UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS

Disclosing the aggregation mechanism of β2-microglobulin in amyloid disease

" Documento Definitivo"

Doutoramento em Biologia

Especialidade de Biologia de Sistemas

Rui João de Sousa Loureiro

Tese orientada por:

Patrícia Ferreira Neves Faísca

Eugene Shakhnovich

Documento especialmente elaborado para a obtenção do grau de doutor

2019

UNIVERSIDADE DE LISBOA

FACULDADE DE CIÊNCIAS

Disclosing the aggregation mechanism of β2-microglobulin in amyloid disease

Doutoramento em Biologia

Especialidade de Biologia de Sistemas

Rui João de Sousa Loureiro

Tese orientada por:

Patrícia Ferreira Neves Faísca

Eugene Shakhnovich

Júri:

Presidente:

- Doutor Rui Manuel dos Santos Malhó, Professor Catedrático e Presidente do Departamento de Biologia Vegetal da Faculdade de Ciências da Universidade de Lisboa

Vogais:

- Doutor Rui Davide Martins Travasso, Professor Auxiliar
  Faculdade de Ciências e Tecnologia da Universidade de Coimbra;
- Doutor Alexandre Lopes de Magalhães, Professor Auxiliar
  Faculdade de Ciências da Universidade do Porto;
- Doutora Patrícia Ferreira Neves Faísca, Professora Auxiliar
  Faculdade de Ciências da Universidade de Lisboa (orientadora);
- Doutor Miguel Ângelo dos Santos Machuqueiro, Professor Auxiliar Convidado
  Faculdade de Ciências da Universidade de Lisboa

Documento especialmente elaborado para a obtenção do grau de doutor

Apoio financeiro da FCT através da bolsa *SFRH/BD/108259/2015* atribuída no âmbito do programa doutoral BioSys e do projeto PTDC/FIS-OUT/28210/2017

2019

Acknowledgments

To my supervisor, Professor Patrícia Faísca, who was always available to give me support, guidance and constructive critics and suggestions, that were very important during all the four years of this PhD project but that were particularly important in the beginning of the project as I had then a weak background in Physics and Computer Science.

To my co-supervisor, Professor Eugene Shakhnovich, for the available way he hosted me at his group and for the helpful discussions and suggestions he gave me during my stay there.

To the present and past members of my research group who shared with me knowledge, ideas and suggestions on my work during this period: Sílvia Estácio, Jules Morand, João Especial, Professor Ana Nunes.

To Miguel Machuqueiro and Diogo Vila-Viçosa, who gave a very important contribution to the methodological approach of this project and who were always available to give useful suggestions on the methodological aspects of the project and to participate on the discussion of the results. They, along with the other members of their groups, were especially supportive in the last year of my Project when I applied some of the methodologies usually carried out in their groups to my Project.

To Professor Margarida Amaral, director of the BioSys PhD program to whom my project was assigned and director of BioISI, for the support she gave to the BioSys PhD students during these years.

To my family and friends, for all the support and motivation they gave me during these years.

To all the BioSys and BioISI and Boston/Harvard colleagues for their support and friendship.

Abstract

Protein β2-microglobulin (β2-m) is the causative agent of dialysis-related amyloidosis (DRA), a prevalent pathology affecting individuals undergoing long-term hemodialysis. The goal of this PhD project is to explore the early stage of the aggregation mechanism of β2-m with molecular simulations, using two model systems: the ΔN6 variant, a cleaved form lacking the six N-terminal residues, which is a major component of *ex vivo* amyloid plaques from DRA patients, and the single point D76N mutant, recently identified as the cause of an hereditary systemic amyloidosis affecting visceral organs. Methodologically, the main goal of this research project is the development of a Monte Carlo Ensemble Docking method with a cost function that considers shape, hydrophobic and electrostatic complementarity, the major drivers of protein-protein association.

The D76N mutant populates two folding intermediates called I_1 and I_2, which display an unstructured C-terminus and two unstructured termini, respectively. The ΔN6 variant populates one folding intermediate, with an unstructured N-terminus.

Protein-protein docking simulations predict an essential role for the termini and for the DE-loop (both variants), EF-loop (D76N mutant) and BC-loop (ΔN6 variant) in the dimerization mechanism of β2-m. The terminal regions are more relevant under acidic conditions while the BC-, DE- and EF-loops gain importance at physiological pH.

Our results recapitulate experimental evidence according to which Phe30 and His31 (BC-loop), Arg45 (CD-loop), and Trp60 and Phe62 (DE-loop) are dimerization hotspots (i.e. residues triggering dimerization). Additionally, we predicted the involvement of new residues such as Tyr10 (A-strand), Lys75 (EF-loop), and Trp95 and Arg97 (C-terminus), thus providing new testable predictions to guide the research on β2-m amyloidogenesis.

Finally we predicted that β2-m tetramerization is mainly driven by the self-association of dimers via the N- and C-terminal regions and the DE-loop, and identify Arg3 (N-terminus), Tyr10, Arg45, Phe56 (D-strand), Trp60 and Arg97 as essential residues in the process.

Keywords: protein folding; intermediate states; protein aggregation; amyloidosis; protein-protein docking

Resumo

A proteína β2-microglobulina (β2-m) é o agente etiológico da amiloidose relacionada com a diálise, uma patologia humana prevalente que afeta >90% dos indivíduos com insuficiência renal crónica em tratamento de hemodiálise de longa duração. O objetivo deste projeto de doutoramento é explorar com detalhe microscópico a fase inicial do mecanismo de agregação da proteína β2-m com simulações moleculares. O objetivo deste projeto doutoral compreende (1) a caracterização estrutural de intermediários de enrolamento propícios a agregar e a (2) caracterização estrutural dos oligómeros iniciais (dímeros e tetrâmeros) que se formam ao longo da via de agregação usando docking proteína-proteína. Os métodos de docking proteína-proteína são métodos que prevêm a estrutura tridimensional de um complexo proteína-proteína a partir das coordenadas dos seus componentes monoméricos. De um ponto de vista metodológico, o principal objetivo do presente projeto de investigação é o desenvolvimento de uma nova função custo para o método "Monte Carlo Ensemble Docking" desenvolvido no nosso grupo. Esta função custo considera a complementaridade da forma, hidrofóbica (i.e. mimitizando as interações entre grupos hidrofóbicos que representam o efeito hidrofóbico) e electrostática (incluindo as pontes de hidrogénio intermoleculares), que são os principais determinantes da associação proteína-proteína. Isto representa uma evolução significativa em relação à versão original do algoritmo, baseada exclusivamente na complementaridade da forma.

Infelizmente, a forma wt não agrega *de novo* em condições fisiológicas *in vitro*, e ao longo dos anos têm sido explorados modelos naturais ou artificiais para obter pistas sobre o mecanismo fibrilogénico da espécie parental. Neste trabalho, usámos dois sistemas modelo: a variante ΔN6, uma forma clivada sem os seis resíduos N-terminais, que é um componente maior das placas amilóides *ex vivo* dos pacientes, e a mutação pontual D76N, recentemente identificada numa família francesa como o agente etiológico de uma amiloidose hereditária sistémica afetando os órgãos viscerais. Foram apresentadas duas hipóteses para o mecanismo de agregação da β2-m na amiloidose relacionada com a diálise baseadas na variante ΔN6: (1) um mecanismo "prionlike" em que a variante ΔN6 é capaz de desencadear a conversão da forma wt numa conformação amiloidogénica de uma forma análoga às proteínas priónicas; (2) e uma hipótese baseada na existência de um intermediário de enrolamento com potencial para agregar que apresenta uma região N-terminal desestruturada e deslocada do "core" da proteína, particularmente ao pH ligeiramente acídico 6.2 presente no fluido sinovial dos pacientes, iniciando assim a cascata de agregação. No entanto, a importância biológica desta

variante não está ainda estabelecida, dado que esta não é detetada no sangue dos pacientes, o que pode limitar o seu uso como modelo da agregação da β2-m. Em contraste, o reconhecimento da mutante D76N como causa de uma amiloidose hereditária sistémica a par do seu potencial amiloidogénico *in vitro* em condições fisiológicas "unseeded" tornam esta variante num modelo biológica e clinicamente interessante da agregação da β2-m, que tem vindo a ser extensivamente usado nos últimos anos. Os resultados apresentados aqui ajudam a obter pistas acerca do mecanismo de fibrilogénese da espécie parental, mas eles não reservam um papel exclusivo da espécie truncada no mecanismo fibrilogénico da proteína wt "full-length", nem reduzem o mesmo à via de agregação da mutante D76N. De facto, é provável que a agregação da proteína wt "full-length" seja estritamente dependente em condições ambientais únicas presentes no sistema osteoarticular dos pacientes em diálise, e, assim, essas devem ser identificadas e mimetizadas quer *in vitro* quer em simulações de modo a obter um retrato mais preciso da agregação da wt β2-m na DRA.

Nós focámos a nossa análise no processo de associação de estados intermediários de enrolamento com potencial para agregação que foram identificados em estudos de simulação (57, 116) baseados em modelos "structure-based" para enrolamento de proteínas, i.e. que realçam as características topológicas do processo de associação. A mutante D76N apresenta dois intermediários de enrolamento designados I_1 e I_2, que apresentam um C-terminal desestruturado e dois términos desestruturados, respetivamente. A variante ΔN6 apresenta um intermediário de enrolamento, com um N-terminal desestruturado. A importância de regiões terminais desestruturadas no mecanismo de agregação da β2-m e na agregação de outros sistemas modelo tem sido demonstrada em diversos estudos.

Enquanto que a versão original do algoritmo Monte Carlo ensemble docking previa um papel direto das regiões terminais desestruturadas de todos os estados intermediários no desencadear da agregação, a nova versão do algoritmo indica que o desenrolamento e o distanciamento das regiões terminais do "core" da proteína pode aumentar a mobilidade e a exposição ao solvente de outros elementos estruturais que agora aparecem como regiões "sticky", nomeadamente o DE-loop (na dimerização das duas variantes da β2-m), o EF-loop (na dimerização da mutante D76N) e o BC-loop (na dimerização da variante ΔN6). Em particular, a nova função custo realça um papel claramente mais importante para o DE-loop e o EF-loop na dimerização do intermediário I_2 (D76N) a pH 5.2. Em geral, o DE-loop, o EF-loop e o BC-loop dominam a pH fisiológico e as regiões terminais a pH acídico. Interessantemente, o papel surpreendentemente dominante do C-terminal e da G-strand

desestruturada adjacente na dimerização do I_2 a pH acídico previamente identificado é substancialmente suprimido quando as interações electrostáticas (ex: aquelas envolvendo resíduos polares e carregados do N-terminal e da A-strand) são também incluídas na função custo. A nova função custo indica um papel relevante para a A-strand na dimerização do ΔN6 ao pH ligeiramente acídico 6.2 e uma marca distinta do BC-loop, que não era tão visível com a versão original do método, muito provavelmente devido às interações electrostáticas estabelecidas pela His31.

Também analisámos pela primeira vez as interfaces resultantes das interações intermoleculares entre o estado nativo da ΔN6 e o estado nativo da wt β2-m, que estão na base do mecanismo "prion-like" para a amiloidogénese da β2-m. Os dímeros que obtivemos são os mais instáveis de todos os dímeros estudados neste estudo, apresentando energias de ligação relativamente elevadas de acordo com os dados experimentais reportados por Radford e seus colaboradores (239). Os nosssos resultados suportam o envolvimento do DE-loop, do BC-loop e do FG-loop nas interfaces dos heterodímeros da ΔN6 e da wt β2-m, também em linha com dados experimentais.

Os resultados de simulações extensivas realizadas no presente estudo estão em linha com os resultados experimentais suportando um papel essencial para a Phe30 e His31 (BC-loop), Arg45 (CD-loop), e Trp60 e Phe62 (DE-loop) na dimerização da β2-m. Adicionalmente, eles prevêm novos hotspots como a Tyr10 (A-strand), a Lys75 (EF-loop), e o Trp95 e a Arg97 (C-terminal).

Finalmente, através do estudo da dimerização dos dímeros do intermediário I_2 populado pela mutante D76N e do intermediário I do ΔN6, obtivemos primeiras pistas acerca da interface de tetramerização. Verificámos que a tetramerização da ΔN6 é menos favorável do que a da D76N tal como observado nas curvas de densidade das energias de ligação dos tetrâmeros, o que contribui para explicar a amiloidogénese *in vitro* mais baixa da ΔN6 em relação à D76N. Nós previmos também que as regiões N- e C-terminais e o DE-loop têm um papel importante na estrutura da interface do tetrâmero, e propomos que a formação dos tetrâmeros pode ser mediada por interacções envolvendo o Trp60 (DE-loop), a Arg3 (N-terminus), a Phe56 (D-strand), a Tyr 10 (A-strand), a Arg97 (C-terminus), a Arg45 (CD-loop) e, num grau menor, a Gln89 (FG-loop) e a Lys58 (DE-loop).

Palavras-chave: enrolamento de proteínas; estados intermediários; agregação de proteínas; amiloidoses; docking proteína-proteína

Table of Contents

Figures Index

Tables Index

List of abbreviations

AIR	Ambiguous Interaction Restraints
ASP	Atomic solvation parameters
β2-m	β2-microglobulin
CD	Circular dichroism
CpHMD	Constant pH Molecular Dynamics
DMD	Discrete Molecular Dynamics
DRA	Dialysis related amyloidosis
E	Energy
EM	Energy minimization
HADDOCK	High Ambiguity Driven Biomolecular Docking
HSQC	Heteronuclear single quantum coherence
IDP	Intrinsically Disordered Protein
IPMs	Intermolecular Probability Maps
MC	Monte Carlo
MC-ED	Monte Carlo Ensemble Docking
MD	Molecular Dynamics
MHC-1	Class I major histocompatibility complex
MM/PBSA	Molecular Mechanics/Poisson-Boltzmann Surface Area
MMTSB	Molecular Modelling Tools for Structural Biology
NMR	Nuclear Magnetic Resonance
PD	Parkinson's disease
PDB	Protein Data Bank
PDF	Probability Density Function
PED	Protein Ensemble Database
R_g	Gyration Radius
RMSD	Root-Mean-Square-Deviation
SASA	Solvent accessible surface area
TAD-SA	Simulated annealing in torsion angle space
TFE	Trifluoroethanol
WT	wild-type

Chapter 1. Introduction

1.1. Protein folding, aggregation and amyloid disease

1.1.1 Protein folding and aggregation

Protein folding is the self-assembly process according to which a linear polypeptide chain acquires a specific three-dimensional, biologically functional native structure. A major driving force for folding is the so-called hydrophobic effect. The latter forces the hydrophobic amino acids to get buried inside the protein's core, while the hydrophilic ones expose themselves to the solvent being located on the protein's surface. All globular proteins fold spontaneously into their respective compact native structures, but the so-called intrinsically disordered proteins only acquire the native structure upon substrate binding. A paradigmatic example of an intrinsically disordered protein is alpha-synuclein, the causing agent of Parkinson's disease.

While most small (~100 amino acids), single-domain proteins fold via a two-state process where the folding process is dominated by the native and unfolded states (1), large proteins typically populate intermediate states along their folding pathway. The fact that a process is thermodynamically (and kinetically) two-state does not imply the absence of intermediate states. However, such intermediate conformations can interconvert so rapidly that they are not experimentally detectable, unless high-resolution methods are deployed (2). Furthermore, the formation of partially folded intermediates can be triggered by environmental factors (e.g. temperature and pH), genetic mutations that thermodynamically destabilize the native state, or chemical modifications such as acetylation, methylation or glycosylation. Sometimes the partially folded intermediates expose hydrophobic patches, which render them prone to self-associate to reduce solvent exposure. Additionally, intrinsically disordered proteins and peptides, as well as unfolded protein fragments produced by proteolysis can also self-associate under appropriate conditions (e.g. concentration increase) (3).

Protein aggregation is the process by which monomeric proteins self-associate to form higher-order oligomers (e.g. dimers, trimers, tetramers) (4). Most often, the process of protein aggregation gives rise to amorphous (i.e. disordered) aggregates with a granular appearance (4). Sometimes, however, the final outcome of protein aggregation are highly ordered aggregates known as amyloid fibrils, and the process leading to them is called amyloidogenesis (4). The amyloid fibrils (or fibres) exhibit a common cross β-structure,

characterized by the formation of β-sheets perpendicular to the axis of the fibril growth (Figure 1). They are also characterized by binding the dyes thioflavin T and Congo red and by displaying red-green birefringence when observed under polarized light (3-7). Notwithstanding, the mature amyloid fibrils display a high morphological, structural and thermodynamic heterogeneity, which reflects the structural heterogeneity of protein conformations that can trigger the amyloid cascade as well as the different environmental conditions under which protein aggregation may occur (e.g. temperature, pH) (3, 4).

Figure 1. Representation of the structure of the amyloid fibrils of the 42-residue human amyloid β (Aβ(1-42)) peptide with the characteristic cross β-sheet motif of amyloid fibrils (adapted from Ref. (4)).

Establishing the amyloidogenesis mechanism requires the identification of all microscopic steps leading to mature fibrils. This comprises the determination of the size distribution and structures of the oligomeric assemblies, filaments, fibrils and protofibrils that form along the amyloid cascade (Figure 2) (4). Furthermore, important insights may be gained by determining the kinetics of protein aggregation (i.e. by evaluating the rate constants governing each microscopic step), and how the latter depends on protein sequence and environmental conditions (8). The oligomers formed in the initial steps of aggregation can either dissociate back to monomers or may undergo further growth to form a critical nucleus. The critical nucleus is the smallest oligomer that is stable enough such that further growth by monomer addition is faster than its dissociation into monomers (4). The critical nucleus is

formed during the lag phase of protein aggregation, which is the initial, rate-determining phase, in which the monomers and small soluble oligomers are still the main species (4). Subsequently, in the so-called growth or elongation phase, the oligomers in this critical nucleus may undergo a structural rearrangement to form filaments and amyloid-like protofibrils. When these protofibrils are present in a sufficient number, the energetically favourable enthalpic contribution of their regular stacking overcomes the unfavourable loss in configurational entropy, and they can grow into mature fibrils by further self-association or by the subsequent addition of monomers. These events take place mainly in the last (plateau) phase of protein aggregation (3, 4).

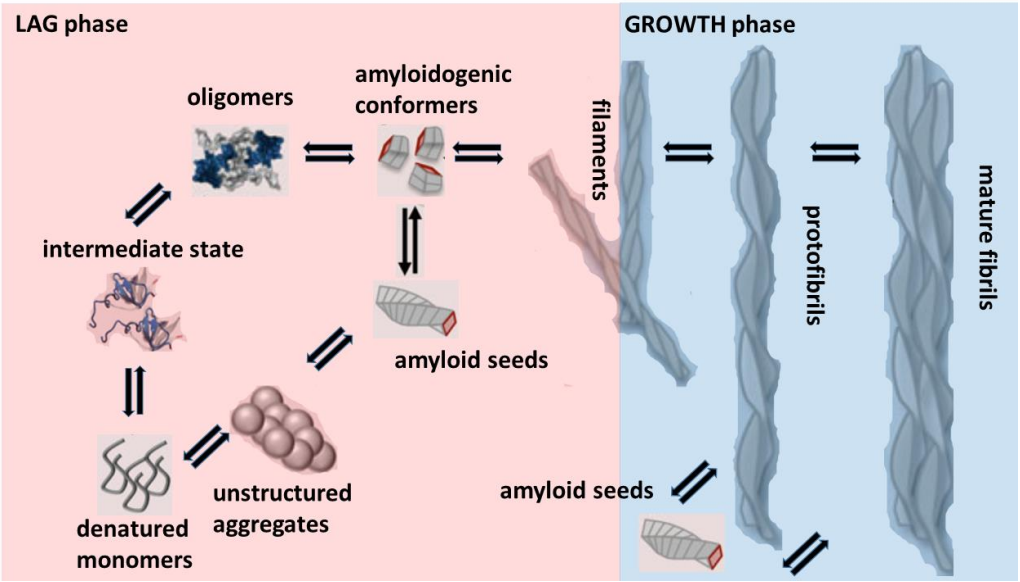


Figure 2. Different phases of the protein aggregation mechanism and molecular species involved (Adapted from Ref. (4)).

The transient and heterogeneous nature of the conformational states populated along the aggregation pathway makes their structural characterization by standard biophysical methods extremely difficult. Additionally, the differences in the size and timescale of formation of these conformational states require the use of a multitude of different techniques to cover the different sizes and timescales involved (4). Furthermore, the formation of the different types of aggregates is critically dependent on environmental conditions such as the pH, temperature and salt concentration, which further hinder their experimental characterization. As such, molecular simulations have been emerging as an important tool in the study of protein aggregation (9).

1.1.2 Amyloid disease

The so-called protein folding diseases (or conformational disorders) refer to a vast group of pathologies that are related to faulty protein folding or to misfolding and aggregation (3, 4). The prevalence of conformational disorders is dramatically increasing worldwide (10, 11). Currently, there are more than 40 diseases associated with the formation and deposition of amyloids (3). The so-called amyloid diseases (or amyloidosis) can be classified into neurodegenerative diseases like Alzheimer's and Parkinson's diseases (aggregation occurs in the nervous system), non-neuropathic localized amyloidosis (aggregation occurs in a single type of non-neuronal tissue), and non-neuropathic systemic amyloidosis. An example of the latter is dialysis related amyloidosis (DRA), a condition affecting patients undergoing long-term hemodialysis, which results from the deposition of amyloid plaques of protein β2-microglobulin (β2-m) in the osteoarticular system eventually leading to bone destruction and neuropathic symptoms (12, 13).

A fundamental question in the field of amyloid diseases concerns the origin of cytotoxicity. Indeed, while the classical view of amyloidosis states that amyloid fibrils are the toxic species (14), recent evidence suggests that the oligomers formed along the amyloid cascade have serious cytotoxic effects (e.g. the disruption of the permeability of cellular membranes through the formation of pores (15) and the formation of reactive oxygen species (ROS) (16)) being the primary toxic species, with the amyloid fibrils being innocuous or even protective (17).

The cellular housekeeping and repair machinery, particularly the protein quality control systems, play a fundamental role in preventing protein aggregation. The protein quality control systems comprise folding catalysts like enzymes and molecular chaperones and the degradation system (4). Indeed, specific enzymes such as prolyl isomerases and disulphide isomerases are essential in the proper folding of some proteins through the catalysis of essential rate-limiting steps of their folding (4). Molecular chaperones are proteins that assist other proteins in the acquisition of their functional native states (18). Furthermore, they also rescue the proper fold of misfolded proteins, while the cell's degradation machinery degrades misfolded proteins or the oligomeric aggregates they create, thus inhibiting the formation of protein aggregates (3, 4). These systems also eliminate most of the normal proteins as part of the normal recycling of the cell components. Notwithstanding, sometimes the aggregation processes can circumvent these defense mechanisms, which occurs

particularly due to accumulative defects in the cell's quality control and repair machinery, explaining the higher prevalence of conformational disorders in elderly people.

It is therefore of paramount importance a good understanding of protein folding and aggregation for the rational design and development of effective therapies for conformational disorders, particularly for amyloid diseases.

1.2. β2-m and amyloid disease

1.2.1 The protein β2-m

This thesis is focused on protein aggregation and, in particular, on exploring the early phase of the aggregation mechanism of protein β2-m with molecular simulations. β2-m is a 99 long residue protein of the immunoglobulin superfamily that constitutes the non-covalently bound light chain of class I major histocompatibility complex (MHC-1), assisting the efficient transport of nascent MHC-I chains to the surface of all nucleated cells (19-21). Its structure comprises a classical β-sandwich fold with seven antiparallel β-strands (A through G) organized in two sheets of antiparallel β-strands, one comprising the strands A-B-E-D and the other comprising the strands C-F-G. The native structure is stabilized by a disulfide bridge between the Cysteine residues at positions 25 (at B strand) and 80 (at F strand) (19, 20, 22) (Figure 3), which has been regarded as fundamental in β2-m fibrillogenesis at neutral pH (23, 24). Another fundamental structural feature of β2-m is the His31-Pro32 peptide bond in the BC loop, which adopts the thermodynamically unfavourable *cis*-isomer in the native state and the *trans*-isomer when the protein partially or totally unfolds (19, 25). The docking of β2-m onto the β_3 domain of MHC-1 heavy chain involves the four-stranded beta-sheet A-B-E-D, giving rise to a 595 Å interface (26). Interestingly, four aromatic residues (Phe56, Trp60, Phe62 and Tyr63) and one aliphatic residue (Leu65) that are not accessible to the solvent in the quaternary structure of the MHC-1 become highly solvent-exposed upon the dissociation of the β2-m from the MHC-1 heavy chain.

