-37.

23. Norel R, Petrey D, Wolfson HJ, Nussinov R. Examination of shape complementarity in docking of unbound proteins. *Proteins*. 1999;36(3):307-17.

24. Kufareva I, Budagyan L, Raush E, et al. PIER: Protein Interface Recognition for Structural Proteomics. *Bioinformatics*. 2007;417(December 2005):400-417.

25. Gregoret LM, Sauer RT. Additivity of mutant effects assessed by binomial mutagenesis. *Proceedings of the National Academy of Sciences of the United States of America*. 1993;90(9):4246-50.

26. Bairoch A, Apweiler R, Wu CH, et al. The Universal Protein Resource (UniProt). *Nucleic acids research*. 2005;33(Database issue):D154-9.

27. Chenna R. Multiple sequence alignment with the Clustal series of programs. *Nucleic Acids Research*. 2003;31(13):3497-3500.

28. Larkin MA, Blackshields G, Brown NP, et al. Clustal W and Clustal X version 2.0. *Bioinformatics (Oxford, England)*. 2007;23(21):2947-8.

29. Thompson JD, Higgins DG, Gibson TJ. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic acids research*. 1994;22(22):4673-80.

30. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. *Journal of molecular biology*. 1990;215(3):403-10.

31. McGinnis S, Madden TL. BLAST: at the core of a powerful and diverse set of sequence analysis tools. *Nucleic acids research*. 2004;32(Web Server issue):W20-5.

32. Vej GW. Bioinformatics Explained. 2007.

33. BLAST - Download. Available at: http://blast.ncbi.nlm.nih.gov/Blast.cgi?CMD=Web&PAGE_TYPE=BlastDocs&DOC_TYPE=Download34. PSI-BLAST Tutorial. Available at: http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/psi1.html.

35. Bhagwat M, Aravind L. PSI-BLAST tutorial. *Methods in molecular biology (Clifton, N.J.)*. 2007;395:177-86.

36. Chang J, Chapman B, Friedberg I, et al. *Biopython Tutorial and Cookbook*.; 2009.

37. Cock PJ, Antao T, Chang JT, et al. Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics (Oxford, England)*. 2009;25(11):1422-3.

38. Yip KY, Patel P, Kim PM, et al. An integrated system for studying residue coevolution in proteins. *Bioinformatics (Oxford, England)*. 2008;24(2):290-2.

39. Halperin I, Wolfson H, Nussinov R. Correlated mutations: advances and limitations. A study on fusion proteins and on the Cohesin-Dockerin families. *Proteins*. 2006;63(4):832-45.

40. Gouveia-Oliveira R, Pedersen AG. Finding coevolving amino acid residues using row and column weighting of mutual information and multi-dimensional amino acid representation. *Algorithms for molecular biology : AMB*. 2007;2:12.

41. Dekker JP, Fodor A, Aldrich RW, Yellen G. A perturbation-based method for calculating explicit likelihood of evolutionary covariance in multiple sequence alignments. *Bioinformatics (Oxford, England)*. 2004;20(10):1565-72.

42. Mintseris J, Weng Z. Structure, function, and evolution of transient and obligate protein-protein interactions. *Proceedings of the National Academy of Sciences of the United States of America*. 2005;102(31):10930-5.

43. Bourne Y, Watson MH, Hickey MJ, et al. Crystal structure and mutational analysis of the human CDK2 kinase complex with cell cycle-regulatory protein CksHs1. *Cell*. 1996;84(6):863-74.

44. Mossessova E, Lima CD. Ulp1-SUMO crystal structure and genetic analysis reveal conserved interactions and a regulatory element essential for cell growth in yeast. *Molecular cell*. 2000;5(5):865-76.

45. Nassar N, Hoffman GR, Manor D, Clardy JC, Cerione RA. Structures of Cdc42 bound to the active and catalytically compromised forms of Cdc42GAP. *Nature structural biology*. 1998;5(12):1047-52.

46. Zesławski W, Beisel HG, Kamionka M, et al. The interaction of insulin-like growth factor-I with the N-terminal domain of IGFBP-5. *The EMBO journal*. 2001;20(14):3638-44.

47. Vigers GP, Anderson LJ, Caffes P, Brandhuber BJ. Crystal structure of the type-I interleukin-1 receptor complexed with interleukin-1beta. *Nature*. 1997;386(6621):190-4.

48. Huizinga EG, Tsuji S, Romijn RA, et al. Structures of glycoprotein Ibalpha and its complex with von Willebrand factor A1 domain. *Science (New York, N.Y.)*. 2002;297(5584):1176-9.

49. Gorina S, Pavletich NP. Structure of the p53 tumor suppressor bound to the ankyrin and SH3 domains of 53BP2. *Science (New York, N.Y.)*. 1996;274(5289):1001-5.

50. Ostermeier C, Brunger AT. Structural basis of Rab effector specificity: crystal structure of the small G protein Rab3A complexed with the effector domain of rabphilin-3A. *Cell*. 1999;96(3):363-74.

51. Apache Ant. Available at: http://ant.apache.org/.

52. DS Visualizer. Available at: http://accelrys.com/products/discovery-studio/visualization-download.php.