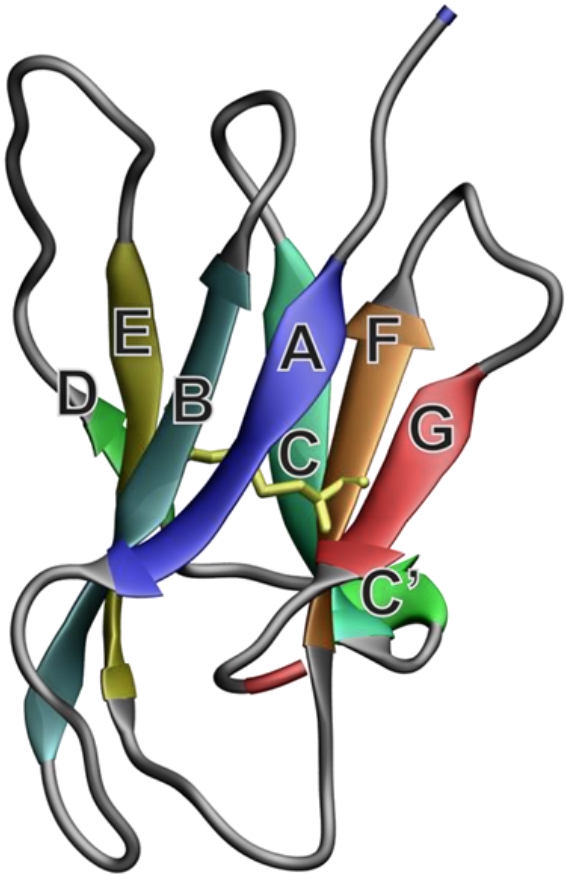


Figure 3. Cartoon representation of the structure of β2-m showing the seven β-strands and the disulfide bond (PDB ID: 2XKS).

As mentioned before, β2-m is the causative agent of a very prevalent human pathology known as DRA (27, 28). This disease is characterized by the deposition of β2-m amyloid fibrils in the osteoarticular tissues of individuals with chronic renal failure undergoing long-term hemodialysis (usually more than 90% of the individuals undergoing hemodialysis for > 10 years develop the condition) (19, 29). The generated amyloid deposits are responsible for a destructive arthropathy, cystic bone lesions, and carpal tunnel syndrome and other neuropathies, ultimately leading to joint pain, impaired function and bone fractures (12, 13).

The pathological process of DRA is partially driven by a dramatic 60-fold increase in the plasmatic concentration of β2-m (20, 29), which results from the incapacity of the kidney to catabolize the protein and the inability of the dialysis apparatus to filter the protein. The protein's high affinity for collagen leads to its deposition in the bones and cartilages (19, 30).

However, β2-m serum levels do not correlate with the fibril load in osteoarticular tissue (20, 28), and experiments *in vitro* show that a very high concentration (100-200 μM) of β2-m is not enough to trigger the amyloid cascade in physiological conditions (19). This observation suggests that other factors such as the patient's age (31), duration of renal failure (29), the dialysis procedure (32), rare nucleation events of conformationally destabilized monomers (33), and the interaction with local factors present in the osteoarticular tissues are necessary conditions for β2-m amyloid formation. Among the most influential local factors influencing β2-m amyloidogenesis stand out copper (Cu^{2+}) ions (34), local inflammation and subsequent pH lowering (35), and the presence of molecules like glycosaminoglycans (36), lysophosphatidic acid (37), non-esterified fatty acids (38) and collagen (39) (Figure 4).

Figure 4. Tissue specificity of the amyloid deposition in DRA and factors influencing DRA pathogenesis.

In 2012, the D76N point mutant, which features an asparagine instead of an aspartate (Figure 5), was identified in a French family as the etiological agent of a hereditary systemic amyloidosis affecting visceral organs (40). Indeed, all the heterozygous carriers of the

mutation presented a rare form of systemic amyloidosis (autosomal dominant inheritance) characterized by the deposition of amyloid fibrils in several visceral organs (liver, kidney, spleen and the heart) without the existence of any amyloid deposits in bones and ligaments (40). This localization of the amyloid deposits is quite unexpected considering the known tropism of wild-type (wt) β2-m for the musculoskeletal system (41). Other surprising findings are the fact that the wt β2-m is not present in the deposits (40) in spite of its intrinsic propensity to aggregate.

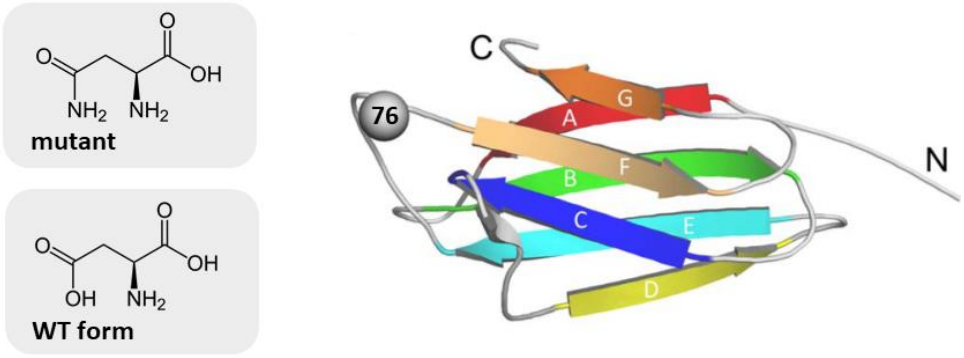


Figure 5. Location of the D76N mutation in the β2m structure and representation of the structures of the involved residues.

1.2.2 β2-m as a model system to study protein folding and aggregation

Classically, protein aggregation is viewed as the result of a strong destabilization of the native state such as that induced by low pH, high temperature, high ionic strength, or by a single-point mutation leading to a misfolded species by partial unfolding (42). β2-m was the first protein that challenged the classical view of protein aggregation. In particular, studies on β2-m showed for the first time that the processes of protein folding and aggregation may directly compete with each other. Indeed, Chiti and co-workers identified a partially folded intermediate in the folding landscape of β2-m that is also capable of elongating preformed amyloid fibrils (43, 44). This intermediate, originally termed I_2 and later renamed as I_T (due to its non-native *trans*-isomerization of the His31-Pro32 peptide bond) (25), accumulates in the slower phase of folding of β2-m, achieving a significant equilibrium concentration of ~14 ± 8% at conditions close to physiological (pH 7.4, 30 ºC) (44). This species has a lower level of β-sheet structure and a more exposed and unstructured hydrophobic core, which renders it more prone to aggregate (44). Additionally, it presents a 5-fold increase in its propensity to aggregate in the presence of preformed amyloid fibrils when compared with the native state (44). Therefore, the identification of the I_T intermediate was a landmark in protein science as this intermediate is a species that can either follow the folding pathway to generate the native

state, or trigger the aggregation cascade by interacting with other monomers (45, 46), thus connecting the folding and aggregation pathways (45, 47) (Figure 6).

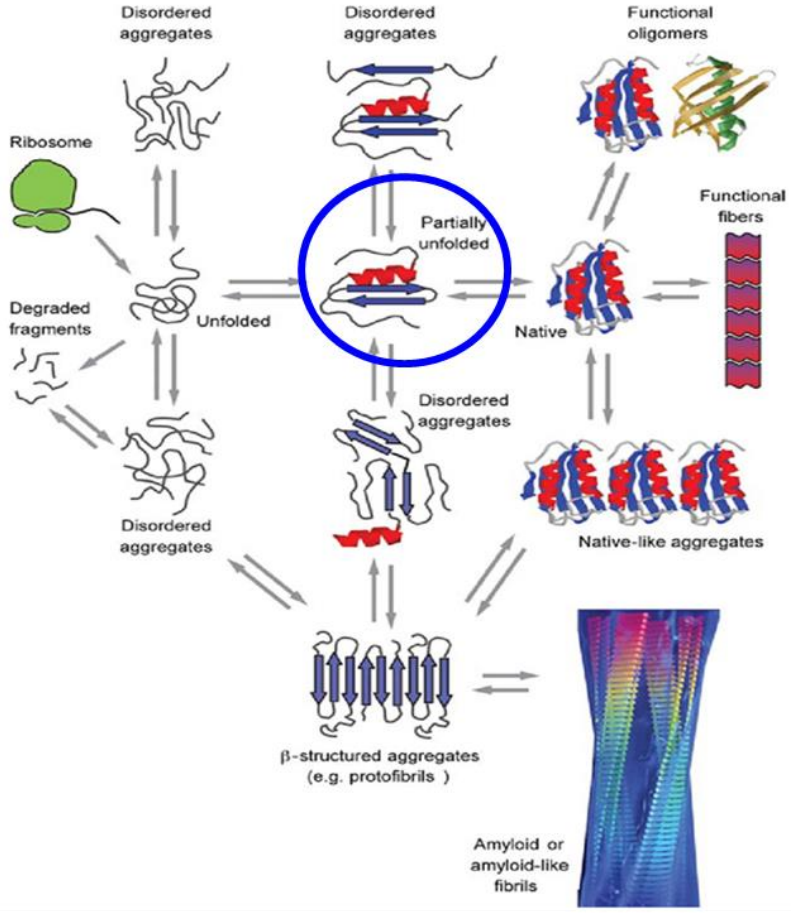


Figure 6. Folding and aggregation are competing pathways contrarily to the classical view of these processes ((Adapted from Ref. (3)).

1.3 An overview of the aggregation mechanism of β2-m in dialysis-related amyloidosis

1.3.1 The I_T intermediate and other intermediate states of β2-m

It has been proposed that the formation of one or more folding intermediate states, resulting from structural fluctuations of the native state, is necessary to initiate the β2-m aggregation process (20, 48, 49). These intermediate states may expose aggregation-prone sequences that are normally buried in the native structure, as a result of local and/or global unfolding events, a common feature of the aggregation mechanism of several globular proteins (20, 46, 50). In the case of β2-m, the I_T intermediate is a paradigmatic example of a partially folded species capable of initiating the aggregation cascade.

The idea that folding and aggregation are not independent of each other but directly competing processes prompted a multitude of experimental and computational studies (25, 33, 43, 44, 46, 48, 49, 51-59) directed towards the identification and structural characterization of folding intermediates of β2-m with the potential to initiate the aggregation process. Some of these studies focused on the structural characterization of I_T (25, 33, 46, 48, 53-55, 58, 59), including the influence of Cu^{2+} ions in its structure (58, 59), and the early oligomers generated by I_T (59).

Although the presence of a non-native *trans*-isomerization of the His31-Pro32 peptide bond is I_T defining feature, experimental studies using nuclear magnetic resonance (NMR) by Kameda *et al.* and x-ray crystallography by Eakin *et al.* and Calabrese *et al.* (25, 58, 59) provided a more complex picture of the conformational transition giving rise to I_T. This conformational transition includes the repacking of the hydrophobic core, particularly of Phe30, Phe56, Trp60, Phe62, Tyr63, Tyr66, Phe70 and Trp95 (25, 33, 58, 59), as well as a conformational rearrangement involving the N-terminus, the BC and DE loops and their adjacent strand extremities as observed in NMR studies by Rennella *et al.* (48), and Corazza *et al.* (33, 60). Some studies proposed a role for Cu^{2+} ions in the process of conformational conversion leading to I_T, namely in the isomerization of the His31-Pro32 bond (58, 59), by binding to His31 imidazole ring, and in a conformational change of Phe30 side-chain (59). These conformational changes ultimately lead to a reorganization of the aromatic side-chains of the BC and DE loops, giving rise to an alternative well-defined hydrophobic core (58, 59).

Additionally, several computational studies based on molecular dynamics (MD) also contributed to the structural characterization of I_T (53-55). In one of these studies Esposito and co-workers reported that the non-native *trans* conformation of the His31-Pro32 peptide bond disrupts a network of hydrogen bonds involving the residues His31 and Pro32 and residues located in the N-terminus and in the FG-loop (54). Additionally, Torbeev *et al.* predicted that, despite maintaining a nativelike tertiary structure, the I_T intermediate has an increased conformational flexibility, particularly in the AB-, BC- and DE-loops (55). These results rationalize the experimental results of Kameda, Eakin, Calabrese, Rennella and Corazza mentioned above by stressing out the fact that the formation of I_T and its increased aggregation propensity result from several conformational alterations. Additionally, Chong *et al.* (53) used MD simulations and solvation thermodynamics analysis on the wt β2-m, D76N, D59P and W60C mutants to determine which are the main structural and thermodynamic traits of the I_T intermediate state that contribute to amyloidogenicity. They identified three

features of the I_T intermediate that are correlated with the aggregation propensities of the different mutants, namely 1) the disruption of the edge D-strand, 2) the increase in the solvent exposure of the hydrophobic core, and 3) the increase in the solvation free energy relative to the native state.

Several other works identified and structurally characterized potentially aggregation-prone folding intermediates. An example is the study by Daggett and colleagues (52), that used MD simulations to predict a folding intermediate of β2-m characterized by having a α-pleated sheet structure over the central β-strands B-F. Subsequently, this research group identified and structurally characterized two partially folded intermediates of β2-m (49): an early native-like unfolding intermediate they called I_1, and a late unfolding intermediate with a more disordered structure they called I_2. The I_1 intermediate is characterized by having a native-like tertiary structure in the β-strands B, C, E and F while the I_2 intermediate is characterized by having α-extended chain conformations in the β-strands B, E and F, as well as hydrophobic clustering of side chains between these regions which contributes to its stability. Interestingly, these regions map to amyloidogenic peptides, which led the authors to propose that the formation of folding intermediates with α-sheet secondary structure promotes self-assembly into prefibrillar amyloidogenic oligomers.

Faísca *et al.* also studied the folding transition of the D59P and W60C mutants and of the wt β2-m with discrete molecular dynamics (DMD) of a full atomistic Gō model and found a common intermediate state in the three variants (56). The identified intermediate has a well preserved core region (strands B-F) and two unstructured termini. It also presents an increase in the solvent accessible surface area (SASA) of up to 40 times relative to the native state in about 40% of the hydrophobic residues, suggesting that this species has aggregation potential. The free energy surfaces of the folding space of wt β2-m suggest that this intermediate forms fast from the denatured state, which indicates that it can represent the I_1 intermediate reported by Chiti *et al.* (43, 44), or some conformational excursion of I_1 on the way to the I_T intermediate. Interestingly, the structural characteristics of the identified intermediate resemble that of a molten globule state of β2-m identified at pH 4.0 (61, 62), whose structure consists also of a stable and compact core comprising strands B, C, D, E and F and intervening loops, and in highly unstructured termini. Additionally, PROPKA predictions of pKa suggest that the population of the intermediate state may be favored at pH 4.0 as the charge of the native protein becomes more positive at this pH, which indicates that the identified intermediate could be the molten globule state identified *in vitro*.

1.3.2 Insights into β2m aggregation from engineered mutants

One intriguing feature of wt β2-m is its inability to aggregate *in vitro* under physiological conditions in the absence of denaturants like trifluoroethanol (TFE), Cu^{2+} or pre-formed *ex vivo* amyloid fibrils (19). This limitation makes the study of the β2-m aggregation mechanism particularly challenging, which led researchers to look to either natural or engineered variants of β2-m that aggregate *in vitro* at physiological conditions as model systems of β2-m aggregation. Among the natural variants most frequently studied are structural variants such as the ΔN6, devoid of the six N-terminal residues (57, 63-68), and the cleaved variant ΔK58, devoid of the residue Lys58 (63, 69-72). There are also point mutants of β2-m frequently used as model systems of β2-m aggregation such as the D76N natural mutant (40, 53, 73-76), the DE-loop engineered mutants D59P, W60C, W60V and W60G (53, 56, 77-85), the AB-loop engineered mutant H13F (59, 86) and the BC-loop engineered mutants H31Y (87, 88) and P32A (58, 86).

In the last years researchers have been resorting to engineered point mutants of β2-m to overcome the challenges of studying β2-m aggregation. Some of these mutants increase the aggregation propensity while others decrease it. For instance, Santambrogio *et al.* (81) have shown that the mutant W60G presents a significant increase in conformational stability using Trp fluorescence and circular dichroism (CD), while the W60V has no significant differences in conformational stability in relation to the wt form and the D59P mutant has a decrease in the conformational stability. Similarly, while all these three mutants display a decreased propensity to form oligomers, this tendency is more pronounced for the W60G than for the W60V and D59P. These observations provide a rational for the reason why the W60G mutant has less propensity to form amyloid fibrils than the wt as shown in an earlier study of the same group (82) and conversely why D59P has more aggregation propensity, which they also have shown previously (83).

Subsequently, the study by Natalello *et al.* (80) have reinforced the idea that the different aggregation propensities of the DE-loop mutants are the result of the different stabilities of the mutants as the amyloid fibrils generated by the different DE-loop mutants have the same general morphology and fibrillary architecture, which suggests that the aggregation pathways are similar. Interestingly, however, is the fact that the W60V mutant, as well as the W60C mutant, have lower aggregation propensity in relation to the wt despite having similar conformational stabilities (83, 84), which can be explained by the recognized relevance of the aromatic residues of the DE-loop for β2-m aggregation (59, 89).

Accordingly, the W60F mutant, which has an aromatic residue in the place of tryptophan 60, has a similar aggregation behavior to the wt (85).

Faísca *et al.* also proposed that the D59P is considerably less thermally stable than the wt (56) using DMD of a full atomistic Gō model of the folding pathways of both W60C and D59P. Furthermore, it increases the population of a folding intermediate with a molten-globule like character and in which about 40% of its hydrophobic residues are significantly solvent exposed. Together, these two findings suggest that the D59P mutant is more aggregation-prone than the wt, which rationalizes the experimental evidence by Santambrogio and Ricagno presented above. In contrast, the W60C mutant has an intermediate thermal stability between the wt and the D59P mutant and has a lower population of the molten-globule intermediate than the D59P mutant, which agrees with the W60C mutant lower aggregation propensity. Similarly, Narang and co-workers (79) have conducted MD simulations of the D59P mutant that showed an increased conformational flexibility of this variant due to a reduction in the number of hydrogen bonds in the loop regions as well as the presence of thermodynamically unstable intermediate states. Accordingly, Ham and colleagues (53) reported that the I_T intermediate of the D59P mutant has an enhanced β-sheet forming propensity in its disordered D-strand, an increased SASA of its hydrophobic residues, and an increased solvation free energy comparatively with the wt I_T intermediate. In contrast, the W60C mutant has a reduced β-sheet forming propensity in its disordered D-strand, a reduced SASA of its hydrophobic residues and a reduced solvation free energy comparatively with the wt I_T intermediate, which corroborates the experimental evidence and rationalizes the crucial importance of hydrophobic residues like Trp60 in β2-m *in vitro* aggregation.

The AB-loop H13F mutant and the BC-loop H31Y and P32A mutants have also been used to get insights on the β2-m aggregation mechanism. Indeed, Miranker and co-workers (59) used the H13F point mutant to study the oligomerization mechanism of β2-m induced by Cu^{2+} as this variant keeps the folding stability and the affinity of the Cu^{2+} binding of the wt while presenting a higher stability of its initial oligomers. Indeed, they provided evidence that this mutant mainly exists in oligomeric forms, mostly hexamers, allowing the structural characterization of the hexamer and its constituent interfaces as well as of the Cu^{2+} mediated conformational alterations already described. Similarly, Eakin *et al.* (58) have used the P32A mutant to establish the isomerization of the His31-Pro32 peptide bond as a crucial event in the Cu^{2+}-dependent generation of amyloidogenic conformations as the mutation converts the *cis* isomer of this bond characteristic of the native state to the *trans* isomer. Additionally, Blaho

and co-workers (86) used double mutants of P32A and H13F and other histidine substitutions to determine the role of individual imidazole side chains in the affinity of Cu^{2+} binding, native state stability and oligomerization mechanism. Indeed, they observed that Cu^{2+} binding induces the formation of tetramers for the P32A variant in which His51 has an essential role in oligomerization and that the His31 is the primary binding residue for Cu^{2+} like in the wt protein (90, 91). Interestingly, the double mutant P32A/H13F did not induce significant changes in the Cu^{2+} binding affinity nor in the oligomerization propensity in relation to the P32A, which contrasts with the single mutant H13F in relation to the wt, suggesting different oligomerization mechanisms.

Esposito and co-workers structurally characterized the H31Y mutant of β2-m (87, 88), a variant that, although being structurally similar and more stable than the wt protein, allowed them to get insights into the β2-m aggregation mechanism. Indeed, this mutant has a remarkable conformational heterogeneity populating a minor conformational state characterized by the detachment of the N-terminal strand-A from its native position, which is a structural modification frequently regarded as an early event in the β2-m aggregation cascade (57, 73, 91).

1.3.3 The ΔN6 variant

The structural variant ΔN6, one of the most extensively studied natural variants of β2-m, results from the proteolytic cleavage of the N-terminal hexapeptide and is significantly present (approximately 26%) in the *ex vivo* amyloid fibrils from DRA patients (92). This variant has a high aggregation propensity *in vitro* at physiological conditions, forming readily amyloid fibrils at neutral pH, in contrast to the full-length protein (that is not able to fibrillate in the absence of external factors (Cu^{2+}, denaturants) or amyloid seeds), and it displays less structural and thermodynamic stabilities. Indeed, it displays increased flexibility, loss of structure in the β-strands A, C and part of strand-B and a dislocation of strand-D that may be able to induce the formation of intermolecular contacts (26, 66).

Other researchers used computational simulations to characterize ΔN6. For instance, in an early study employing different types of MD simulations (63), Ma and Nussinov structurally characterized the ΔN6 conformational changes during folding and concluded that this species is more unstable than the wt protein, and that it facilitates the β-strand to α-helix transition. Moreover, they found that the strands B and E have increased flexibility, leading to a larger separation between them and to a concomitant dislocation of strand-D, which may be

responsible for the increased aggregation propensity of this species with regard to the wt protein. Subsequently, Fang *et al.* conducted MD simulations of ΔN6 (64) and reported that the removal of the N-terminal hexapeptide and the loss of the salt-bridge interaction between residues R3 and D59 lead to decreased structural stability and to an increased exposure of the K3 peptide (Ser20 – Lys41) to the solvent. This leads to a greater mobility of the strands B and E with a consequent increase in the separation between them, in line with Nussinov's observations (63).

The ΔN6 structural variant assumed a large relevance as a model system to elucidate the mechanism of β2-m aggregation, with some researchers proposing an important role of this variant in the amyloidogenic mechanism of β2-m. Indeed, Radford and co-workers proposed a mechanism for β2-m amyloidogenesis in which the ΔN6 structural variant is responsible for inducing β2-m fibrillogenesis (67). At pH 7.5 and 25 ºC this truncated variant populates a conformational species that is structurally similar to the folding intermediate I_T, i.e. it retains the native fold and preserves the *trans* isomerization of Pro32, while simultaneously undergoing a major reorganization of several side chains within the hydrophobic core, particularly of Phe30 and Phe62 (20, 67).

According to Radford and co-workers, ΔN6 is capable of inducing the transition of the native full-length molecule to a fibril-competent conformation via a mechanism akin to prion conversion (Figure 7A). Indeed, Radford and colleagues reported evidence supporting the interaction between ΔN6 β2-m and the full-length β2-m, since they observed a conversion of wt β2-m into an aggregation prone conformer induced by bimolecular collision between the wt protein and the ΔN6 variant and cross-seeding (67). They also studied the mechanism of bimolecular collision by which the prion-like templating may occur using NMR. The results suggested that ΔN6 binds specifically and transiently to native wt β2-m and that residues of β-strands A, B and D and of DE-loop are possibly involved in this binding. This interaction leads to an increase in the conformational dynamics of the N-terminal strand, leading to a highly dynamic configuration of Pro14 in the AB-loop (20, 67), which has been shown to induce an alternative conformation in which the hydrogen bonding between β-strands A and B is critically impaired (87). The hydrogen bonding between β-strands A and B, together with the native N-terminal strand conformation, is essential for maintaining a low concentration of the intermediate I_T in equilibrium. Therefore, binding of ΔN6 to wt β2-microglobulin disrupts important interactions between the N-terminal strand and the BC-loop, facilitating the

isomerization of the His31-Pro32 peptide bond and the formation of the amyloidogenic intermediate state (20, 67).

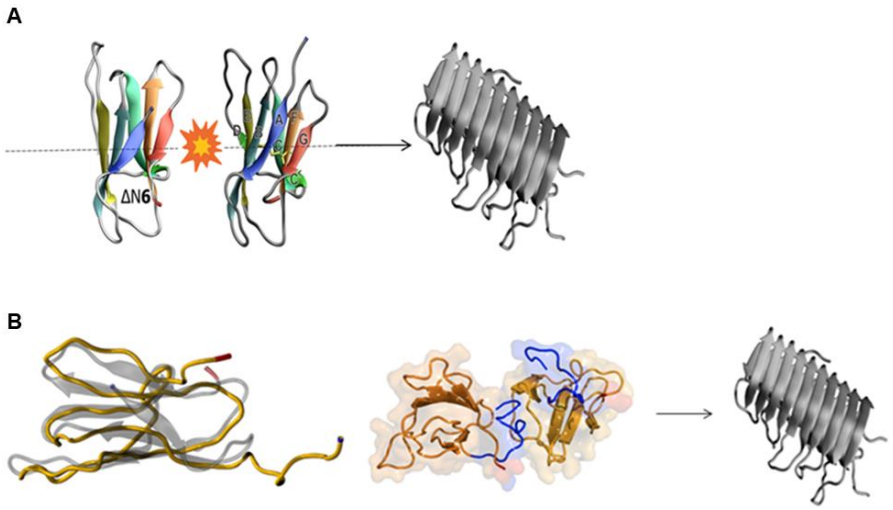


Figure 7. Mechanistic hypotheses for β2m aggregation based on the ΔN6 variant: prion-like hypothesis (A), and folding intermediate with an unstructured and detached A strand (B).

The evidence presented for such mechanism relied on the similarity between the ΔN6 and the I_T intermediate state and on the ability of ΔN6 to fibrillate *in vitro* under physiological conditions (67). According to this proposal, ΔN6 derives, at least partially, from the direct proteolysis of the full-length protein, an essential event to initiate fibrillogenesis in DRA. Then, this variant can populate rare conformers capable of nucleation and elongation because of its increased conformational dynamics (67). Additionally, a slight decrease in pH (from 7.2 to 6.2), which occurs in joints during inflammation (30, 35), will increase the population of these rare conformers by the destabilization of the ΔN6 conformation caused by the protonation of His84 (close to Pro32) (67). Indeed, this histidine undergoes a large pKa shift from 4.1 to 5.9 from the full-length to the truncated species, substantially increasing its protonation at pH 6.2 (19).

The higher affinity of ΔN6 to collagen (particularly when the pH lowers from 7.2 to 6.2 (30)), and its incorporation into fibrils could explain why the truncated variant is not detected in serum proteomic analysis (93).

Faísca *et al.* has also proposed a mechanism for β2-m aggregation based on the ΔN6 structural variant (57). Indeed, these authors identified and structurally characterized an intermediate state for folding and aggregation of ΔN6 using an integrative molecular

simulations approach comprising DMD, constant-pH molecular dynamics (CpHMD) and protein-protein docking (57). The identified intermediate is characterized by preserving the *trans* isomerization of Pro32 characteristic of the I_T intermediate and by having an unstructured strand A and a native-like core region comprising residues 21 to 94. Docking simulations predict a central role of the N-terminal region (strand A and AB-loop) in dimerization, acting as a "sticky hook" in dimer formation, the first step of aggregation (Figure 7B). Moreover, the strand A becomes maximally detached from the core region at pH 6.2, the pH of inflamed joints, which results in higher aggregation propensity as measured by the number of intermolecular contacts of the resulting dimers. This prediction rationalizes the higher aggregation potential observed *in vitro* at pH 6.2 and could also rationalize the pathogenesis of DRA as the slightly low pH of the synovial fluid can maximize the aggregation efficiency of the ΔN6 intermediate and hence contribute to the deposition of β2-m in the joints where it eventually forms the amyloid fibrils characteristic of DRA. However, the biological and clinical significance of this structural variant is not yet established as this variant, although being present in significant amounts in the *ex vivo* amyloid fibrils from DRA patients, it is not present in the blood of the patients (93). This observation raises questions concerning on whether the proteolytic cleavage of the six N-terminal residues occurs before or after fibril assembly as there is some evidence of a post assembly cleavage (94), which could limit the usefulness of the ΔN6 as a model system of β2-m aggregation.

1.3.4 The ΔK58 variant

Another structural variant of β2-m that has been receiving attention is ΔK58 (lacking residue Lys58), although the role of this variant in fibrillogenesis *in vivo* is not clear. Indeed, while it is present in the blood of many DRA patients (71) it is not present in the *ex vivo* amyloid fibrils (95). The ΔK58 variant can extensively fibrillate *in vitro* upon seeding with β2-m amyloid fibrils (70). It is less structurally stable than the wt β2-m (69, 70), populating an intermediate state with increased affinity for Congo red at physiological conditions (72), and having a considerably larger unfolding rate than wt β2-m (70). It also forms high molecular weight non-fibrillar aggregates when incubated *in vitro* at physiological unseeded conditions in contrast with wt β2-m (70). Notwithstanding, Ma and Nussinov (63) used MD to report that with the exception of an increased flexibility of the strands C' and D, the unfolding behavior of ΔK58 is similar to the one of wt β2-m, in sharp contrast to that of ΔN6. Therefore, although this structural variant could be a useful model system for studying β2-m folding and aggregation, its relevance in β2-m amyloidogenesis is still to be clarified.

1.3.5 Aggregation of β2-m *in vivo*

The uncertainty regarding the biological significance of the cleaved variants and the resilience of the wt protein to aggregate *in vitro* under physiological conditions led researchers to propose an alternative mechanism for β2-m aggregation exclusively based on the full-length protein in conditions that typically occur *in vivo* (e.g. presence of collagen and glycosaminoglycans) (19, 36, 39, 96). A specific environmental trait resulting from the presence of collagen and glycosaminoglycans is the existence of charge arrays that may have relevant effects on protein conformational stability by destabilizing the native structure (19) (Figure 8). Additionally, the positive charge arrays of the collagen surface can concentrate and orientate the negatively charged proteins generating a gradient of protein concentration in the vicinity of the collagen surface (19). Afterwards, under the high ionic strength existent in the highly concentrated and oriented layer of protein molecules close to the collagen surface, a spontaneous conformational transition may take place, characterized by the detachment of the N- and/or C-terminal strand and by the reorganization of the aromatic side chains in the hydrophobic core of the protein (19, 88). This transition gives rise to a fibril-competent species that is capable of nucleating an aggregate, which is a plausible event that does not require any partially processed β2-m species like the ΔN6 (19).

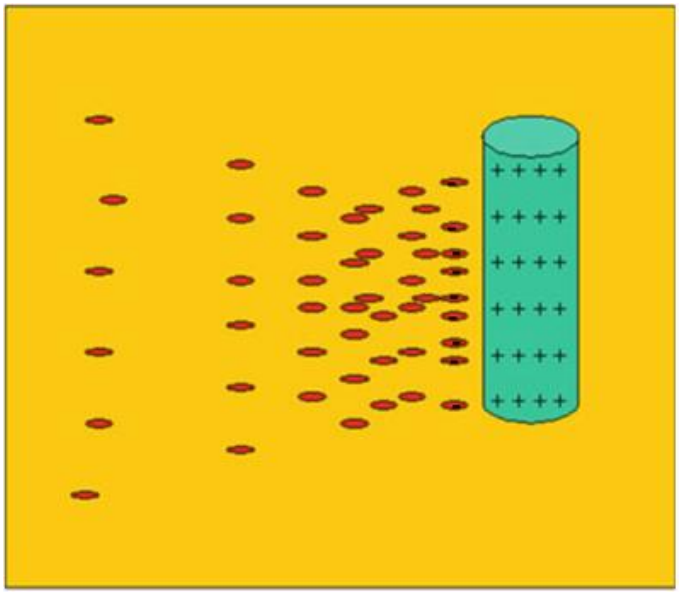


Figure 8. β2m aggregation mechanism based on the destabilizing effect of collagen charge arrays on β2-m (Adapted from Ref. (19)).

1.4. The D76N mutant: New insights into β2-m aggregation

The finding that the systemic amyloidosis caused by the D76N variant is not associated with an increase in the plasmatic concentration of β2-m (41) suggests that the mechanism of amyloidogenesis of this form could be somehow different from the mechanism of amyloidogenesis in DRA. Indeed, Bellotti and colleagues reported that this mutation has a destabilizing effect on the protein, causing a five-fold increase in the concentration of the I_T intermediate at equilibrium in a physiological buffer (97). This decreased stability of the D76N form combined with the shear stress in the extracellular matrix of visceral organs is sufficient to partially unfold the mutant protein and initiate a series of events leading to fibrillogenesis, particularly the exposure of normally hidden hydrophobic patches and the establishment and subsequent breaking of a condition of supersaturation whereby the previously soluble oligomers precipitate into insoluble aggregates (41).

The lack of deposition of β2-m fibrils in bones and ligaments in the systemic amyloidosis caused by the D76N mutant is explained by the fact that the preferential accumulation of β2-m on the collagen's surface only becomes significant for the micromolar concentrations observed during hemodialysis, meaning that the sub-micromolar physiological concentrations observed in the D76N β2-m amyloidosis are not sufficient to induce the deposition of β2-m fibrils in bones and ligaments (41).

In the last years, the D76N mutant of β2-m emerged as a biologically and clinically valuable model of β2-m folding and aggregation. Two reasons have fundamentally contributed for this. First, in contrast to what happens with the wt protein, the D76N readily aggregates *in vitro* under physiological unseeded conditions, which markedly facilitates the experimental study of its aggregation mechanism. Furthermore, since this mutant is the causative agent of a fatal hereditary systemic amyloidosis, it has a clear biological, biomedical and clinical relevance in contrast to what happens with the engineered point mutants, ΔK58 and ΔN6, used in the study of β2-m aggregation.

The D76N mutant has, therefore, been the subject of several experimental and computational studies aimed at explaining its increased aggregation propensity and the underlying aggregation mechanism (53, 73-76). Ricagno and co-workers (74) have shown that the deleterious effects of the D76N mutation are due to the crucial location of this residue in the EF-loop of the β2-m structure and not due to the removal of the negative charge or the change in the β2-m isoelectric point (pI) upon the substitution of an aspartate for an

asparagine. To reach this conclusion they constructed several D to N point mutants (at positions 34, 38, 53, 59, 96 and 98) and obtained high-resolution crystal structures of the mutants. They observed that the D to N mutations lead only to minor reorganizations of the side-chains in the vicinity of the mutated residues with the exception of the D38N, where they observed some backbone readjustments and a redistribution of the surface electrostatic charges. Additionally, they reported that although the thermal stabilities of the mutants change when compared to that of the wt, none of the mutants displayed the dramatic drop (~10ºC in the melting temperature (Tm)) observed for the D76N mutant. Moreover, none of the mutants aggregates *in vitro* at physiological pH contrarily to what happens with the pathogenic D76N mutant. However, in a later study (98) Ricagno *et al.* showed that any mutation in position 76 leads to a marked decrease of the thermodynamic stability, and to an increase in the amyloidogenicity of β2-m without major structural alterations. These results point to a crucial role of the mutation site in the determination of the protein's thermodynamic stability and aggregation propensity.

The basis of the increased aggregation propensity of the D76N mutant has been also studied by computational methods. For instance, Ham and colleagues (53) conducted MD simulations that indicate that the D76N I_T intermediate has an enhanced β-sheet forming propensity in its disordered D-strand, an increased SASA of the hydrophobic residues, and an increased solvation free energy in comparison with the wt I_T intermediate. Similarly, Rajasekaran and colleagues (76) also observed by MD that the D76N β2-m presents a longer β-strand D due to an inward movement of residue Asp53, local misfolding of all β-strands and turn regions due to the inability to form essential hydrogen bonds, and an increased flexibility of the DE-loop.

A recent study combining experimental and computational methods by Le Marchand *et al.* (73) shed new light into the molecular bases of the D76N increased aggregation propensity. Indeed, they identified a native-like conformational state of D76N with unstructured strands A and D, and a detached C-terminus exposing aggregation-prone regions like strands B, E and F. Furthermore, this intermediate exhibits increased conformational dynamics, particularly in the EF-loop (where the mutation is located), E-strand and in the end of the A-strand, and a disruption of a large network of electrostatic interactions involving the N- and C-termini and the EF-loop. This leads to the destabilization of the protein's native structure and to the exposure of the hydrophobic core to the solvent, rationalizing the higher aggregation propensity of the D76N mutant in relation to the wt β2-m. According to Le

Marchand and co-workers, the shear forces present in the extracellular fluid under physiological conditions would be enough to further unfold the termini and induce amyloid formation.

1.5. Early phase of ß2-microglobulin aggregation

1.5.1 Dimerization

In vitro experimental evidence by different groups suggests that the dimerization is the first phase of the β2-m aggregation mechanism (51, 99-101).

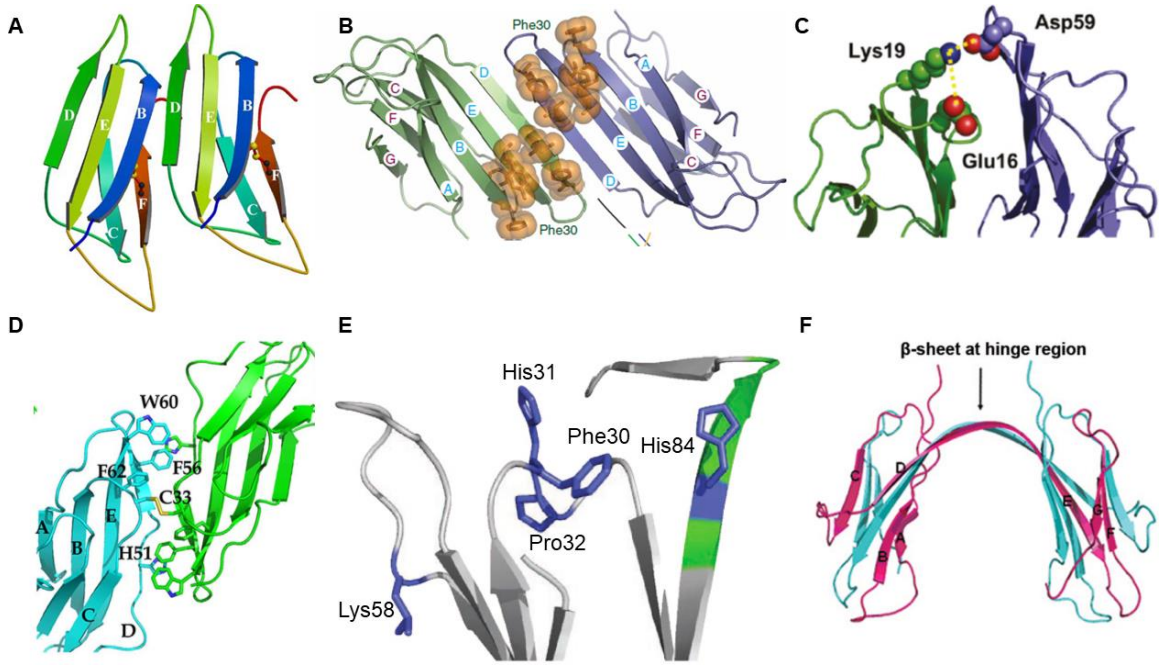


Figure 9. Representation of several dimer structures of β2-m experimentally and/or computationally characterized (Reproduced from (58, 68, 101-104)).

Early studies by Nussinov and co-workers (102, 105) used a combination of sequence and structural conservation analysis, and docking techniques to propose a model for β2-m fibrillogenesis based on stacking of β2-m monomers. In this model, monomers with native-like conformations associate via head-to-tail pairwise interactions originating a new inter-monomer β-sheet formed by the strand B of one monomer and the strand D of the other monomer (Figure 9A). Dimer stability is enhanced by the establishment of an aromatic cluster involving Phe56 from one monomer, and Phe30, His31 and Trp60 from the other monomer. Moreover, addition of further monomers in the same orientation elongates the inter-monomer sheet, which agrees with the cross-β model proposed for amyloid formation.

Subsequently, Eakin *et al.* (58) structurally characterized the dimer formed by the P32A mutant, a variant that mimics the effects of Cu^{2+} binding on β2-m structure, particularly

the conversion of the His31-Pro32 peptide bond to a *trans* isomer. They observed that the P32A dimer is assembled by antiparallel interactions between the two D-strands, yielding an eight-stranded ABED-DEBA β-sheet and forming a buried hydrophobic patch composed of Phe30, Leu54, Phe56, Phe62 and Tyr63 (Figure 9B). This dimer arrangement allows further oligomerization either by head-to-head (through the D-strands) or tail-to-tail (through the A-strands) interactions and is compatible with the typical dimensions of amyloid fibers.

Esposito and co-workers conducted a 5-ns MD simulation of an ensemble of 27 copies of β2-m in explicit solvent (106) and reported that Trp-60 and neighbour residues (Phe56, Lys58, Asp59) as well as N- (Ile1, Arg3) and C-terminal (Arg97, Asp98, Met99) residues are those that form more intermolecular contacts, and that hydrophobic interactions involving Trp60, Phe56, and Ile1 are essential in β2-m aggregation. This agrees with the results of Nussinov's group outlined above regarding the important role of the hydrophobic Trp60 and Phe56 in β2-m dimerization. Interestingly, these residues are predominantly localized in the apical regions of the protein relatively to the β-sandwich structure (N- and C-termini and DE-loop) and those of the DE-loop (e.g. Phe56, Trp60) have been implicated in aggregation in *in vitro* studies (59, 101, 103, 107, 108). Thus, the authors propose a mechanism of aggregation for β2-m in which the "sticky" apical ends of the protein associate, giving rise to transient complexes in a head-to-head arrangement in which the β-strands of different monomers are in an antiparallel and linear arrangement, in line with data from Eakin *et al.* (58). This conformational arrangement is not compatible with the cross-β structure of amyloid but the authors do not rule it out and put the hypothesis that the rearrangement of the intramolecular β-strand pairing requires a longer timescale that cannot be accessed by simulations.

Vachet and co-workers used a combination of computational and experimental methods including the docking method ZDOCK2.3, an energy minimization (EM) with explicit solvent, covalent labelling and mass spectrometry to study β2-m dimerization (103). They proposed that the β2-m dimer interface is formed by an antiparallel stacking of the ABED β-sheets from the two monomers, stabilized by electrostatic interactions between residues in the AB-loop and residues in the DE-loop (Arg12 with Tyr63 and Lys19 with Asp59) (Figure 9C), which resembles the results of Eakin and colleagues (58). A previous study by the same group (108) proposed that the establishment of the stabilizing salt-bridge between Asp59 and Lys19 is a consequence of a Cu^{2+} binding induced repositioning of Asp59. Analogously, the Cu^{2+} binding also dislocates Arg3 at N-terminus, allowing it to establish a stabilizing salt bridge with Glu16 in the AB-loop. The involvement of the AB-loop

in dimerization was also shown by Colombo *et al.* (109) in an engineered disulphide-linked homodimer generated by the mutation of the two serines at position 20 to cysteines.

Halabelian *et al.* (101) proposed a dimerization mechanism mediated by the association of the BC-loop, strand D, DE-loop and the E strand of each monomer after constructing and structurally characterizing a covalent homodimer obtained by the mutation of the serine at position 33 to a cysteine. The generated mutant dimer maintains the normal fold of each monomer while locking the observed association interface by a disulphide bridge between the mutated cysteine 33 residues. These observations recapitulate the importance of residues located on the BC and DE-loops in β2-m dimerization (Figure 9D) (58, 102, 103, 105, 106), and also a similar study from Colombo *et al.* (109) supporting the involvement of the D- and E-strands in the interfaces of disulphide-linked covalent homodimers. Radford and co-workers (67) have also suggested that ΔN6 dimerization proceeds through interactions involving the BC- and DE-loops, based on NMR data.

Given the importance of the Cys25-Cys80 disulphide bond on β2-m structure, Dokholyan and colleagues (110) studied the effect of the Cys25-Cys80 disulfide bond on β2-m oligomerization using MD. They found that, under oxidizing conditions (i.e. when the disulfide bond is formed), β2-m originates domain-swapped dimers in which the two monomers exchange their N-terminal segments. Dimerization by domain-swapping appears to be a relevant mechanism in β2-m aggregation. Indeed, Domanska and colleagues (68) detected and structurally characterized a domain-swapped dimer of the ΔN6 variant. In this dimer, the C-terminal G-strand is exchanged between the two monomers as the result of the partial unfolding induced by the Pro-32 *cis* to *trans* switch, and of the rotation of Phe-30 towards the solvent (Figure 9E). Similarly, Eisenberg and co-workers (104) structurally characterized a β2-m domain-swapped dimer in which the β-strands E, F and G are exchanged between the two monomers (Figure 9F).

In contrast, under reducing conditions (i.e. that disrupt the disulfide bond) β2-m forms dimers and trimers characterized for having parallel β-sheets between monomers, and for being stabilized by the hydrogen bond network along the backbone. Thus, the Cys25-Cys80 disulfide bond appears to modulate the type of oligomeric precursors that are formed under oxidizing and reducing conditions. Moreover, these differences in the aggregation mechanism may explain the differences observed in the amyloidogenic behaviour at oxidizing and reducing conditions, with the oxidized β2-m forming amyloid fibrils at pH 2.5 while the

reduced β2-m is not able to form typical amyloid fibrils at this acidic pH, originating instead thinner and more flexible filaments.

1.5.2 Tetramerization and beyond

The formation of tetramers from dimers appears to be essential for β2-m amyloidogenesis. Indeed, while several studies point to the existence of other oligomeric species in the way to β2-m fibrillogenesis (i.e. trimers and pentamers) (111, 112), dynamic light scattering data obtained from Vachet and co-workers in the presence of Cu^{2+} suggested the presence of only even-numbered oligomers formed through the addition of dimeric units (113). This originates soluble tetramers and hexamers prior to the appearance of insoluble aggregates and amyloid fibrils. Moreover, White *et al.* (114) studied the architecture of the β2-m amyloid fibrils by cryo-electron microscopy and proposed that the basic assembly units of the fibril protofilaments are tetramers obtained by a dimer-of-dimers arrangement.

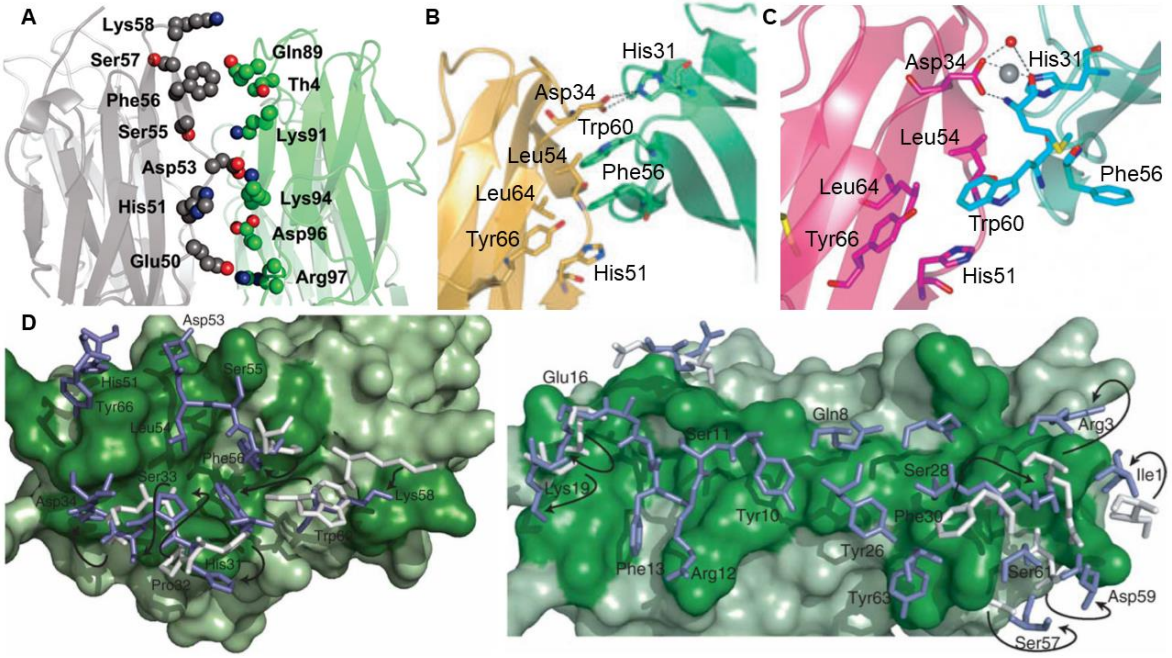


Figure 10. Representation of several tetramer (A, B and C) and hexamer (D) structures of β2-m experimentally and/or computationally characterized (Reproduced from (59, 109, 115)).

Motivated by these results, Vachet and co-workers extended their study of β2-m dimerization to the tetramerization phase (115). The tetramer model they proposed is characterized for having a tetramer interface formed by the D-strands of one dimer and the G-strands of the other dimer, in which salt bridges between residues Glu50 and Arg97, His51 and Asp96 and Asp53 and both Lys91 and Ly94 are essential for tetramer stabilization

(Figure 10A). On the other hand, Colombo *et al.* structurally characterized the tetramers formed by the disulphide-linked covalent homodimers DIMC20 and DIMC50 (formed by mutations of the residues at positions 20 and 50, respectively, to cysteines) and verified that tetramerization proceeds through the association of the D-strands of each dimer (109). Indeed, they observed that the interface of DIMC50 tetramer comprises the BC-loop, the D-strand, the DE-loop and the E-strand of the facing chains, with His31, Asp34, His51, Phe56 and Trp60 being the essential residues in the tetramer interface. Particularly, Phe56 and Trp60 from one subunit are inserted in a hydrophobic pocket formed by Leu54, Leu64 and Tyr66 of the other subunit while a hydrogen bond between His31 of one subunit and the Asp34 of the other is established (Figure 10B). Similarly, the interface of the DIMC20 tetramer involves the same regions and residues as the DIMC50 tetramer with only a few differences (e.g. the solvent exposure of one of the Phe56 instead of its location in the hydrophobic pocket formed by Leu54, Leu64 and Tyr66 of the other subunit) (Figure 10C). In agreement with this study, Miranker and co-workers (86) gathered evidence that the interface of the Cu^{2+}-bounded tetramer of the P32A mutant of β2-m is mediated by the D-strands and possibly also by the neighbouring DE-loops.

Considering the possible role of hexamers in the β2-m aggregation cascade, Calabrese *et al.* (59) structurally characterized a hexamer of the Cu^{2+}-bounded β2-m H13F point mutant where one of the interfaces is mediated by the interaction of D-strands from adjacent monomers and the other is mediated by the stacking of the ABED sheets from two adjacent chains. The first interface has a surface area of 1340 $Å^2$ and is formed as a result of a displacement of residues Phe56 (3.1 Å) and Trp60 (8.0 Å) upon Cu^{2+} binding. It comprises hydrogen bonds between the Leu54 backbone and the Asp34 and His31 side chains, and hydrophobic interactions between Phe56 and Trp60 of one monomer and the nonpolar atoms of His51 and Asp34 of other monomer (Figure 10D). The second interface has a surface area of 1950 $Å^2$ and is characterized by an antiparallel arrangement of the strands in which each strand approximately opposes its counterpart from the other chain (e.g. A:A, B:B). Its interface core comprises both aromatic and polar interactions between tyrosines 10, 26 and 63 and includes also the residues Ile1, Arg3 and Phe30 of the Cu^{2+} binding site. These residues are displaced from their positions in the wt Cu^{2+}-free state, suggesting an important role of Cu^{2+} in β2-m oligomerization (Figure 10D).

Chapter 2. Integrative Modeling approach for studying β2-m folding and aggregation

The goal of this PhD research project is to explore with molecular simulations the early phase (dimerization and tetramerization) of the aggregation mechanism of β2m by considering the D76N mutant and the ΔN6 variant as model systems of β2m amyloidogenesis. Our working hypothesis is that the protein populates intermediate states *en route* to folding with the ability to trigger the aggregation pathway. Accordingly, we follow an integrative computational approach that comprises three main steps: (1) study of the folding transition with a full atomistic native-centric Gō model combined with replica-exchange DMD simulations; (2) CpHMD simulations with explicit titration to access the effect of pH on monomer's structure; and (3) Monte Carlo ensemble docking (MC-ED) simulations to study protein-protein association (Figure 11).

2.1. The folding space of β2-m explored with DMD simulations

The equilibrium folding space of the considered model systems is explored with replica-exchange DMD simulations of a full atomistic protein representation, combined with a simple, structure-based Gō potential. Simple Gō potentials are native centric, which means that protein folding energetics is exclusively driven by native interactions. Since they do not incorporate non-native interactions, Gō potentials will not be able to capture misfolding processes leading to compact non-native states or, more generally, regions of the folding free energy landscape where non-native interactions play a determinant role (e.g. the denatured state). Thus, they can only detect native-like intermediates. However, the adopted level of structural resolution encompasses the effect of detailed atomic contacts of the native structure in the folding mechanism. This is crucial to correctly evaluate the impact of single point mutations on folding pathways (i.e. to get a realistic comparison of the folding pathways of proteins with very similar native structures). Furthermore, side-chain packing, a fundamental ingredient of the folding process, is fully taken into account. The intermediate states we identified in our previous study and in the current work (57, 116) are native-like in the sense that they exhibit a well-preserved native core, but feature unstructured termini. We then computed the SASA of the identified intermediates to predict their aggregation potential.

2.2. Structure refinement with CpHMD simulations

pH has a relevant role in protein aggregation and, in particular, in β2-m amyloidogenesis. Indeed, pH controls the charge of the ionizable side-chains, thereby modulating the pattern of electrostatic interactions, which influences protein aggregation by inducing minor structural rearrangements, or larger scale structural alterations like modifications of secondary structure (61). Additionally, the charge of ionizable side-chains also influences the pattern of interfacial electrostatic interactions established upon protei-protein association, modulating the stability of the generated oligomers.

The effect of pH on the structure of the identified folding intermediates is assessed with CpHMD simulations with explicit titration starting from conformations representative of the intermediates. These simulations generate ensembles of conformations representative of each intermediate state at a specific pH, i.e. whose structure and charge pattern have been modulated by pH. Additionally, the ensembles of conformations have a higher structural accuracy than the ones obtained with the Gō model, particularly at the level of the dihedral angles. This is important as the *in-house* developed Monte Carlo ensemble docking (MC-ED) method used to study the initial stage of β2-m aggregation naturally relies on the structural accuracy of the input monomers.

2.3. Dimerization phase analysed with protein-protein docking simulations

The ensembles of conformations generated in 2.2 are used in the third stage of the adopted procedure, which is the study of dimerization via protein-protein docking. The outcome of docking simulations is an ensemble (typically containing 1000 conformers) of statistically representative (homo- or hetero-) dimers formed by monomers of intermediates under different pH conditions (e.g. an ensemble of dimers of I_1-I_1 intermediates at pH 5.2, an ensemble of dimers of I_1-I_2 intermediates at pH 7.2, etc.). We perform a statistical analysis over the ensembles of dimers in order to get information about the triggers of dimerization, i.e. the most likely regions involved in the process and, at a finer structural level, the residues that will most likely establish a larger number of intermolecular interactions acting as aggregation hot-spots. The operational implementation of the last methodological stage is described in Supplementary Figure 5.

In adopting this approach we are not considering the possibility of protein association occurring concurrently and concomitantly with folding, a situation that would lead, e.g., to domain-swapped dimers (68). Instead, we are considering the scenario according to which

protein association occurs upon the formation of intermediate states *en route* to the native state, which have the potential to trigger the aggregation pathway because they are aggregation prone, and whose thermodynamic stability is large enough to guarantee a timespan compatible with the establishment of intermolecular interactions.

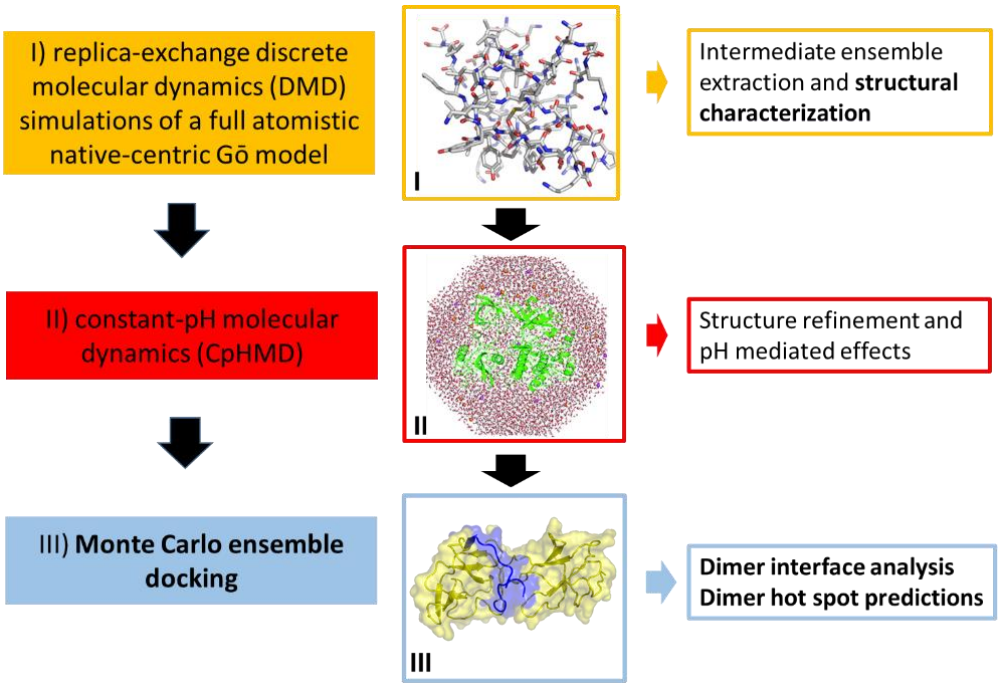


Figure 11. Representation of the stages of the methodological approach followed in this Project.

Chapter 3. Intermediate states for folding and aggregation

3.1. Structural clustering

MD simulations of proteins generate large amounts of data that describe in detail the MD trajectories (117). The latter comprise the sequential time-dependent sets of protein conformations (i.e. the position coordinates of all the protein's atoms) together with physical and geometrical properties (e.g. the energy, bond lengths, gyration radius and root-mean-square deviation (RMSD) (117)) calculated at each time step of the simulation. There are inherent relationships between subsets of the sampled conformations that determine their structural similarity, which are often hidden by the complexity of the data (117). An efficient way to expose these relationships is to cluster protein conformations into subgroups based on their structural similarity (e.g. as measured by the RMSD to a reference structure) (117-119).

Clustering methods are a class of data-mining techniques that can be applied to a given collection of data elements to unveil and visualize subgroups of elements sharing similar properties (117, 120). These techniques can be applied to any collection of data elements characterized by some property that can be measured and compared between pairs of different elements (e.g. RMSD of a protein conformation to its native structure) and generate disjoint sets of elements called clusters (117, 121, 122). A cluster is characterized by a higher similarity of its elements to each other than to the elements of the other clusters (117, 121, 122). By using clustering algorithms one can, in principle, identify and characterize distinct conformational states populated by a given protein, which are represented by different clusters whose elements share a similar structure (117, 123). By focusing on the representative conformation of each state, which corresponds to the average structure of each cluster, these procedures reduce the variance of the conformations to analyse from each MD.

There is a wide variety of clustering algorithms that can be applied to analyse MD trajectories (117, 122). These algorithms are usually classified in three types: 1) top-down or hierarchical clustering, 2) bottom-up or agglomerative clustering (single-linkage/edge-joining, centripetal, complete-linkage, centroid-linkage, average-linkage and centripetal-complete), and 3) refinement clustering (k-means, Bayesian and self-organizing maps (SOM)).

3.1.1. Top-down or hierarchical clustering

The top-down or hierarchical clustering starts by creating a single large cluster to which all elements are assigned (117, 124). Subsequently, the algorithm consecutively divides the largest cluster into two smaller clusters until the desired number of clusters, which is

defined *a priori*, is reached (117, 124) (Figure 12). The main advantages of hierarchical clustering methods is that they are the fastest of all clustering algorithms when the number of clusters is small, and that changes in metrics, such as the variance explained by the data (i.e. ratio between the sum of the variances within all clusters and the total variance of the sample), dependent on the number of clusters, are easy to interpret (117). A major drawback is that they only produce clusters with similar diameters (i.e. the distance between the two farthest apart elements) and avoid clusters with different diameters that may correspond to local energy minima of different depths (117, 124). Another disadvantage is that data elements are only reassigned to the two clusters generated at each iteration, (117, 124), which does not allow for the correction of errors in the partition of data generated in previous iterations. Also, the hierarchical clustering methods are very sensitive to outliers (117, 124).

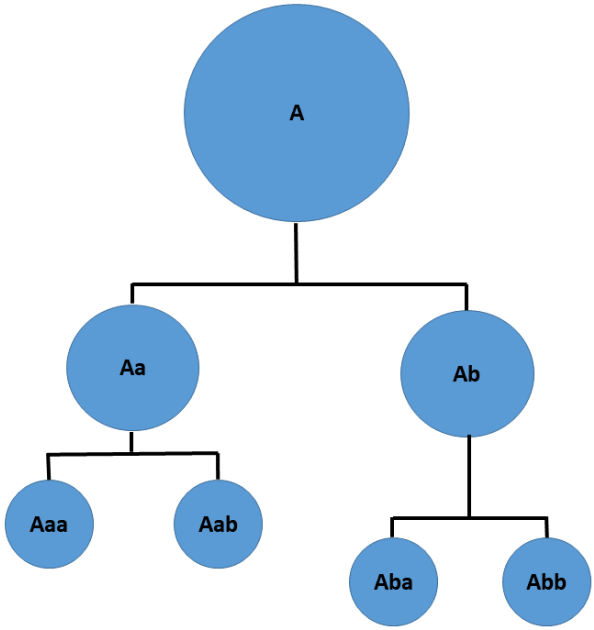


Figure 12. Representation of the principle of top-down hierarchical clustering. At each step the largest cluster is divided in two smaller clusters until the desired number of clusters, defined *a priori*, is reached.

3.1.2. Bottom-up or agglomerative clustering

The bottom-up or agglomerative clustering begins by creating a cluster correspondent to each data element and proceeds by iteratively merging two clusters until the desired number of clusters, defined *a priori*, is reached (117, 124) (Figure 13). The differences amongst the algorithms belonging to this class stand on the different criteria used to choose the pairs of clusters to merge, and from using different definitions of intercluster distance (117). Their main advantage is that the clustering merging information can be saved at each

iteration, providing the set of distinct clusters generated in individual runs arising from different initial choices of the number of clusters to form (117, 124). This information can be analysed to provide useful insights into the appropriate number of clusters for the data. The main limitation is their tendency to group most of the data into a single large cluster originating small singleton clusters with only one or few data elements (117, 124).

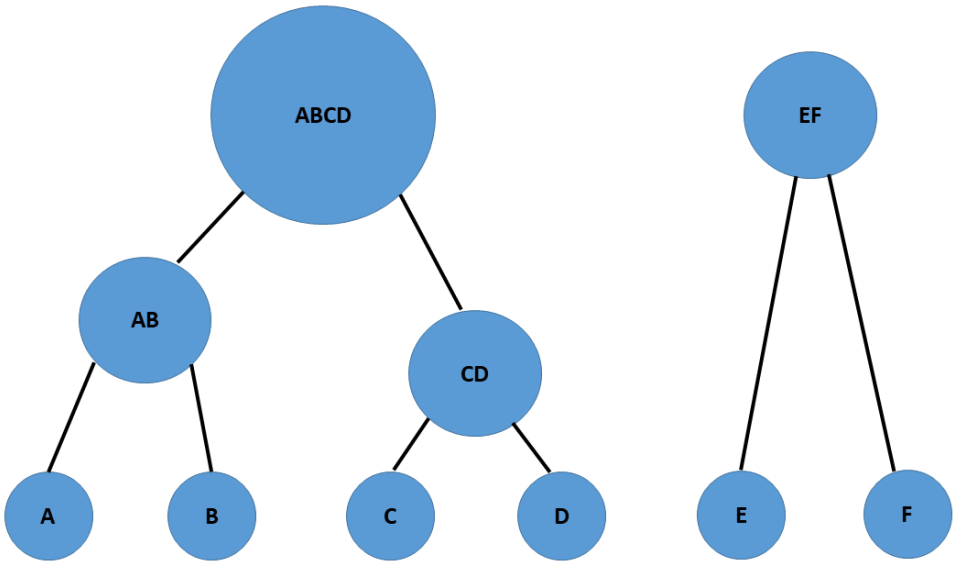


Figure 13. Representation of the principle of bottom-up agglomerative clustering. The algorithms of this class begin by creating a cluster correspondent to each data element and proceed by iteratively merging two clusters until the desired number of clusters, defined *a priori*, is reached.

3.1.3. Refinement clustering

The refinement clustering starts by creating seed clusters (i.e. clusters defined at the beginning - different algorithms use different criteria to create these initial clusters) that are iteratively refined by the algorithm through the optimization of the distribution of the data elements between the different clusters (117). This process proceeds until the clustering results are stable enough (i.e. the partition of the data elements between the different clusters do not change significantly) (117, 120). The number of final clusters is determined *a priori* by the number of seed clusters, and usually does not change during the refinement process (117, 120). The main advantage of refinement clustering methods is their relative speed in comparison to other types of clustering methods, requiring less iterations to generate the final results (120), while having good performance metrics in the clustering of MD trajectories data (117). However, these methods have some tendency to generate homogeneously sized clusters (117).

3.1.3.1. The k-means clustering algorithm

We used the *k-clust* implementation of the k-means refinement clustering algorithm available in the MMTSB Tool Set (http://feig.bch.msu.edu/mmtsb/Main_Page) (125) to cluster conformations obtained from DMD of the folding transition of protein β2m. This is a necessary step of our methodological approach that allows isolating and structurally characterizing potentially aggregation-prone folding intermediate states. We choose this algorithm because it is relatively fast (120) and displays good performance metrics in the analysis of MD trajectories (117).

The k-means clustering algorithm starts by choosing a set of k randomly placed seed centroids, which represent the center of each cluster to be formed (117, 120). The clusters are formed by assigning the data points to their closest centroid, and the centroids are then moved to the average location of the points assigned to them. Afterwards the assignments are redone (117, 120). This process repeats iteratively until the clustering partitioning becomes stable (i.e. the assignments of the data points to the different clusters stop changing) (120) (Figure 14).

The *k-clust* tool of the MMTSB Tool Set (125) implements the k-means clustering algorithm with the particularity of restricting the clustering radius (the maximum RMSD value that the structures in one cluster can have in relation to the cluster centroid) instead of defining the number of initial seed points.

While preparing the DMD conformations for the clustering procedure, it is necessary to obtain the mirror image of each conformation. Indeed, we have to compute the RMSD of each conformation to the native structure as well as that of its mirror image to check which one has the lowest RMSD. This step is needed to adequately compare the structures in the clustering procedure because without this step the reflected (i.e. mirror image) structures would be considered structurally different and hence wrongly assigned to separate clusters.

In order to use the *kclust* tool, we need to decide the radii of each cluster by bearing in mind that, when we decrease the cluster radius, we increase the number of generated clusters, which allows a more fine-grained study of the conformational ensemble. This comes, however, with an increase in the computational cost of the algorithm (i.e. it increases the number of pairwise distances between the data points and the centroids that have to be measured and compared) whereby we have to choose the minimum value of the clustering radius that gives a manageable number of clusters (usually 10 clusters at maximum).

At the end of the clustering procedure we have to extract representative conformations of the obtained clusters. The latter will provide a structural representation of the folding intermediates. In order to do so, we choose as representative structure the one within the DMD dataset that is the most similar to the centroid of each cluster. We stress that the cluster centroids are not conformations of the DMD trajectories; they are, instead, structures created by the clustering algorithm featuring the average properties of each created cluster.

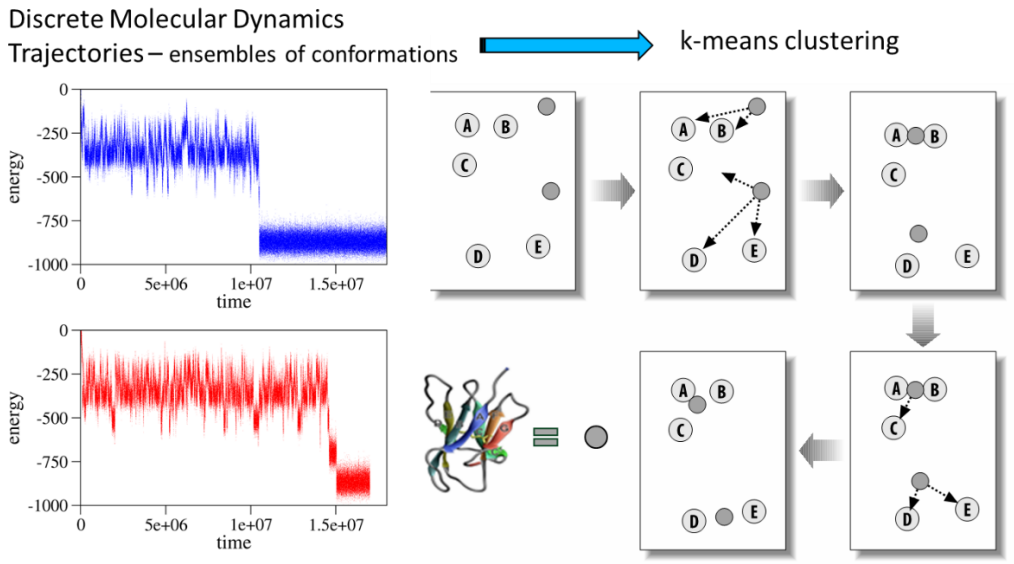


Figure 14. Representation of the procedure used for clustering DMD conformations based on k-means clustering. The algorithm starts by choosing a set of seed centroids (by restricting the clustering radius) and then it assigns the data points (i.e. protein conformations generated in DMD) to their closest centroid. Afterwards, the centroids are moved to the average location of the points assigned to them and the assignements are redone. This process repeats iteratively until the clustering partitioning becomes stable.

To further structurally characterize the identified intermediate states, we have computed the SASA per residue of these intermediate states with the algorithm NACCESS Version 2.1.1 (Hubbard and Thornton, 1992–6). This algorithm is an implementation of the Lee and Richards (126) method that computes the atomic accessible surface area defined by the center of a probe of a given size (usually of the radius of a water molecule, 1.4 Å) rolling around a van der Waals surface. It produces a series of thin slices through the 3D volume of the macromolecule and sums the accessible surface area of each slice to obtain a fairly accurate approximation of the total surface area for each atom.

3.2. Intermediate states in the folding space of β2-m

The first stage of the adopted methodology is the exploration of the folding space of D76N-β2-m with equilibrium replica-exchange DMD simulations of a full atomistic structure-based (SB) Gō potential, i.e. a potential that only considers the stabilizing native interactions (127, 128). This step was not performed by myself. The simulation data is analysed with the WHAM (weighted histogram analysis method) in order to compute free energy surfaces (i.e. the projection of the free energy on selected reaction coordinates (namely energy, E, gyration radius, R_g, and root-mean-square deviation to the native structure, RMSD) at some selected temperature. The free energy surfaces at the transition temperature T_f (i.e. the temperature at which the native and denatured state both have the same free energy) highlight the existence of additional basins besides the native basin and denatured state basin, which highlight the population of intermediate states of β2-m (Figure 15B).

One of the basins (E ~ -700, RMSD ≤ 10 Å) is present in the folding space of both mutant and wt protein and conformations therein are reached through thermal fluctuations of conformations representative of the native state (E ~ -900, RMSD <5Å), while the folding space of the D76N mutant exclusively features another basin (E ~ -500, RMSD ~ 17 Å), populated by conformations that may represent potential aggregation-prone states.

The ensemble of conformations populating each basin represents an intermediate state that will be isolated by structural clustering. We term the representative conformation of the first basin I_1, while the representative conformation of the second basin is termed I_2.

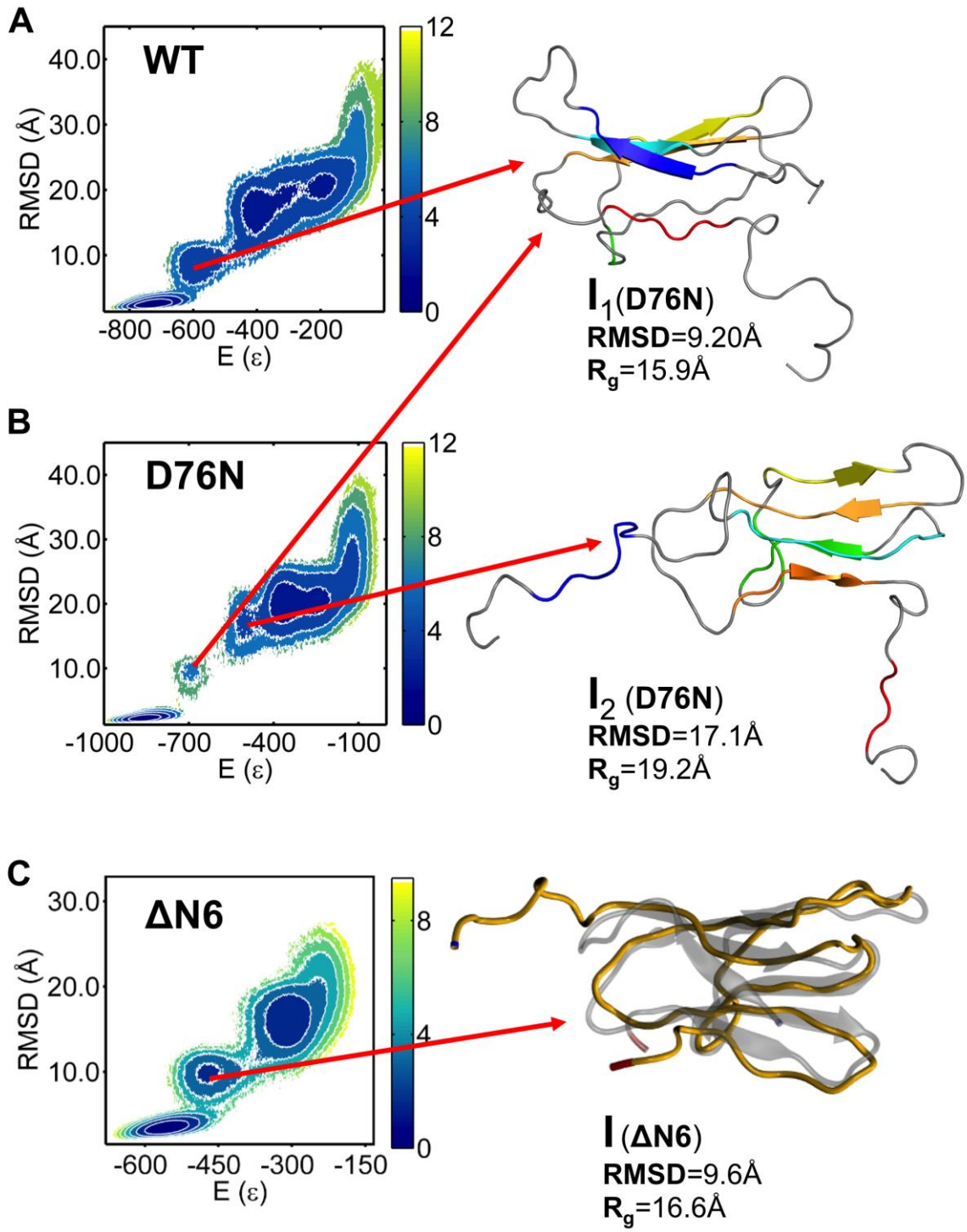


Figure 15. Folding intermediates. (B) The two intermediate states populated by the D76N mutant mapped on the folding free energy landscape, which shows a projection of the free energy on the energy and RMSD to the native structure. The color code represents the free-energy (in units of energy). Both intermediates feature a well-preserved core (strands B-F). While intermediate I_1 displays an unstructured C-terminal region (C-terminus and G-strand), in intermediate I_2 both terminal regions are unstructured. I_1 is also populated by the wt variant (A). (C) The intermediate state populated by the

ΔN6 variant, in which the core is also preserved but the N-terminal region (N-terminus and A-strand) is unstructured.

In order to identify and structurally characterize I_1 and I_2 we performed extensive structural clustering over ensembles of conformations collected from DMD simulations at fixed temperature (T_f) by using the k-means algorithm. The analysis of the representative conformations shows that the intermediate I_1 presents an unstructured and detached C-terminus (Figure 15A and Figure 15B) while the I_2 intermediate exhibits both termini unstructured and detached from the protein's core (Figure 15B). Additionally, both intermediates present a *trans* isomerization of the His31-Pro32 peptide bond and show a well-preserved core ($RMSD_{21-83} < 3.2$ Å). The evaluation of the SASA per residue shows that 76% of the most hydrophobic residues become solvent exposed in I_2 while this number drops to 53% in I_1 (Figure 16A and Figure 16B). This observation suggests that I_2 is more aggregation-prone than I_1.

The identification of intermediate states in the folding space of D76N-β2-m follows a previous study by Faísca *et al.* focusing on the folding space of ΔN6 (57). The latter populates an intermediate state termed I (E ~ -500, RMSD ≤ 10 Å) (Figure 15C). This intermediate is topologically similar to I_1 as it features a well-preserved core ($RMSD_{21-94}$ ~3.2 Å) and a detached and unstructured terminus. However, in I the detached and unstructured terminus is the N-terminus instead of the C-terminus. The evaluation of the SASA of this intermediate indicates that 62% of the most hydrophobic residues become solvent exposed in I, an intermediate value between those of I_1 and I_2 (Figure 16C).

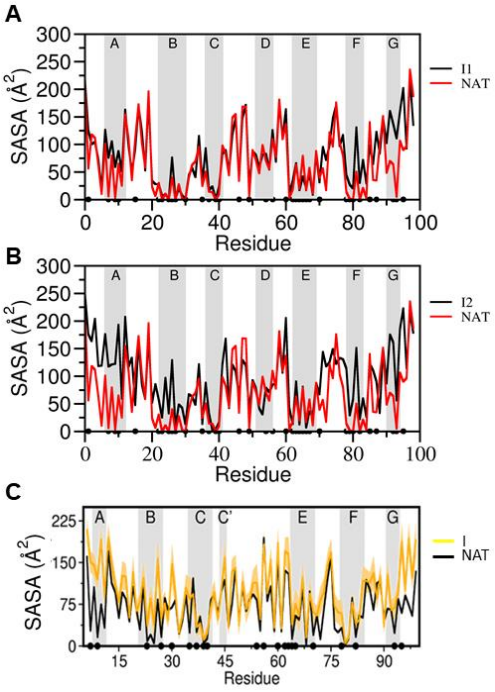


Figure 16. Solvent accessible surface area (SASA) of the residues in I_1 (A) and I_2 (B) folding intermediates of D76N β2-m and in folding intermediate I of ΔN6 β2-m (C). The hydrophobic residues are represented by black circles.

3.3. Effect of pH on the structure of the intermediate states

To determine how the pH affects the structure of the identified intermediate states and construct ensembles of monomers representative of the intermediates at acidic (5.2) and physiological (7.2) conditions, our collaborators at Miguel Machuqueiro's group conducted CpHMD simulations starting from conformations representative of I_1 and I_2. Additionally, CpHMD simulations starting from the native structures of the D76N and of the wt β2-m were also conducted for comparison purposes. As mentioned before, this procedure is also important to obtain monomeric structures with a higher structural accuracy than the ones generated in the DMD simulations, which lack structural accuracy at the level of dihedral angles. Indeed, the generation of high accuracy monomer structures is essential for the protein-protein docking analysis.

Besides running CpHMD simulations at pH 5.2 and 7.2, simulations at the slightly acidic pH 6.2 were also conducted in order to compare the results for D76N with those previously obtained by Faísca *et al.* for the ΔN6 truncated variant (57).

The analysis of the CpHMD simulations indicates that the D76N mutation increases the isoelectric point (pI) by ~0.5 pH units (Figure 17), which appears to be the result of

mutating an acidic residue with a neutral one. At physiological pH, the wt protein is more negative than the mutant. This suggests that the protein will become more prone to aggregate upon mutation as the presence of charged residues prevents aggregation due to electrostatic repulsion. Furthermore, this tendency will be higher at the slightly acidic pH around the pI values (6.40-6.66).

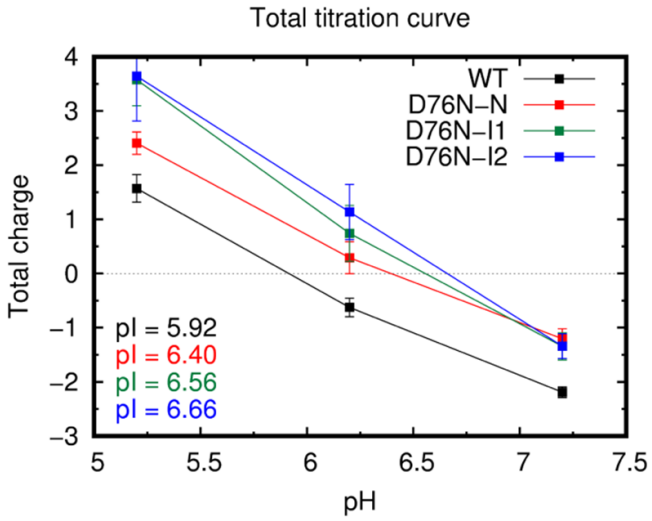


Figure 17. Total titration curves obtained from the CpHMD simulations with the isoelectric points (pI) shown.

At slightly acidic pH, histidine residues are usually protonated, contrary to what usually occurs at a physiological pH. There are four histidines in β2-m and two of them (31 and 84) are more internalized and interact with each other. The other histidines are solvent exposed having regular pKa values in all conformational states (Table 1 and Table 2). His13 is located in the N-terminus, which is detached in I_2, allowing this residue to interact with several neighbouring residues and lowering its pKa (Table 1). Similarly, His13 also has a slightly lower pKa (6.0) in the intermediate I (with a detached N-terminus) than in the native state of ΔN6 (6.4) (Table 1), which is likely the result of its increased solvent exposure (57). Conversely, His31 and His84 have pKa values considerably shifted to lower pH, triggered by the low solvent exposure and the presence of proton donors for hydrogen bonding. The I_1 and I_2 intermediates disrupt the hydrogen bonding network and expose these histidines to the solvent which renders their pKa values less shifted to low values, especially in I_2, where both termini are detached.

Residue	wt	ΔN6	I	D76N	I_1	I_2
His 13	6.3±0.0	6.4±0.1	6.0±0.1	6.4±0.0	6.4±0.0	5.3±2.8
His 31	4.8±0.6	5.4±0.2	5.2±0.4	4.1±3.1	5.6±0.3	6.2±0.3
His 51	6.6±0.0	6.2±0.2	5.7±0.2	6.6±0.1	6.6±0.1	6.5±0.1
His 84	<3	<3	<3	<3	5.2±0.9	5.8±0.6

Table 1. pKa values of key histidine residues calculated from the CpHMD simulations.

Residue	wt	D76N	I_1	I_2
His 13	0.56±0.07	0.50±0.13	0.55±0.12	0.45±0.19
	0.56±0.07	0.58±0.07	0.63±0.15	0.51±0.16
	0.56±0.07	0.62±0.09	0.67±0.08	0.52±0.18
His 31	0.40±0.19	0.32±0.06	0.43±0.19	0.44±0.22
	0.30±0.08	0.36±0.09	0.42±0.13	0.57±0.16
	0.31±0.03	0.38±0.09	0.46±0.11	0.58±0.15
His 51	0.59±0.03	0.57±0.06	0.55±0.11	0.59±0.11
	0.59±0.02	0.53±0.09	0.59±0.11	0.53±0.14
	0.61±0.03	0.61±0.12	0.53±0.16	0.51±0.13
His 84	0.21±0.04	0.19±0.01	0.44±0.21	0.43±0.14
	0.21±0.04	0.23±0.08	0.34±0.16	0.57±0.20
	0.18±0.01	0.21±0.07	0.51±0.20	0.53±0.14
Trp 60	0.56±0.14	0.62±0.03	0.66±0.19	0.59±0.15
	0.63±0.07	0.59±0.08	0.64±0.09	0.66±0.11
	0.64±0.06	0.60±0.16	0.67±0.09	0.71±0.14

Table 2. Relative solvent accessible surface area (SASA) values of key residues calculated from the cpHMD simulations. The 3 pH values (5.2, 6.2 and 7.2) are reported in each cell.

In what regards the effect of pH on the mobility of the different protein regions, we observe that the intermediate I_2 of D76N presents large deviations (up to ~20Å) of the two terminal regions both at neutral and acidic pH (Table 4), in line with results reported by Le Marchand (73). It also exhibits significant deviations of the DE-loop and EF-loop from their native position (up to ~9Å) across the investigated pH values (Table 4). As for I_1, we highlight a striking mobility of the C-terminus (up to ~20Å) while the N-terminus, DE-loop and EF-loop present more conservative motions (between 5.4 and 7.6Å) (Table 3). As for the intermediate I of ΔN6, the slightly acidic pH 6.2 induces significant deviations (~16Å) of the N-terminal region comprising strand A and the AB-loop relative to the native structure (Table

5), likely as the result of the increased protonation of His13 (AB-loop) caused by the similarity between the medium pH and the pKa of His13 imidazole ring (57).

pH	Cα RMSD (Å)	Cα RMSD $_{6-83}$ (Å)	N-terminus (A-strand + AB-loop) Cα RMSD (Å)	BC-loop Cα RMSD (Å)	DE-loop Cα RMSD (Å)	EF-loop Cα RMSD (Å)	C-terminus Cα RMSD (Å)
5.2	8.15 ± 0.13	3.20± 0.05	7.59±0.12	4.17±0.07	6.58±0.10	7.47±0.12	20.45± 0.32
6.2	7.27±0.11	2.79±0.04	7.03±0.11	3.31±0.05	4.98±0.08	6.42±0.10	18.43±0.29
7.2	7.67±0.12	2.77±0.04	6.23±0.10	3.14±0.05	5.42±0.09	5.90±0.09	19.40±0.31

Table 3. Cα RMSD of the full I_1 intermediate as well as of specific protein regions in relation to the native structure.

pH	Cα RMSD (Å)	Cα RMSD $_{21-83}$ (Å)	N-terminus (A-strand + AB-loop) Cα RMSD (Å)	BC-loop Cα RMSD (Å)	DE-loop Cα RMSD (Å)	EF-loop Cα RMSD (Å)	C-terminus Cα RMSD (Å)
5.2	10.62 ± 0.17	3.16± 0.05	17.99±0.28	5.67±0.09	8.27±0.13	7.23±0.11	19.65± 0.31
6.2	10.90±0.17	4.00±0.06	21.21±0.34	9.04±0.14	9.30±0.15	6.99±0.11	16.23±0.26
7.2	10.69±0.17	3.57±0.06	17.90±0.28	5.92±0.09	9.07±0.14	7.69±0.12	19.50±0.31

Table 4. Cα RMSD of the full I_2 intermediate as well as of specific protein regions in relation to the native structure.

pH	Cα RMSD (Å)	Cα RMSD $_{21-94}$ (Å)	N-terminus (A-strand + AB-loop) Cα RMSD (Å)	BC-loop Cα RMSD (Å)	DE-loop Cα RMSD (Å)	FG-loop Cα RMSD (Å)
6.2	7.85 ± 0.95	5.23 ± 0.57	16.08 ± 2.98	3.72 ± 1.13	7.41 ± 0.83	2.44 ± 0.40
7.2	8.08 ± 1.11	6.72 ± 1.57	13.39 ± 1.52	5.95 ± 2.25	9.71 ± 2.21	3.05 ± 1.07

Table 5. Cα RMSD of the full ΔN6 intermediate as well as of specific protein regions in relation to the native structure.

Chapter 4. Protein-protein docking

4.1. Protein-protein interactions

Proteins exert their biological functions through the establishment of interactions with other molecules, particularly with other proteins, forming protein-protein complexes that are essential in many cellular processes such as signal transduction, information storage and processing (129). Protein complexes form by the establishment of different types of intermolecular interactions such as hydrophobic, electrostatic and hydrogen bonds, which, together with shape complementarity, are the main drivers of protein-protein association (130-132) (Figure 18).

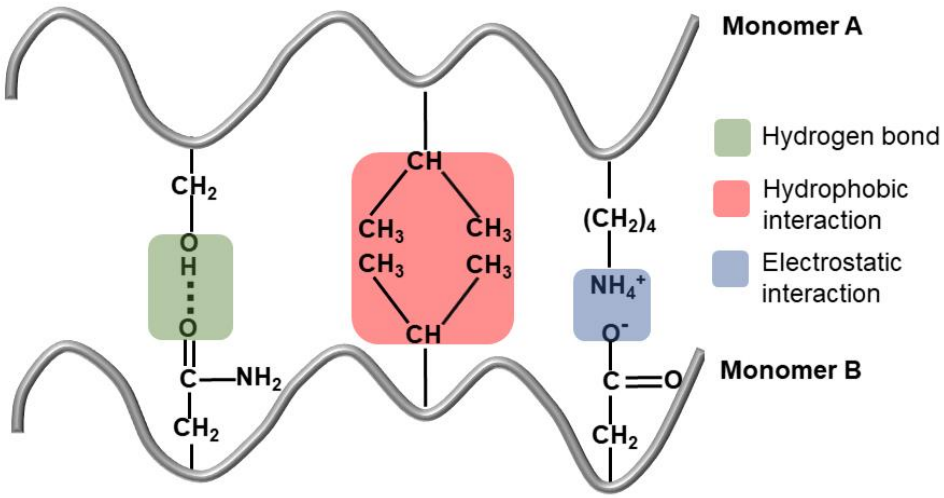


Figure 18. Representation of the main intermolecular interactions involved in protein association.

Hydrophobic interactions (i.e. interactions established between hydrophobic atoms) are fundamental for stabilizing protein-protein complexes and translate the free energy loss that occurs upon association of two atoms (one from each monomer) in a aqueous environment (131). In the process of protein-protein association the interaction between two hydrophobic atoms reduces the number of thermodynamically unfavourable solute-solvent interactions as a result of the hydrophobic effect (131). The maximum distance between the interacting atoms in an energetically stable hydrophobic interaction is ~5Å (133, 134).

Nevertheless, establishing exactly how much the hydrophobic interactions contribute to the stabilization of protein-protein complexes is not straightforward and several atomic solvation parameters (ASP) have been proposed to model and quantify the gain/loss of free energy upon desolvation of each atom type (135-137) based on experiments that transfer amino acids either from octanol to water (138), cyclohexane to water (139), or from vapour to

water (140). The ASP's are derived from the experimental free energies of transfer using the values of the atomic surface area for each amino acid type analogue used in the experiment. The ASP's derived free energies associated with vapour to water transfer lack accuracy as they are based on a collection of data determined from different laboratories (135, 141). This limitation is also shared by the cyclohexane to water transfer energy data. Additionally, the ASP derived from vapour to water transfer data tend to have negative or small positive values for carbon (a negative ASP value means that the exposure of this atom type to the solvent is favourable) (135, 137), which challenges the utility of this data given that the burial of carbon atoms is generally considered to be energetically favourable (135, 142). The ASPs derived from octanol to water transfer energy data appear to be the most accurate ones for treating protein desolvation and the hydrophobic interactions. Indeed, Cummings *et al.* have shown that the ASPs derived from the octanol to water transfer data are the best to discriminate between correct and incorrect solutions in the scoring of protein-protein docking results (135).

Another fundamental type of interaction in protein-protein association is the electrostatic interaction between charged atoms, which plays a major role in determining the strength and specificity of the interactions between proteins (131, 143). Electrostatic interactions in proteins are difficult to understand and quantify for several reasons. In particular, their long-range nature, the cooperativity of the acid-base equilibrium, the difference in the dielectric properties of proteins and the surrounding solution, and specific effects induced by ions in the vicinity of proteins (143). Indeed, some of the electrostatic interactions in proteins are between charges separated by distances comparable to the dimensions of the protein (144) and, thus, the electrostatic free energy of a protein system is the result of a large number of interactions. Additionally, the cooperativity of acid-base equilibrium implies that the individual pKa value of each ionisable group in a protein may depend considerably on their interactions with other charged and polar groups in the protein, in an interacting protein, or in the solvent (143, 145, 146). This interdependence of charges in a protein system deeply challenges a quantitative description of the electrostatic free energy of a protein or protein-protein complex (143). As the free energy of interaction between charges depends on the local dielectric properties of the medium(s) surrounding the charges (143), the heterogeneity of the dielectric properties between the protein interior, protein surface and solvent and, sometimes, within the protein interior itself (147) has a strong impact on its accurate evaluation. Another aspect that has impact on the strength of electrostatic interactions is the presence of ions in the medium surrounding the proteins. Since these

effects are not completely understood they are not included in most descriptions of protein systems (143). Therefore, the complex nature of electrostatic interactions makes them particularly difficult to study experimentally, which prompted the development of computational approaches to predict their role in protein-protein complexes (143). The most popular method is the numerical solution of the Poisson-Boltzmann equation that captures the dependence of the electrostatic potential on the density of charge embedded in a non-uniform dielectric continuum (143, 148, 149). However, solving the Poisson-Boltzmann equation is computationally expensive, and protein-protein docking algorithms often capture electrostatics interactions with simpler models such as the Coulomb's law with a distance-dependent dielectric (150).

The interactions between charged groups in proteins can be classified as local or long-ranged considering the distance between them (143, 151). Indeed, if a water molecule does not fit in the space between two charges (≤4Å), the interaction is defined as local. Similarly, if a water molecule can fit in this space, the interaction is considered long-range (143). Local interactions are considerably more energetically stable than long-range interactions. Indeed, these interactions involve a higher gain in the electrostatic free energy and also in the free energy of charge desolvation in comparison with long-range electrostatic interactions (143). Thus, stabilizing electrostatic interactions in proteins or protein-protein complexes have at maximum a ~7Å distance (151) between interacting charges, because of the high dielectric constant of water (around 80 at 25ºC) and the screening of the electrostatic potential by the dissolved ions between the two charges (143). The experimentally estimated free energy gain upon the formation of a salt bridge on the protein surface ranges from -0.7 to -1.7 kcal/mol (152-154).

The other fundamental interaction in protein-protein association is the hydrogen bond, which confers specificity to protein-protein interactions (131, 155). It is the strongest type of interaction, representing a free-energy gain of 1-5 kcal/mol (152, 156, 157). A hydrogen bond is an interaction in which a positively charged hydrogen atom, covalently bound to an electronegative atom (e.g. N, O, S), interacts with the lone pair of electrons of an acceptor electronegative atom (143, 155, 158). Therefore, the donor-H covalent bond should preferentially point along the axis of the lone electron pair of the acceptor, which results into the hydrogen bond having a directional character (143, 155, 158). The maximum distance between the electronegative donor and acceptor atoms in a hydrogen bond is around 3.2 Å (158-160).

Hydrogen bonds in proteins can be established between an NH group (donor group) and a C=O group (acceptor group) in the protein main-chain (158, 159). Alternatively, all the polar side-chains can form hydrogen bonds, inclusively with the main-chain groups, either acting as donors, acceptors or both (158, 159). Indeed, the hydroxyl groups of Ser, Thr and Tyr can be either donors or acceptors in hydrogen bonds as well as the imidazole nitrogen atoms of His (158, 159). The side-chains of Asn and Gln can act as hydrogen bond donors through their NH_2 groups, and as hydrogen bond acceptors through their C=O groups (158, 159). The side-chains of Asp and Glu can participate in hydrogen bonds as acceptors through the two carboxylate oxygen atoms of their carboxyl groups, as well as the Cys and Met side-chains through their sulphur atoms (158, 159). Arg side-chain can establish hydrogen bonds through their NH_2 and NH groups that act as donors while the side-chain of Lys mediates hydrogen bonds by its NH_3 donor group (158, 159). The Trp side-chain can only establish one hydrogen bond through its NH group that acts as a donor (158, 159) (Table 6).

Donor/Acceptor classification	Chemical group/Location
Donor	NH - backbone
Donor	NH – Trp, Arg and His side-chains
Donor	NH_2 – Asn, Gln and Arg side-chains
Donor	NH_3 – Lys side-chain
Donor	OH – Ser, Thr and Tyr side-chains
Acceptor	Imidazole N - His side-chain
Acceptor	C=O - backbone
Acceptor	C=O – Asn and Gln side-chains
Acceptor	OH - Ser, Thr and Tyr side-chains
Acceptor	COO – Asp and Glu side-chains
Acceptor	S – Cys and Met side-chains

Table 6. Donor and acceptor chemical groups for hydrogen bonds in proteins.

Hydrogen bonds, as well as electrostatic interactions in general, tend to have a more prominent role in protein binding than in protein folding, in contrast to what happens with hydrophobic interactions (161-163). These differences arise from the larger amount of polar and hydrophilic side-chains at protein surfaces in comparison with the protein interior.

Therefore, there is a higher proportion of main chain-side chain and side chain-side chain hydrogen bonds and a lower proportion of main chain-main chain hydrogen bonds at interfaces (161). However, the geometry of the hydrogen bond in protein interfaces is usually less optimal and has a wider distribution than those observed in the interior of proteins (161). Therefore, the hydrogen bonds laying on protein interfaces are weaker than those located in protein interior, that are typically involved in the formation of secondary structural elements (161).

4.2. Applications of protein-protein docking

In order to understand the role of each type of protein-protein interaction and the molecular mechanisms underlying the biological processes mediated by these interactions we need to know the three-dimensional structures of the resulting protein-protein complexes with atomic detail (129). However, there are relatively few high-resolution structures of protein-protein complexes (20700 out of 154015 Protein Data Bank (PDB) structures - http://www.rcsb.org/ (164)) generated by experimental methods such as x-ray crystallography or NMR spectroscopy (165-167). This has stimulated the development of computational methods to model the structure of identified protein-protein complexes in order to obtain their atomistic structures and understand their biological functions. Computational docking methods are methods that predict the structure of a protein complex by taking as input the atomic coordinates of the unbound monomeric components (168-170). They are increasingly being used to fill the gap of structural information on protein-protein complexes by taking advantage of the increasing number of experimentally solved monomeric protein structures (165, 171). Therefore, protein-protein docking is an invaluable tool to clarify and predict the biological function of many protein interactions, and to elucidate the molecular mechanisms behind many essential cellular processes, thus being an essential tool in systems biology.

A type of biological problem in which protein-protein docking methods have a relevant role is in providing insights into the aggregation mechanism of amyloidogenic proteins such as β2-m. Understanding the aggregation mechanism of amyloidogenic proteins is essential for the development of effective therapies targeted at amyloid disorders, which result from protein aggregation and deposition of amyloid fibrils. However, the highly transient and heterogeneous nature (172) of the initial oligomeric states formed along the aggregation pathway precludes the use of standard biophysical methods such as NMR or x-ray crystallography for structural characterization. Therefore, protein-protein docking methods are increasingly being used to overcome the limitations faced by current

experimental methods and to provide testable predictions regarding the aggregation mechanisms (57, 65, 103, 105, 115).

4.3. Stages of general docking procedures

Despite the large variety of protein-protein docking methods currently available, which are based on different search methods and scoring functions, most of them comprise four common stages that we will describe in detail the following sections: (1) a rigid body search, (2) selection of the region(s) of interest of the conformational space, (3) refinement of the docked structures, and (4) scoring and selection of the best models (165, 169, 173) (Figure 19).

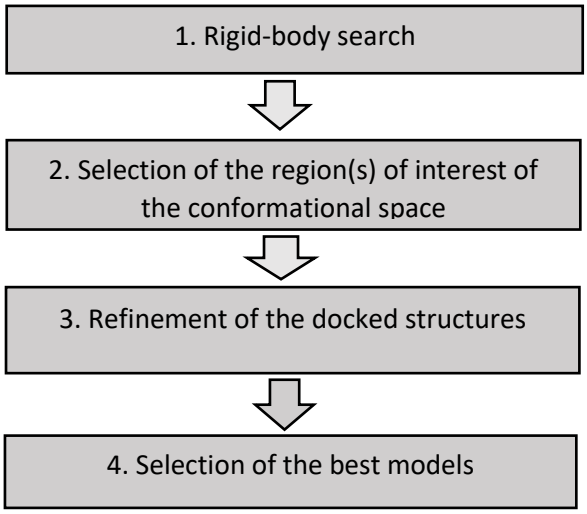


Figure 19. Representation of the phases of the general docking procedure adopted by most protein-protein docking algorithms.

4.3.1. Rigid body search

Most docking methods begin with a rigid body search based on rotational and translational moves that can be done over the entire conformational space of one of the monomers in relation to the other (global search), or be restricted to specific regions of the conformational space (local search) (169). The rigid body search generates a set of thousands of models in which the two proteins are in contact and satisfy minimal criteria of interaction, particularly at the level of the geometric complementarity and also, in some algorithms, at the level of hydrophobic and electrostatic complementarity (165, 169, 174-179). While methods such as those based in Fast Fourier transforms (FFT) (180-182) or geometric hashing (183-185) perform a global search in this first stage, there are other methods such as the Monte Carlo minimization methods ICM-DISCO (186) and RosettaDock (187) that only explore the rotational/translational space around a known or hypothetical binding site, which greatly

simplifies the selection of the region of interest as well as the final model selection. Similarly, the restraint-driven method HADDOCK (High Ambiguity Driven Biomolecular DOCKING) (188) uses biochemical and/or biophysical information that supports the presence of certain residues in or near the interface (defined as active or passive residues, respectively) as interaction restraints to guide the exploration to the regions of the conformational space that satisfy these restraints.

4.3.2. Selection of the region(s) of interest

Although the rigid body search generates a large set of models, including some nativelike conformations (i.e. conformations close to the native structure of the complex, which is the global energy minimum of the complex correspondent to its actual structure) these models lack atomic accuracy. Indeed, the rigid body search is based on "soft" scoring functions that allow the occurrence of atomic overlaps in the interfaces (165, 169). Thus, these models should be structurally refined by taking into account protein flexibility. However, since the generated models typically include many false positives, and that the refinement phase is computationally expensive (165, 169), the number of structures to be refined must be reduced. Therefore, a selection of the native-like models needs to be performed. This selection can be done using the method's inherent scoring functions (including geometric, physical and chemical properties of the interfaces), knowledge-based atom-atom (or residue-residue) potentials, and information on residue evolutionary conservation (165). Another possibility for selecting the best models for refinement is to use clustering algorithms that isolate the lowest energy structures (169, 189). Lorenzen and Zhang (190) showed that this approach gives better results than using the inherent scoring functions of four FFT-based protein-protein docking methods. Typically, this stage reduces the number of models generated by the rigid body search from thousands to a few dozens (169).

4.3.3. Refinement of the docked structures

In the refinement stage, the structural accuracy of the models selected in the previous stage is enhanced by removing atomic clashes, optimizing side chain conformations and improving the complementarity of the non-bonded interactions, particularly electrostatic and hydrophobic interactions (165, 169, 191). This can be achieved by simple EM as in RDOCK (191) and ATTRACT (192) and by Monte Carlo simulated annealing and minimization coupled with side-chain rotations and backbone movements as implemented in RosettaDock (187, 193). The RosettaDock force field includes side-chain rotamer preferences and the possibility to change predefined main-chain dihedral angles (165, 187, 193). Currently,

RosettaDock also uses variable loop modelling (194) and ensemble docking (195) to further account for backbone conformational changes. Structure refinement can also be performed by MD in explicit solvent as in HADDOCK (188). These procedures account for protein flexibility upon binding and improve the structural resolution and accuracy of the refined models, which is necessary to obtain near-native structures.

4.3.4. Scoring and selection of the best models

The last stage of the general docking procedure selects the candidate models that most likely correspond to the native structure. Although at this stage the scoring functions does not have to allow for the inaccuracies that occur during the soft rigid body docking phase, this selection process is not trivial (169). Indeed, sometimes is not clear how the models submitted to the community experiment on protein-protein docking methods CAPRI (Critical Assessment of Predicted Interactions) are chosen (169, 196). As an example, while the lowest energy structures are selected in the ICM-DISCO method (186), in the RosettaDock (187, 193) and HADDOCK (188) methods clusters of structures located on low energy basins are selected. In this respect, it is important to have in mind that the energy scoring functions are not globally discriminatory in any of these methods (169), which could explain the difficulty in selecting the native structure. While this problem is minimized in HADDOCK (because the search is restricted by experimental information), it is an important limitation to be addressed in the other methods.

4.4. An overview of protein-protein docking algorithms

The different protein-protein docking methods available can be classified according to the level of *a priori* information they require into global methods, medium range methods and data-driven methods (169) (Table 7).

4.4.1. Global methods

The global methods based on FFT or geometric hashing perform a rigid-body global search over the entire conformational space of one of the monomers in relation to the other (169).

The FFT methods are based on a representation of proteins in which the atoms are represented by cubes. In this approach, the position of one of the monomers, say, monomer A is fixed, while monomer B (initially centred at the origin of the Cartesian coordinate system) is allowed to make translational moves only. Two weights $a(x)$ and b(x) are assigned to each point x of the cubic grid (165, 173, 180). These weights are 0 outside the region occupied by

the proteins and 1 on the grid's surface. Furthermore, *a(x)* is 1 inside protein A, and *b(x)* is a large negative value inside protein B. The scoring function is calculated as the sum of the product of the two weights at all grid points. Thus, there are favourable positive terms in the points of the grid where the two protein's surfaces overlap and large unfavourable negative terms in grid points where the two proteins interpenetrate (165, 173, 180). For each translation of protein B, the correlation between the two weights is calculated. The FFT is applied to calculate simultaneously the correlations of all translations of protein B, which makes the translation search very efficient although this process has to be repeated for each orientation of protein B (165, 173, 180). This search lasts a few hours on standard personal computers (165, 170).

Additionally to shape complementarity, several FFT-based docking methods have been incorporating terms on electrostatics, desolvation energy, hydrophobicity and knowledge-based pairwise potentials (165, 173-179, 181). One major drawback of this type of docking methods is that they produce many false positive solutions because, by getting rid of artificial steric clashes resulting from discretization, they accept real steric clashes (173). Therefore, these methods often include a refinement phase in which the resolution is increased up to the atomic level and the discrete grid search is replaced by a continuous search (e.g with EM or MD) (173). Some of the most known FFT-based docking methods are the FTDock (197), ZDOCK (174, 182), PIPER (181), MolFit (180) and GRAMM-X (198).

Some alternatives to grid FFT-based docking algorithms employ spherical harmonics to represent the surfaces of the interacting proteins and overcome the necessity of repeating the FFT procedure for each relative orientation of the interacting proteins (165, 173, 199). Examples of methods using this approach are Hex (200) and FRODOCK (201). These methods employ the Fast Fourier on the rotational space rather than in the translational space, which allow them to be faster than the grid FFT-based docking methods, taking just a few minutes on standard personal computers (170, 173, 199).

Another approach to protein-protein docking is the use of geometric hashing, in which protein surfaces are represented by a set of discrete points, termed critical points, representing holes and knobs on proteins surfaces. This representation was firstly proposed by Connolly (202, 203) and later adapted by the groups of Wolfson and Nussinov (184, 204) to improve its computational efficiency. These adaptations, besides using a set of critical points representing convex caps, toroidal belts and concave pits, add normal vectors at these points (165, 173,

204). The geometric hashing algorithm rapidly matches the caps and pits with opposing normal directions on the surfaces and generates in seconds or minutes rigid body solutions that are geometrically acceptable, being one of the fastest types of docking methods (165, 173, 184). This advantage makes these methods suitable for the prediction of small multi-protein complexes (199). The most well-known method in this category is PatchDock (183, 184, 205). However, other geometric hashing methods have been developed to solve more complex docking problems such as FlexDock (184, 206), which is capable of docking proteins with domain motions of large amplitude while preserving computational efficiency, and SymmDock (184, 207), which incorporates molecular symmetry to generate models of oligomeric proteins with up to twelve subunits. CombDock (208, 209) also predicts the structure of multi-protein complexes by using a graph-based algorithm to reduce the complexity of the combinatorial assembly.

Overall, although geometric hashing docking methods cannot build high resolution models, they can generate likely binding geometries in a relatively short computer time (173).

4.4.2. Medium-range methods

Global methods based on FFT or geometric hashing generate a large number of models. However, these are low-resolution models (including many false positive solutions) that need to be further refined to yield native-like models. It is thus necessary to use docking methods that only explore the regions of the conformational space that appear to be close to the native state (local search), hence called medium-range methods (169).

The medium-range docking methods include methods that perform a deterministic sampling by MD (210) or simple EM (192), and a stochastic sampling by Monte Carlo minimization (186, 193). There are also medium-range methods based on genetic algorithms (211, 212).

Typically, these methods represent the monomers with atomic resolution and use force fields where van der Waals interaction parameters and a partial charge are attributed to each atom (173). These algorithms compute interaction energies as the sum of pairwise van der Waals and electrostatics interactions, and frequently include desolvation terms to account for the free energy of protein association (173). In this approach, the regions of the conformational space that present minimum energy values correspond to favourable binding geometries (173). One advantage of many medium-range methods over global methods is that they frequently search the conformational space beyond six rigid-body degrees of freedom

(173), taking into account the movements of the side-chains and, sometimes, even of the backbone (187, 193), albeit at a larger computational cost.

Some of the most known medium-range methods based on deterministic sampling are the integrative method developed by Zhou and colleagues (210) and the multi-minimization method ATTRACT (192, 213). The ATTRACT algorithm carries out a series of EMs based on rotations and translations of one monomer in relation to the other, using a coarse-grained protein representation with up to three pseudo-atoms per residue (the C_α and up to two side-chain atoms) and a scoring function comprising an electrostatic Coulombic term and a smoothed van der Waals term (173, 192, 213). This simplified model speeds up calculations allowing the docking of many thousand starting structures in just a few hours. Furthermore, they generate much less energy minima than those generated by a full-atomistic protein representation (192, 214). Additionally, this method takes into account protein flexibility by using conformational copies of surface side-chains (representing possible rotameric states of the side-chains), and by using conformational copies of loop regions (192, 213, 214). Therefore, ATTRACT can generate native-like solutions in a short computer time (192, 213).

The most well-known medium-range methods using a Monte Carlo stochastic sampling are the ICM-DISCO (186) and the RosettaDock (193) algorithms. The ICM-DISCO algorithm uses a full-atomistic protein representation and comprises two main stages: (1) a pseudo-Brownian Monte Carlo rigid-body search in the six translational and rotational degrees of freedom starting from multiple positions of the mobile protein around the fixed one; and (2) a side-chain refinement of the residues at the protein interface (186). In the first stage, a possible new position or orientation of the mobile protein is randomly generated, after which the energy of the new binding pose is compared to the energy of the previous one and accepted or rejected according to the Metropolis criterion (173, 186). The scoring function of this procedure is thus based on an evaluation of the interaction energy of each binding geometry composed of several contributions: a van der Waals interaction term, an electrostatic interaction term corrected for the solvation effect, a hydrogen-bonding interaction term and a hydrophobicity term (173, 186, 214). Additionally, a solvation energy term based on atomic accessible surfaces was added to the original scoring function to re-evaluate the generated docking solutions, improving the method's predictions (186, 214). The conformations generated at the end of the rigid-body stage are clustered according to their RMSD and ranked according to their interaction energies (173, 186). Then, the representative conformations of the lower energy clusters are subjected to the second stage of the ICM-

DISCO algorithm, in which the side-chains of these structures are fully flexible (173, 186, 199). Indeed, in this step, the values of the torsion angles are sampled in addition to the six translational and rotational degrees of freedom, using also a Monte Carlo procedure (173, 186, 199). The ICM-DISCO docking algorithm is therefore an accurate docking method that handles well the induced conformational changes of the surface side-chains although it is less successful when the backbone undergoes large scale rearrangements (186, 214).

The RosettaDock docking method uses a Monte Carlo minimization algorithm on both rigid-body and side-chains degrees of freedom to find the lowest energy complex of two protein structures. This method comprises two main stages: (1) a rigid-body low resolution Monte Carlo search in the six translational and rotational degrees of freedom starting with random orientations and positions of each partner; and a (2) high resolution refinement to simultaneously optimize the rigid-body and side-chains degrees of freedom (187, 193). The first stage is a rigid-body Monte Carlo search in which one of the monomers translates or rotates around the surface of the other monomer in each one of the 500 Monte Carlo steps employed in the search (187). In this phase, the algorithm uses a low-resolution representation in which each amino acid is represented by the backbone atoms and by a centroid pseudo atom representing the average position of the side-chain. The average position of each side-chain is determined using a set of structures presented in the Protein Data Bank (PDB) (187, 193). The scoring function used in this first step is based on residue-scale statistical potentials derived from the PDB and in a score to reward contacting residues while applying a penalty for overlapping residues (steric clashes) (187). These scores are computed after each proposed move and subjected to the Metropolis criterion to decide the acceptance or rejection of the move (187).

In the second stage, initial side-chains are added to the backbones by using Monte Carlo simulated-annealing over a backbone-dependent rotamer library. Then, to simultaneously optimize the rigid-body and the side-chains degrees of freedom, a series of 50 cycles of Monte Carlo minimizations is applied. A full-atom scoring function comprising a van der Waals interaction term, an implicit solvation term, a hydrogen bonding term, a rotamer probability term, a residue-residue pairwise statistical potential term and a electrostatics term is used in this search (187, 193). Each cycle comprises an initial rigid-body move (translation or rotation) followed by a packing step (in which the positions of the side-chains are optimized), and an explicit minimization to find the nearest local energy minimum (187). At the end of each cycle, the new position is accepted or rejected according to the

Metropolis criterion (187). The whole process is repeated 10^4 times to ensure that the conformational space is thoroughly sampled and, therefore, that the large ensemble of models generated includes some native structures (193). The generated models are ranked according to their free-energies and then clustered according to their RMSD (187). The lowest-energy models from the largest clusters are selected as the final predictions (187).

The RosettaDock algorithm is, therefore, a very accurate docking method that can generate models very close to the native complexes, both in their rigid-body orientations and in their side-chains conformations (193). However, it fails when there are large backbone conformational changes upon binding (193). This limitation was taken into account in recent versions of the method that introduce modelling of variable loops (194, 215), ensemble docking using computational and NMR ensembles, and induced fit using energy-gradient-based backbone minimization (195).

4.4.3. Restraint- or data-driven docking

Some protein-protein docking methods explicitly use experimental biochemical and/or biophysical data that support the presence of certain residues in the binding interface in the search procedure of their algorithms. This information is encoded in the form of restraints that guide the search to the regions of the conformational space that satisfy these restraints. Thus, the conformational space explored by these algorithms is significantly reduced, increasing their computational efficiency.

The most well-known docking method of this category is HADDOCK (188), which is used in this project to evaluate the results obtained with our MC-ED method. HADDOCK uses biochemical and/or biophysical data on the interacting residues derived from NMR titration chemical shift perturbations or mutagenesis to create Ambiguous Interaction Restraints (AIR) (188, 216). The AIR are ambiguous distance restraints with a maximum value of 3 Å between any atom of a so-called “active” residue of one monomer and any atom of the “active” and “passive” residues of the other monomer (188, 216). The active residues are the ones having a high NMR chemical shift perturbation upon complex formation and a high solvent accessibility in the free monomer (>50% relative SASA) or, alternatively, residues that when mutated abrogate complex formation, and that are also solvent exposed (188, 216). The passive residues are the residues that show a less significant NMR chemical shift perturbation upon complex formation and/or are surface neighbours of the active residues and, simultaneously, have a high solvent accessibility (>50% relative SASA) (188, 216). The 3 Å distance limit is established as a compromise between hydrogen-hydrogen and

heavy atom-heavy atom minimum van der Waals distances (188). The use of AIR allows HADDOCK to perform a search through all the possible conformations around the interacting site defined by the biochemical and/or biophysical data and find the ones with the most favourable pairs of interacting residues among the active and passive residues (188).

The HADDOCK docking algorithm consists of three stages: (1) a randomization of orientations and a rigid body EM, (2) a semi-rigid simulated annealing in torsion angle space (TAD-SA), and (3) a final refinement in Cartesian space with explicit solvent (188, 216). In the initial randomization step, the two proteins are placed at 150 Å from each other and each one is randomly rotated around its centre of mass (188). Then, rigid-body EM is performed consisting of four cycles of orientation optimization in which each protein rotates to minimize the intermolecular energy followed by optimization in both translational and rotational space (188). In this stage, around 1000 complex structures are generated, of which the best 200 solutions at the level of intermolecular energies are selected for refinement (188). The second stage comprises three consecutive simulated annealing refinements (188). In the first simulated annealing, the two proteins are subjected to an orientation optimization in which they are modelled as rigid bodies. In the second simulated annealing, the side-chains at the interface are allowed to move (188). In the third and final simulated annealing, both the side-chains and the backbone at the interface are allowed to move to account for the conformational rearrangements that can occur upon binding (188). The amino acids at the interface that are allowed to move are the active and passive residues used in the AIR plus the two amino acids immediately before or after that ones (188). Subsequently, the obtained structures are subjected to 200 steps of steepest descent EM (188). The final stage consists in a refinement by classical MD in an 8 Å shell of TIP3P water molecules in order to improve the energetics of the interface, which is important for an adequate scoring of the resulting conformations (188).

The HADDOCK scoring function consists of a weighted sum of van der Waals, electrostatic, desolvation and restraint violation energies together with buried surface area (216). The final structures are clustered using the pairwise backbone RMSD at the interface, and the generated clusters are analysed and ranked according to their average interaction energies and their average buried surface areas (188).

HADDOCK is a robust docking program, one of the most accurate methods participating in the CAPRI experiment, and is capable of generating near-native solutions

even when large conformational changes take place upon binding (169, 217, 218). The only disadvantage of this method is that it requires the existence of reliable and accurate experimental information on the binding interface to be used as restraints (169).

Method	Type of search	Treatment of protein flexibility	Examples
Global methods based on Fast Fourier Transform or geometric hashing	Rigid-body global search	Only rotational and translational degrees of freedom; smoothing of the protein surfaces	FTDock (197) ZDOCK (182) PIPER (181) MolFit (180) PatchDock (183)
Medium-range methods based on EM, Monte Carlo minimization or MD	Local search over a limited region of the conformational space; can be deterministic (eg. EM, MD) or stochastic (eg. Monte Carlo minimization)	Allows side-chains flexibility and, in some cases, also backbone flexibility	ATTRACT (192) ICM-DISCO (186) RosettaDock (193)
Data-driven docking	Restrained by experimental data on interface residues	Allows extensive flexibility of both side-chains and backbones	HADDOCK (188)

Table 7. Classification of protein-protein docking methods according to the level of *a priori* information they use.

4.5. State of the art on protein-protein docking

Despite the recent extraordinary evolution of protein-protein docking methods and their increasing ability to produce models of protein complexes close to the native ones, the docking community still faces some challenges.

One important limitation of many docking methods, particularly of those solely based on shape complementarity, is the generation of false positive docking solutions. This problem arises mainly from the incapacity of the scoring functions in discriminating the near-native models from the non-native models. This limitation has been tackled by several groups who developed docking methods with more accurate and complete free-energy based scoring functions, such as the Monte Carlo based methods ICM-DISCO and RosettaDock, and the restraint driven method HADDOCK. There are also attempts to construct sufficiently robust scoring functions to enable the correct ranking of low resolution models in order to avoid the time-consuming phase of structure refinement (173).

The main challenge of almost all protein-protein docking methods is to correctly predict the structure of protein-protein complexes when there are large conformational

rearrangements upon binding (169). The only method capable of handling this problem is HADDOCK (169) due to the use of ensemble docking (217), but it requires reliable and accurate experimental information about the interaction, which is not always available. Another strategy that has been used (e.g. in HADDOCK, ICM-DISCO and RosettaDock) to tackle this problem is to model the flexibility of the protein complex throughout the docking process. This is usually achieved by introducing flexibility at the level of side-chains and/or backbones in the structure refinement phase. However, this strategy has the disadvantages of being computationally expensive and of generating many false positives. Therefore, the docking community still has to improve the existing docking methods or develop new ones to cope with this major limitation of the actual protein-protein docking algorithms.

In this respect, our *in-house* developed MC-ED (described in the next chapter) has a specific feature that makes it different from most available docking methods. Indeed, it retrieves not a single protein-protein complex from two monomers but, instead, it computes an ensemble of protein complexes (by using ensembles of monomers representative of some conformational state of interest), which is then analysed statistically. In particular, instead of a few native-like conformations, the method generates an ensemble of protein complexes optimized for shape, hydrophobic and electrostatic complementarity, and interfacial hydrogen bonds. The rationale stands from the fact that a given conformational state of a protein, either a folding intermediate or even the native state, comprises not a single conformation but instead many conformations as a result of the intrinsic structural dynamics of proteins. Therefore, the MC-ED algorithm implicitly models the protein conformational flexibility upon binding.

Another major challenge in the field of protein-protein docking, particularly when it comes to predicting higher-order oligomeric assemblies (e.g. trimes, tetramers, hexamers etc), is the high combinatorial complexity of these problems and the limitations of the available current computer resources. Some groups have been tackling this problem by introducing symmetry restraints to reduce the complexity of the search (219). This issue is likely to be met with better results in the near future as a result of developments in the computer power and of progress in the docking methods tackling multi-component docking. In this regard, the relatively simple scoring function of our MC-ED allowed us to adapt our algorithm to predict the structures of ensembles of tetramers in a reasonable computer time.

Chapter 5. Monte Carlo Ensemble Docking

5.1. Overview and rationale

The Monte Carlo Ensemble Docking (MC-ED) is a rigid-body docking procedure whose goal is to provide with atomic detail a statistically resolved picture of the dimerization interface by creating ensembles of dimers that are used to predict the regions of the protein more likely involved in protein-protein interfaces and, at a finer level, which residues are critical to initiate protein self-association (i.e. the aggregation hot-spots) (57, 116, 220).

To construct an ensemble of dimers the MC-ED is consecutively applied to randomly selected pairs of monomers (representative of a conformational state of interest, e.g. an aggregation-prone intermediate state) until the mean and standard deviation of the optimized property (e.g. number of intermolecular contacts or intermolecular binding energy) both converge. This typically requires the docking of approximately 1000 pairs of conformations per studied species and environmental condition.

The resulting ensembles are used to construct probability distribution (density) functions of the intermolecular contacts (intermolecular binding energy), respectively. This statistical analysis allows gauging the quality of the dimerization interface by providing a quantitative description of interface properties (e.g. intermolecular contacts or intermolecular binding energy) and, therefore, to get first glimpses into the relative aggregation potential of the considered monomeric species.

The generated dimer ensembles are also used to construct intermolecular probability maps (e.g. representations of the probability of each intermolecular contact in the ensemble of dimers) to identify the most likely structural regions involved in the interfaces, as well as to determine the aggregation hot-spots (e.g. the residues involved in more intermolecular contacts upon binding, computed from the subset of the 50 most frequent intermolecular contacts). We choose the subset of the 50 most frequent intermolecular contacts as this subset captures most of the probability of intermolecular contact formation in the dimers (i.e the probability decreases markedly from the 50 most frequent intermolecular contacts onwards).

The idea of generating an ensemble of complexes as the outcome of a docking procedure is motivated by the fact that the conformational states formed along the aggregation pathway are significantly structurally heterogeneous. Indeed, structural heterogeneity is one of the hallmarks of protein aggregation. On the other hand, by considering ensembles of monomers representative of the same monomeric state the method is indirectly taking into

account the structural variability that results from the dynamic nature of proteins, and therefore it somehow mitigates a limitation arising from its rigid-body nature. Indeed, in the MC-ED the monomers are considered rigid bodies (i.e. the distances between the atoms do not change) and therefore side-chain and backbone flexibility (which are necessary to capture conformational changes that may occur upon binding) are not taken into account contrary to what happens in other docking methods such as ICM-DISCO (186), RosettaDock (193) and HADDOCK (188). We stress, however, that since the main goal of the MC-ED method is not accurate structure prediction but instead the identification of the regions (and residues) of the monomers that are more likely to trigger the aggregation cascade, the lack of protein flexibility does not represent a severe caveat.

5.1.1. Original version of the Monte Carlo Ensemble Docking

The original version of the Monte Carlo Ensemble Docking developed by Faísca *et al.* and described elsewhere (57, 116, 220) generates ensembles of random dimers optimized for shape complementarity, a major driver of protein-protein association (130, 221).

Operationally the method comprises two stages. The first stage is the choice of the so-called docking axis: the centre of mass (CM) of one of the monomers is fixed at the origin of a Cartesian coordinate system and the CM of the other monomer is placed at a distance (which is equal to the sum of the two monomer's gyration radius) away from the origin along the direction of each of the main Cartesian axes, either to the left or to the right. From the six possible initial poses, the one that maximizes the total number of intermolecular contacts is selected as the initial state of the docking procedure, and the corresponding Cartesian axis is termed docking axis.

Subsequently, in the second stage of the method, the Monte Carlo (MC) optimization stage, the monomer whose CM is located at the origin is kept fixed while the other is allowed to move. In particular, at each MC step the moving monomer is subjected to a translational or rotational move with equal probability 1/2.

In the translational move, the CM of the moving monomer translates along the docking axis from the initial distance of l_{old} between the CMs of the two monomers to a new distance defined by:

$$l_{\text{new}} = l_{\text{old}} + l_{\text{int}}(\zeta[0,1[-0.5) \quad (1)$$

where $\zeta[0, 1[$ is a uniformly distributed random number between zero and one and l_{int} is the perturbation amplitude of the translational move. This amplitude has to be optimized in order to achieve convergence of the MC simulation (i.e. having a probability of acceptance of each MC move of approximately 0.5 (0.45-0.55)). Its initial value has been adjusted to 0.05 Å in order to achieve convergence.

In the rotational move, the moving monomer performs a random rotation around a random axis that passes through its CM according to the right-hand rule. This is achieved by introducing a rotation matrix $R(\mathbf{u}, \boldsymbol{\alpha})$ acting on all atoms of the moving monomer with position vector $\mathbf{r}$ through:

$$\mathbf{r}' = R(\mathbf{u},\boldsymbol{\alpha})\boldsymbol{r} \quad (2)$$

where α is a random angle in the interval $[0,2\pi l_{\text{int}}[$, and l_{int} is the perturbation amplitude of the rotational move and has the initial value $l_{\text{int}}= 0.05$ rad, $\mathbf{u} = (u_{\text{x}}, u_y, u_z)$ is the unit vector that defines the rotation axis, whose components are $u_{\text{x}}= \sin\boldsymbol{\theta}\cos\boldsymbol{\varphi}$, $u_y = \sin\boldsymbol{\theta}\sin\boldsymbol{\varphi}$ and $u_{\text{z}}= \cos\boldsymbol{\varphi}$, $\boldsymbol{\theta}$ is a random angle in the interval $[0, \pi[$, and $\boldsymbol{\varphi}$ is a random angle in the interval $[0,2\pi[$.

Based on Rodrigues' rotation formula, the rotation matrix is defined as:

$$R(\mathbf{u}, \alpha) = I\cos\alpha + \sin\alpha[\mathbf{u}]_{\times} + (1\text{-}\cos\alpha)\mathbf{u}\otimes\mathbf{u} \quad (3)$$

where $\otimes$ designates the tensor product, I is the identity matrix and $[\mathbf{u}]_{\times}$ is the anti-symmetric cross product matrix defined by:

$$[\mathbf{u}]_{\times} = \begin{bmatrix} 0 & -u_{\text{z}} & u_{\text{y}} \\ u_{\text{z}} & 0 & -u_{\text{x}} \\ -u_{\text{y}} & u_{\text{x}} & 0 \end{bmatrix} \quad (4)$$

At each MC step, intermolecular collisions are generated whenever two atoms are at a distance smaller than the hard-core distance (i.e. sum of the van der Waals radii of the two atoms (222)) and intermolecular contacts are generated whenever two atoms are within interaction distance (i.e. 1.25x the sum of the van der Waals radii of the two interacting atoms, the distance cut-off used in the folding DMD simulations with the Gō model). The Monte Carlo optimization seeks to maximize the number of intermolecular contacts while minimizing the number of intermolecular clashes (collisions or excluded volume interactions). To achieve this it uses two cost functions, one for the number of intermolecular contacts,

$$N(\text{contacts}) = \sum_{i=1}^{A} \sum_{j=i+1}^{A} \Theta\left(\frac{\sigma(\lambda - 1)}{2} - \left| l_{ij} - \frac{\sigma(\lambda + 1)}{2} \right| \right) \quad (5)$$

and another for the number of collisions,

$$N(\text{collisions}) = \sum_{i=1}^{A} \sum_{j=i+1}^{A} \Theta(\sigma - l_{ij}) \quad (6)$$

where σ is hard-core distance, λ is the scaling factor that controls the range of the intermolecular contacts, A is the total number of atoms in the protein, l_{ij} is the geometrical distance between atom i in the first monomer and atom j in the second monomer, and Θ is the Heaviside step function, which is 1 for positive arguments and 0 for negative arguments.

For the MC simulation to converge we deploy an entangled Metropolis criterion acting on the two cost functions. A single MC move can:

(a) Decrease the number of collisions and increase the number of contacts: the new conformation is accepted;

(b) Decrease the number of collisions and decrease the number of contacts: the new conformation is accepted with probability

$$P_1 = exp((N(contacts)_{new} - N(contacts)_{old})/T), \quad (7)$$

with T being the temperature at which the simulation is performed and N(contacts) the number of intermolecular contacts;

(c) Increase the number of collisions and increase the number of contacts: the new conformation is accepted with probability

$$P_2 = exp(-(N(collisions)_{new} - N(collisions)_{old})/T), \quad (8)$$

with N(collisions) being the number of intermolecular collisions;

(d) Increase the number of collisions and decrease the number of contacts: the new conformation is accepted with probability $P_3 = P_1 P_2$.

The converged MC simulation should have a 50% probability of acceptance of each MC move, which is achieved by dynamically adjusting the perturbation amplitude of the MC move.

5.1.2. Novel version of the Monte Carlo Ensemble Docking

The major modification introduced in the original version of the method described above was to replace the cost function that creates dimers optimized for shape complementarity by a novel cost function that optimizes the intermolecular (or binding) energy by taking into account electrostatic interactions, interactions between hydrophobic atoms (which we term hydrophobic interactions), as well as hydrogen bonds (130-132) (MC-ED dimerization code with new cost function at Supplementary Information). We follow Urbanc *et al.* (151, 223) and use square-well potentials to model inter-atomic interactions (Figure 20). In what follows we describe the three types of considered intermolecular interactions.

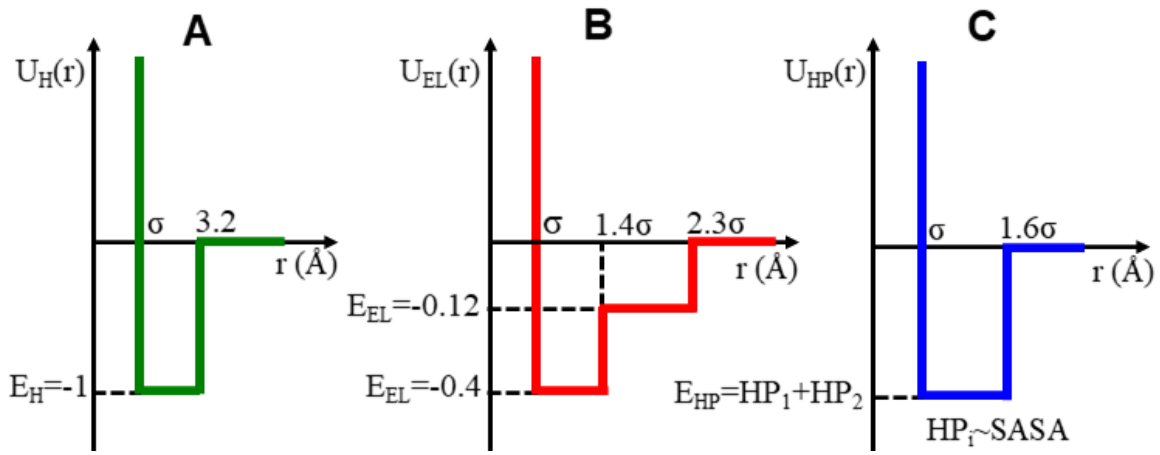


Figure 20. Square-well step potentials employed in the modified version of the MC-ED algorithm to model hydrogen bonds (A) and electrostatic (B) and hydrophobic (C) interactions. σ is the hard-core distance and r is the inter-atomic distance.

5.1.2.1. Electrostatic interactions

Following Refs. (151, 152), electrostatic interactions between two charged atoms can be approximated by a double square-well potential. Two atoms with charges of the same sign interact through a positive (i.e. repulsive) two-step potential while the interaction between two atoms with opposite charge is modeled through a negative (i.e. attractive) two-step potential $U_{EL}(r)$ (Figure 20B). For these pairwise interactions the signs of the atomic charges in the

GROMOS 54A7 force field were used (224). The protonation states of the protein's titrable groups at each pH were assigned accordingly to their pKa values, obtained from the CpHMD simulations. The cut-off distance is set to 2.3 σ, and the width of first potential well is σ < r < 1.4 σ (short-range interactions 4-5 Å) (143). The interaction strength corresponding to the first potential well, E_{EL1} E_{EL1}, is obtained by normalizing the median free energy gain upon salt bridge formation at the protein surface (1.2 kcal/mol) (152-154) to the interaction strength of an hydrogen bond (i.e. 3.0 kcal/mol in our model). As in (151), we set the interaction strength of the second potential to $E_{\mathrm{EL2}} = 0.3 \times E_{\mathrm{EL1}}$.

5.1.2.2. Hydrogen bond

Hydrogen bonds (h-bond) occur when a donor (D) atom donates its covalently bonded hydrogen atom to an electronegative acceptor (A) atom, D—H···A (D, A= N, O, S). In protein-protein association, an h-bond can be established between the backbones of two interacting chains (i.e. with the donor atom located at one of the backbones and the acceptor atom located at the other), or between pairs of (acceptor-donor) atoms pertaining to the side-chains located at the interfaces of the resulting protein complex. In this study, the distance between donor and acceptor atom was the only criteria to determine the occurrence of intermolecular h-bonds. The reason is threefold: 1) there is no consensus regarding the geometric constraints that should be imposed when modeling hydrogen bonds (155, 158, 225); 2) the linearity of the hydrogen bond is not a stringent requirement for its establishment since hydrogen bonds can be found in other geometric arrangements (which are, nevertheless, weaker) (161); and 3) to keep the method as computationally efficient as possible. In order to model h-bond interactions, we considered a square-well potential energy function $U_{\mathrm{H}}(r)$ (Figure 20A), whose width ranges between the hard-core distance σ (which is the sum of the van der Waals radii (222) of the two interacting atoms) and a cut-off of 3.2 Å, which is considered the maximum distance for the establishment of a moderately stable (1-5 kcal/mol) hydrogen bond (152, 156, 157, 160). Since hydrogen bonds in protein-protein interfaces are usually geometrically less optimal (and therefore less stable) than those in the protein interior (161), we took a conservative choice for the corresponding interaction strength $E_{\mathrm{HB}} = 3$ kcal/mol. This approach also avoids overestimating the contribution of non-geometrically optimal h-bonds for dimer stability. As in Urbanc *et al.* (152), we set the potential energy for the h-bond to unit energy $E_{\mathrm{HB}} = -1$.

5.1.2.3. Hydrophobic interactions

Regarding amino acid interactions resulting from their hydropathic nature, we consider a third square-well potential $U_{\mathrm{HP}}(r)$ that captures interaction between pairs of hydrophobic/hydrophilic atoms within the side-chains (Figure 20C). The interaction energy is negative (positive) when the distance between two hydrophobic (hydrophilic) atoms is smaller than 160% of the sum of their van der Waals radii (133), which is the cut-off distance below which the interfacial volume is considered as solvent excluded.

Atomic solvation parameters s_t (cal Å^{-2} mol^{-1})	s_C	$s_{N/O}$	s_{N^+}	s_{O^-}	s_S
	18±2	-7±3	-34±4	-20±8	18±6

Table 8. Atomic solvation parameters derived by Cummings *et al.* (135).

The potential energy between two interacting atoms is defined as $E_{HP} = HP_1 + HP_2$, with $HP = -s_t \times SASA_t \div n_t$ being the hydropathy value of a specific type of atom *t*. In this equation, S_t is the atomic solvation parameter of atom *t* (which corresponds to the free energy gain/loss per unit of solvent exposed area), $SASA_t$ is the solvent accessible surface area of atom *t*, and n_t is an estimate for the number of neighboring atoms (usually two). We use the values of the atomic solvation parameters derived by Cummings et al. (135) (Table 8) based on the transfer free energies obtained by Fauchère *et al.* (138), using the solvent accessible surface areas reported by Lesser *et al.* (226) (Supplementary Table 1). These parameters have a higher discriminating power in the evaluation of protein-protein docking solutions (135).

The hydropathy value of each interacting atom is normalized to the energy of a hydrogen bond, which results in hydrophatic parameters within the interval $-0.1 \leq HP_t \leq 0.4$.

The total energy of the dimer's interface contains the three contributions discussed above, i.e. the new cost function is given by:

$$U = \sum_{i,j} U_{ij}^{H} + U_{ij}^{EL} + U_{ij}^{HP} \text{ with } U_{ij}^{H} = U_H(r_{ij}), U_{ij}^{EL} = U_{EL}(r_{ij}) \text{ and } U_{ij}^{HP} = U_{HP}(r_{ij}) \quad (9)$$

In the modified version of the MC-ED method a single MC move can:

(a) Decrease the number of collisions and decrease the binding energy: the new conformation is accepted;

(b) Decrease the number of collisions and increase the contact energy: the new conformation is accepted with a probability

$$P_1 = exp(-(U_{new} - U_{old})/T); \quad (10)$$

(c) Increase the number of collisions and decrease the contact energy: the new conformation is accepted with a probability

$$P_2 = exp(-(N(collisions)_{new} - N(collisions)_{old})/T); \quad (11)$$

(d) Increase the number of collisions and increase the contact energy: the new conformation is accepted with a probability $P_3 = P_1 P_2$.

We also introduced other modifications into the new version of the MC-ED method. First, we improved the conformational sampling in the first stage (selection of the docking axis). As before, the CM of the moving monomer translates along the direction of one of the three Cartesian axis (to the left and to the right of the origin) but now it is also allowed to perform 90º rotations around each axis in both directions, counter- and clockwise. Additionally, to get initial dimer conformations with a lower number of steric clashes and a higher number of intermolecular contacts (i.e. with a better packed interface), we introduced a second modification in the first stage of the method: if the dimer conformations generated for each axis and/or orientation have more than 200 steric clashes (or less than 3000 contacts) we increase (or decrease) the distance between their CMs by multiplying it by 1.2 or 0.9, respectively, values obtained by a trial and error fine tuning. Accordingly, the monomers are approached if they are loosely packed or are moved away from each other if they have too many clashes. Other minor changes consisted in the adjustment of the amplitude of the translational move from 0.05 Å to 0.025 Å and the adjustment of the amplitude of the rotations from $l_{\text{int}} = 0.05$ rad to $l_{\text{int}} = 0.025$ rad to ensure the stability of the simulations.

5.1.3. Structure refinement with classical MD

In order to gauge the quality of the structures produced with the MC-ED, ensure compatibility of the dimer structures with the GROMOS 54A7 force field, and refine the structure of the dimers produced with the MC-ED protocol, the MC-ED generated dimers were subjected to a classical protocol of structure refinement that removes steric clashes and adjusts the side-chain and backbone conformations of the generated rigid-body poses.

This protocol comprised three steps of EM followed by three steps of initiation by classical MD in explicit water using the GROMOS 54A7 force field as implemented in the version 4.0.7 of the GROMACS software (227). The EM step is essential to correct all large steric clashes between atoms and avoid the crashing of MD simulations in the following steps. This is achieved by optimizing bond lengths and angles and the nonbonded interactions such that the dimers relax to a local energy minimum correspondent to a physically realistic structure, compatible with a MD protocol. However, if the initial structures have severe errors (e.g. excessive steric clashes), the EM procedure may not correct these structural issues and we end up with physically inaccurate and unrealistic structures that will not be stable in a MD simulation. The EM stage started with steepest descent algorithm (with a maximum number of 10 000 steps) without bond constraints followed by a 2000 step minimization using the conjugated gradient algorithm that also does not impose bond constraints. The final step of the EM was again performed with steepest descent method (maximum number of 10 000 steps) with constraints on all bonds imposed by the LINCS algorithm.

The minimization step corrects the majority of clashes, but the structures obtained are still of relatively high energy and need to be relaxed. The initiation step used classical MD simulations to relax the dimer structures without distorting too much their dimer interface, which was generated in the MC-ED protocol. For this, we introduced position restraints on the Cα and side-chain atoms that fix the positions of these atoms, hence avoiding dramatic alterations of their positions. The initiation step began with: (**1**) a 100 ps MD run at constant temperature with an integration step of 1 fs and position restraints of 1000 kJ mol^{-1} nm^{-1} on Cα atoms and of 100 kJ mol^{-1} nm^{-1} on side-chain atoms; (**2**) followed by a 200 ps MD run at constant temperature and pressure with an integration step of 2 fs with position restraints of 100 kJ mol^{-1} nm^{-1} on Cα atoms and of 10 kJ mol^{-1} nm^{-1} on side-chain atoms; (**3**) the last step of initiation is a 200 ps MD run at constant temperature and pressure with an integration step of 2 fs and position restraints of 100 kJ mol^{-1} nm^{-1} on Cα atoms and none on the side-chain atoms. The integrator algorithm used was the leap-frog algorithm and the nonbonded interactions were treated with a twin-range cutoff of 8/14 Å. The electrostatic long-range interactions beyond 14Å were treated using a generalized reaction field with a dielectric constant (i.e. relative permittivity) of 54. Constraints in all bonds were imposed by the LINCS algorithm. The temperature (310K) and pressure (1 bar) were treated using the Berendsen coupling with coupling constants of 0.01 and 2.0, respectively, and an isothermal compressibility of 4.5 x 10^{-5} bar. The dimers were solvated with SPC water molecules

ensuring that one monomer only interacts with the other in one direction (15000–35000 water molecules). The numerical instabilities in the MD simulations of this relaxation step were also used to discriminate and automatically discard the dimers with bad clashes that were not corrected by the minimization protocol.

About 75% of the dimer structures produced with the MC-ED docking protocol are successfully relaxed with the classical force field. The remaining ones have severe conformational clashes corresponding to very high-energy interactions that the minimization algorithms were not able to correct. The generation of some inaccurate dimer structures by the MC-ED was expected considering that rigid-body docking algorithms often produce those structures as a consequence of the lack of structural flexibility in the conformational search.

5.2. Testing the novel cost function

α-synuclein (PED ID: 9AAC) (228) is the etiological agent of Parkinson's disease (PD) and other neurodegenerative disorders (229). It is a highly aggregation-prone protein with high hydrophobic solvent exposure (Figure 21A and Figure 22A). As such, it is expected that hydrophobic interactions will play a more prominent role in α-synuclein dimerization than in the association of native state monomers of β2-m, which are compact and expose significantly less hydrophobic residues to the solvent (Figure 21B and Figure 22B).

To test the novel MC-ED method we carried out a comparative analysis of the ensemble of dimers resulting from the self-association of α-synuclein and the ensemble of dimers resulting from the self-association of native monomers of the D76N mutant of β2-m. For the sake of completeness we also considered the I_2 intermediate state (Figure 21C) of the D76N mutant, which, in terms of SASA per residue, represents a compromise between the other considered model systems (Figure 22B).

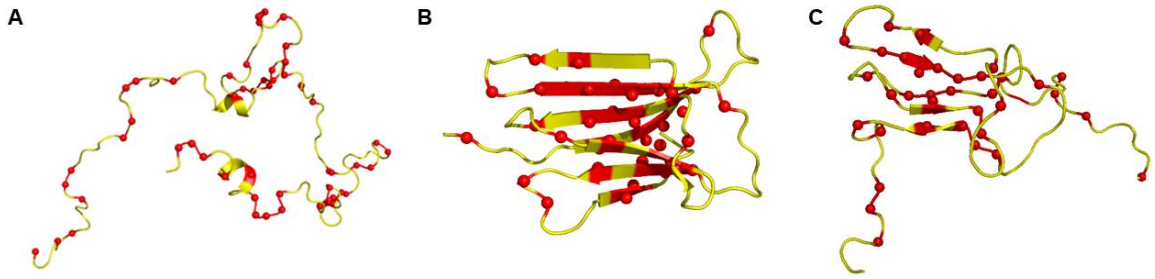


Figure 21. Three dimensional structure of alpha-synuclein (PED database ID: 9AAC) (A) and of the native state (B) and I_2 intermediate (C) of D76N β2-m (PDB ID: 4FXL) with the hydrophobic residues depicted in red.

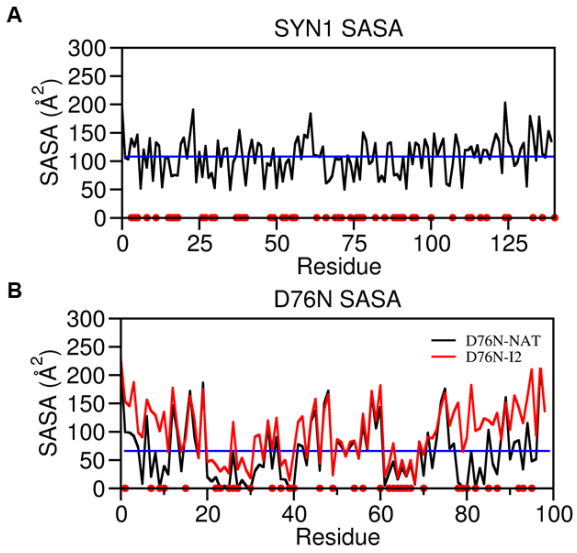


Figure 22. SASA of the residues in alpha-synuclein (upper graph) and in the native state and I_2 intermediate of D76N β2-m (lower graph) with the hydrophobic residues indicated by red points and the mean SASA per residue in the two model systems represented as a blue line.

We performed MC-ED simulations considering each energy contribution separately, i.e. we generated three ensembles of dimers per model system. The first ensemble of dimers was exclusively optimized for shape complementarity and electrostatic complementarity, the second ensemble of dimers was optimized for shape and hydrophobic complementarity, and, finally, in the third ensemble the binding energy contains contributions from packing, electrostatic, hydrophobic and h-bond interactions. The ensembles of dimers of α-synuclein were constructed by generating 1000 random pairs from the 576 NMR structures reported in the PDB file. By comparing the properties of the ensembles of dimers thus generated we should be able to confirm the hypothesis that alpha-synuclein dimerization is predominantly driven by hydrophobic interactions.

The analysis of the probability density curves (Figure 23) indicates that the packing and hydropathic interactions drive the formation of dimers with lower energy, i.e. that are typically more stable than those resulting from packing and electrostatic interactions, indicating that the cost function is correctly capturing the relative importance of hydrophobic interactions in α-synuclein dimerization.

To further explore the physical consistency of the cost function we compared the probability density curves of α-synuclein, I_2 intermediate and native state of D76N β2-m, i.e. conformational states that are progressively more compact and for which hydrophobic interactions should play a progressively minor role in self-association (Figure 24). We

observe that when dimerization is exclusively driven by packing and hydrophobic interactions, the mode of the energy distribution for α-synuclein corresponds to the lowest energy and that of the native state of D76N β2-m to the highest, with the dimers of I_2 showing intermediate stability (Figure 24A), in line with the degree of solvent exposure of the corresponding monomers. When dimerization is driven by packing and electrostatic interactions the three species form dimers with similar stabilities (Figure 24B). The analysis of the probability density curves obtained when all interactions contribute to stabilize the interface shows that the dimers of α-synuclein are the most stable (Figure 24C), and, given the exposed above, the major contribution to stability is likely the hydrophobic interactions.

In summary, the comparative analysis of the probability density curves indicates that the new cost function of the MC-ED algorithm is correctly capturing the fundamental driving forces of protein-protein association.

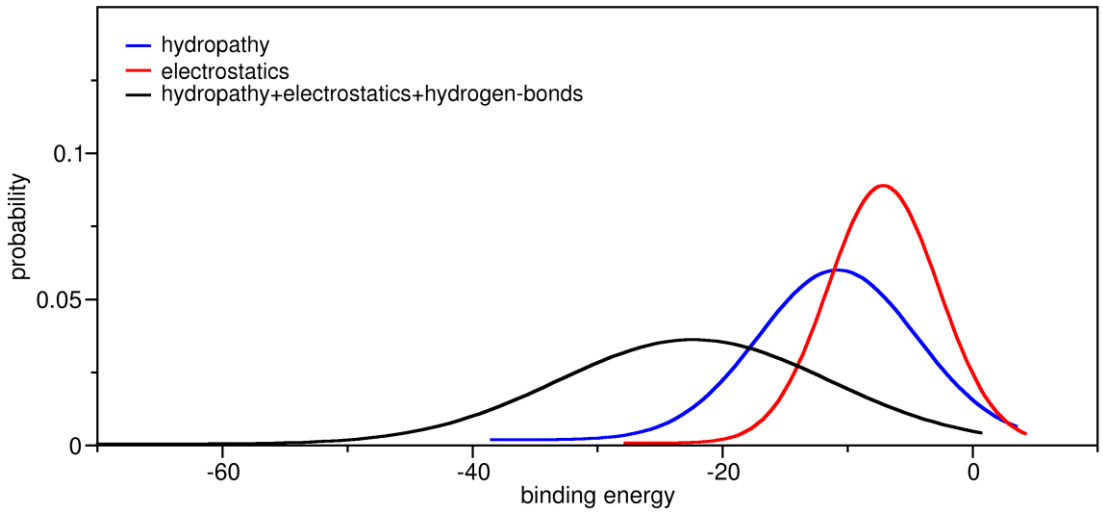


Figure 23. Probability density curves for the intermolecular energy of the alpha-synuclein dimers generated by MC-ED simulations with each individual cost function contribution. It is also represented the density curve for the intermolecular energy of the dimers generated by the correspondent MC-ED simulation with the complete cost function.

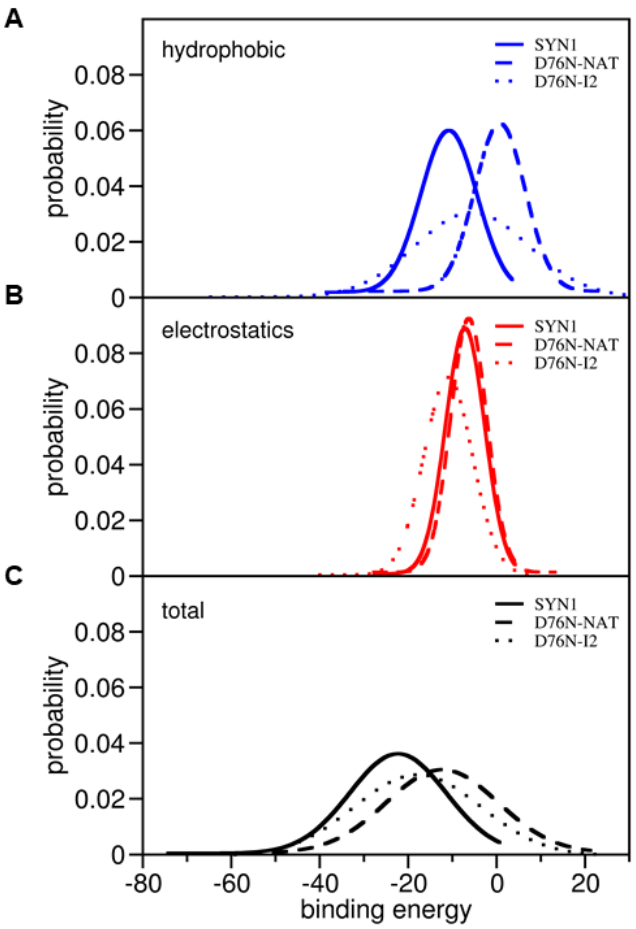


Figure 24. Density curves for the intermolecular energy of the ensembles of dimers of alpha-synuclein, and D76N native state and I_2 intermediate generated with each energy contribution of the new MC-ED cost function: (A) hydropathic, (B) electrostatics, and (C) complete cost function.

5.3. Selecting dimers with MM/PBSA

In future work, the interactions of β2-m dimers with cell membranes will be investigated in order to predict their cytotoxic effects (e.g. membrane disruption). These dimers must be stable enough in order to survive the timespan of classical MD simulations in which dimer-membrane interactions will be probed. We thus need to select MC-ED dimers based on some established criteria. One criterion is to select dimers that show the lowest binding energy in Molecular Mechanics/Poisson-Boltzmann surface area (MM/PBSA) calculations. Therefore we need to calculate the MM/PBSA energy of the MC-ED dimers whose structure was refined with classical MD.

5.3.1 The MM/PBSA method

The MM/PBSA method (230) is commonly used to calculate the binding free energy of protein-protein and protein-ligand complexes (231, 232). This method computes the

binding free energy of the complex as a sum of several energetic contributions, namely the polar solvation energy (or electrostatic solvation energy), the nonpolar solvation energy and the molecular mechanical (MM) electrostatic (Coulombic) and van der Waals energies. Additionally, an entropic term is often added (231-233) to ensure that the calculated binding energies are actually the free energies of binding. However, in our approach we did not include any entropy term in the calculation of the binding energy as this would require the generation of a conformational ensemble for each evaluated dimer, which is computationally expensive.

The polar solvation energy term is computed by solving the Poisson-Boltzmann (PB) equation with a finite-difference method using the *DelPhi* program (234). The aqueous solvent is considered to be a continuum with a dielectric constant (i.e. relative permittivity) of 80. For the solute (protein dimers), several values of the dielectric constant (from 1 to 32) were tested since there is not a single well established dielectric constant value for proteins in this protocol. Several authors have proposed this value to be relatively high, putting forward dielectric constant values in the range of 10 to 25 (232). Following this suggestion, we chose a value of 16, since it was the lowest in which the polar solvation contribution did not disproportionately overweight the other contributions of the binding energy. The protein dimers, with their corresponding charges and dielectric boundaries obtained from the GROMOS 54A7 force field, were mapped onto a grid in which the electrostatic potentials are calculated by solving the Poisson-Boltzmann equation. This calculation is performed for both monomers separately and for the dimer. Then, the electrostatic solvation contribution to the binding free energy is computed as the difference of the electrostatic solvation energy of the dimer minus the sum of the electrostatic solvation energies of both monomers.

The nonpolar solvation term was computed from a linear relation with the SASA translated in the expression $G_{\mathrm{np}} = \gamma SASA + b$, with G_{np} being the nonpolar solvation energy and with the parameters γ and b settled to 0.00542 kcal/mol.Å^2 e 0.92 kcal/mol (235). The electrostatics term was calculated using Coulomb's law with the atomic partial charges defined in the GROMOS96 54A7 force field using an interaction cutoff of 14 Å. The residual interactions longer than this cutoff are treated with a reaction field. The van der Waals term was computed using Lennard-Jones potentials defined in the same force field.

5.3.2 Evaluation of binding energy based on MM/PBSA

After the development and implementation of the new cost function for the MC-ED algorithm we submitted a set of (n=136) I_2 homodimers generated at pH 7.2 to a classical

protocol of structure refinement followed by MM/PBSA calculations of the binding energy. We used this procedure to select stable dimers for a more complete structural analysis using classical MD (Figure 25).

Although the energy terms used in MC-ED are quantitatively different from those included in the free energy calculation with MM/PBSA it is expected that a qualitative match can be found between some of them. In order to investigate this we computed correlations between the values of the different energy terms obtained with the MM/PBSA methodology and the values of the energy terms of the new cost function of our MC-ED (Table 9). We verified that there are moderate correlations between some of the MM/PBSA energy terms and some of the energy contributions of the MC-ED cost function. Indeed, we observed that, while there is no correlation between the solvation contributions of the two methodologies (i.e. the polar solvation term in the MM/PBSA and the hydropathic term in the MC-ED), the vacuum binding energies of the two methods present a moderate positive correlation between them. In particular, the electrostatic and the hydrogen bonding contributions of the MC-ED cost function both correlate with the electrostatic (Coulombic) term of MM/PBSA. Additionally, we also observed moderate correlations between the number of intermolecular contacts established in the dimers and the MM/PBSA vacuum binding energies, especially the van der Waals energy. This is reasonable as the van der Waals interactions are optimized in highly packed interfaces given their short-range nature. Another meaningful correlation is the one between the hydropathic contribution of the MC-ED and the van der Waals energy of the MM/PBSA as the van der Waals interactions are more frequent in nonpolar regions of the proteins, thus being closely coupled to the hydrophobic effect in proteins.

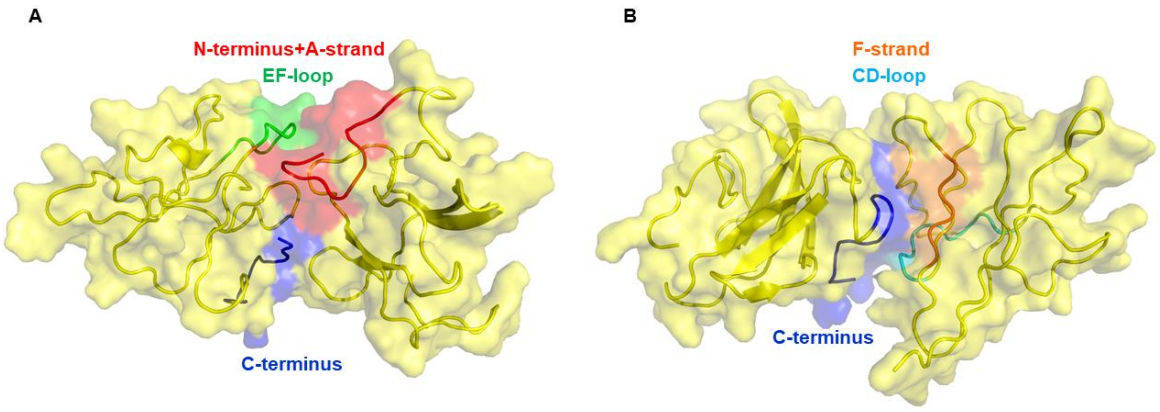


Figure 25. Representation of a dimer with low (A) and high (B) MM/PBSA binding energies.

	MC-ED total	Hydropathic	Electrostatic	Hydrogen bonds	Electrostatic+ Hydrogen bonds	Contacts	Clashes	Interfacial area
MM/PBSA total	-0.15	-0.36	0.35	0.56	0.47	-0.45	-0.55	-0.39
Polar solvation	-0.34	-0.39	0.12	0.19	0.16	0.22	0.10	0.23
Nonpolar solvation	0.31	0.13	0.22	0.40	0.31	-0.84	-0.56	-0.64
Total solvation	-0.30	-0.38	0.15	0.25	0.21	0.10	0.02	0.14
Electrostatic	-0.19	-0.39	0.34	0.53	0.46	-0.37	-0.53	-0.35
Van der Waals	0.56	0.49	0.02	0.11	0.05	-0.77	-0.36	-0.58
Molecular Mechanics total	-0.11	-0.32	0.34	0.54	0.45	-0.47	-0.56	-0.42
Interfacial area	-0.34	-0.16	-0.24	-0.32	-0.31	0.63	0.45	1.00

Table 9. Correlations between the MC-ED and MM/PBSA energy contributions. The correlations between the MM/PBSA energy contributions and the number of contacts and clashes in the dimers as well as with the interfacial area are also shown. The most relevant correlations in the analysis are highlighted with blue circles and squares.

5.4. Monte Carlo Ensemble Docking for tetramer prediction

After the prediction of dimer structures we adapted the MC-ED algorithm for tetramer prediction. To do so, we began by applying the MC-ED with the original cost function but we did not obtain an adequate sampling of the possible configurations of tetramers. Indeed, we observed that all of the tetramers were generated in a unique direction in a tail-to-tail arrangement as only this direction generates tetramers (i.e structures with a reasonable number of contacts between its constituent dimers) while in the other directions the dimers are apart from each other because the initial distance between the centres of mass of the dimers of 2x their gyration radius is too far for the dimers to interact with each other. As such, the initial conformational search always selects the same direction (i.e. the one that produces contacting dimers). Subsequently, we applied the new version of the MC-ED with the complete cost function and a modification in the sampling of the initial docking conformations that included a significant reduction of the distance between the centres of mass of the two dimers (to 0.46 of the initial distance) in the directions and/or orientations where they were not in contact. In the directions and/or orientations in which the tetramers had more than 300 clashes, the distance between the centres of mass was increased to 1.2 times the initial distance. By the end of this stage, the conformations with a maximum number of intermolecular contacts and a minimum number of clashes (< 350 clashes) are selected for the Monte Carlo docking stage. This allowed us to obtain an adequate conformational sampling of the possible β2-m tetramers, generating tetramers along different direction axes and orientations.

Chapter 6. Dimerization stage of the β2-m aggregation mechanism

A major goal of this PhD project is to carry out a comprehensive investigation of the dimerization phase of the aggregation mechanism of the D76N mutant of β2-m. In the follow up of previous studies carried out by Faísca et al. (57, 220), including one focused on the dimerization phase of the ΔN6 variant (57), we started by exploring the role of shape complementarity as a driver of the D76N mutant dimerization by using the original version of the MC-ED algorithm. The results of this work are reported in subsections 6.1 to 6.3 of the present chapter and were published as a regular article (S Loureiro RJ, Vila-Viçosa D, Machuqueiro M, Shakhnovich EI, FN Faísca P. A tale of two tails: The importance of unstructured termini in the aggregation pathway of β2-microglobulin. *Proteins*. 2017;00:1–13 (116)).

Subsequently, we investigated how different types of intermolecular interactions modulate the structure of the dimerization interface by means of MC-ED simulations. Specifically, we considered one cost function that optimizes the interface for shape complementarity and electrostatic interactions, and another cost function that optimizes the interface for shape complementarity and hydrophobic interactions. A comparative analysis of the dimers thus produced allowed us to determine if there is a mapping between interactions and structure, i.e., if specific intermolecular interactions drive the formation of dimers with peculiar structural features that can be exclusively ascribed to that specific type of intermolecular interaction.

Finally, we used the new version of the MC-ED algorithm, i.e. one that simultaneously optimizes the dimer's interface for shape, electrostatic (including hydrogen bonds) and hydrophobic complementarity, to carry a comparative analysis of the dimerization phase of the aggregation mechanism of the two β2-m variants, ΔN6 and D76N, including the analysis of a dimerization mechanism based on the prionlike hypothesis of Radford *et al.* (67).

We further extended our analysis to the tetramerization phase of β2-m aggregation by adapting the MC-ED algorithm to dock pairs of homodimers of the I_2 intermediate of D76N, i.e. the most aggregation-prone intermediate of D76N mutant, at pH 7.2, which is the biological relevant pH for the D76N mutant. The results of this work are reported in subsection 6.6 to 6.7 of the present chapter and were published as a regular article (JS Loureiro R, Vila-Viçosa D, Machuqueiro M, Shakhnovich EI, FN Faísca P. The early phase

of β2m aggregation: An integrative computational study framed on the D76N mutant and the ΔN6 variant. *Biomolecules* 2019, 9, 366 (236)).

6.1. Topology driven dimerization

In vitro experiments carried out by different research groups indicate that the formation of dimers is the first step of the aggregation pathway of β2-m (51, 99-101). In this study, we use protein-protein docking simulations to investigate the self-association of intermediate states I_1 and I_2 using the ensembles of conformations obtained by CpHMD simulations. We employ the *in–house* developed MC-ED procedure that optimizes dimer's interface for shape complementarity (220) as described in Chapter 5. Likewise, this study focuses on the topology component of a general interplay between topology and protein energetics, and this is why we put forward the concept of topology-driven dimerization.

Since the full-atomistic native-centric Gō model used to explore the folding transition predicts the existence of two intermediate states populating the folding space of the D76N mutant, we study the pH-dependent formation of (I_1-I_1 or I_2-I_2) homodimers and (I_1-I_2) heterodimers. As a control experiment, we also investigate the dimerization of the mutant's native conformation. We did not study the formation of heterodimers of wt and D76N monomers because it was shown in a recent study *in vitro* that the conversion of wt β2-m into amyloid fibrils can only be induced by the D76N mutant after the complete fibrillar conversion of the latter (237). Therefore, the *in vivo* formation of heterodimers of D76N and wt monomers with the capacity to aggregate into amyloids appears to be highly unlikely.

We began by computing the probability distribution function (PDF) for the number of intermolecular contacts at the considered pH (Figure 26). The PDFs are computed by fitting the probability histograms of the property under analysis (e.g. the number of intermolecular contacts, a discrete variable) to a Gaussian distribution.

The number of intermolecular contacts provides a quantitative measure of the quality of the geometric matching between the 2 monomers (i.e. shape complementarity), providing first glimpses on the dimerization potential of each intermediate species under different pH conditions. Our results predict that the intermediate I_2 is potentially more aggregation prone than I_1 since I_2-I_2 homodimers exhibit a considerably larger number of intermolecular contacts than I_1-I_1 homodimers at neutral and acidic pH. Indeed, the mode of the distributions (which represents the most likely number of intermolecular contacts within the ensemble of dimers) of the I_2-I_2 homodimers and I_1-I_2 heterodimers are clearly shifted to higher values than that of

the I_1-I_1 homodimers at all studied pH conditions (Figure 26A, B and C). Interestingly, the density curves of I_1-I_2 heterodimers are similar to those of I_2-I_2 homodimers, which suggests that the D76N mutant can initiate the aggregation cascade via two different pathways. Since the intermediate I_2 is exclusively populated by the D76N mutant, this finding is consistent with the mutant's higher amyloidogenicity as observed *in vitro*. Moreover, the analysis of the density curves also indicates that acidity enhances the aggregation potential of intermediate I_2, while intermediate I_1 conserves its aggregation propensity across the different pH values.

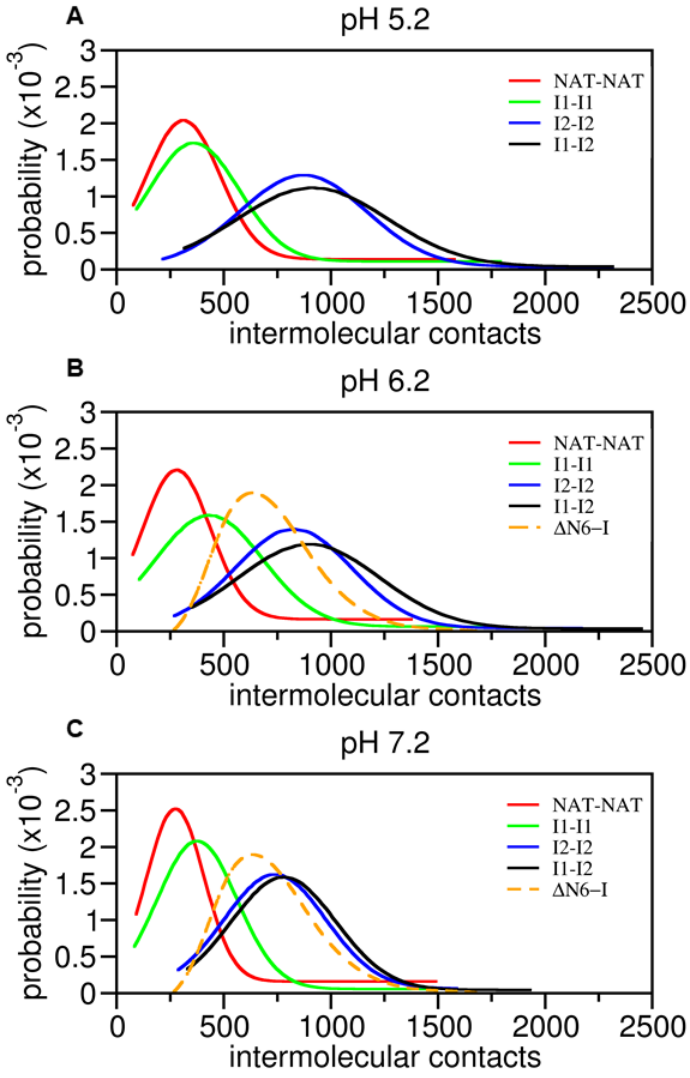


Figure 26. Intermolecular contact formation at different pH conditions. Probability density functions for the number of intermolecular contacts formed in different dimers of the D76N mutant at pH 5.2 (A), 6.2 (B) and 7.2 (C).

As outlined in Chapter 3, the ΔN6 variant of β2-m populates an aggregation prone intermediate state with a detached and unstructured N-terminus (57). This intermediate state is topologically similar to I_1 found here (note, however, that in I_1 the unstructured and

detached terminus is the C-terminus). By comparing the density curves for the number of intermolecular contacts in dimers of I_2 and in dimers of the ΔN6 intermediate at different pH values (Figure 26B and Figure 26C), our results predict that I_2 produces dimers with higher aggregation potential. This finding is in line with the experimental *in vitro* observation that D76N is considerably more amyloidogenic than the ΔN6 variant. We propose that intermediate I_2, which is unique to D76N, contributes to the higher aggregation potential of the D76N mutant relative to the wt protein and to the ΔN6 variant.

6.2. Protein regions involved in the onset of topology-driven dimerization

To identify which regions of the monomer are more likely involved in the formation of dimers at acidic and neutral pH, we computed intermolecular probability maps (IPMs) (Figure 27), which provide the probability of formation of each intermolecular contact in an ensemble of dimers representative of the monomeric state(s) under each considered pH (i.e. the ensembles of dimers used in the computation of the density curves). The IPMs are constructed by counting the frequency of each intermolecular contact in the dimer ensemble and then normalizing it to the total number of intermolecular contacts in the ensemble, thus computing a probability for each intermolecular contact. The distance cut-off used here to define an intermolecular contact corresponds to 1.25 times the sum of the van der Waals radii of the interacting atoms i.e. the distance cut-off considered in the Gō model used for studying the folding transition. The representative three-dimensional structures of the dimers are reported in Figure 28. The selection of these conformations was done based on two criteria: a representative dimer conformation should have a number of intermolecular contacts matching the mode of the density curves, and its interface should be representative of the IPMs.

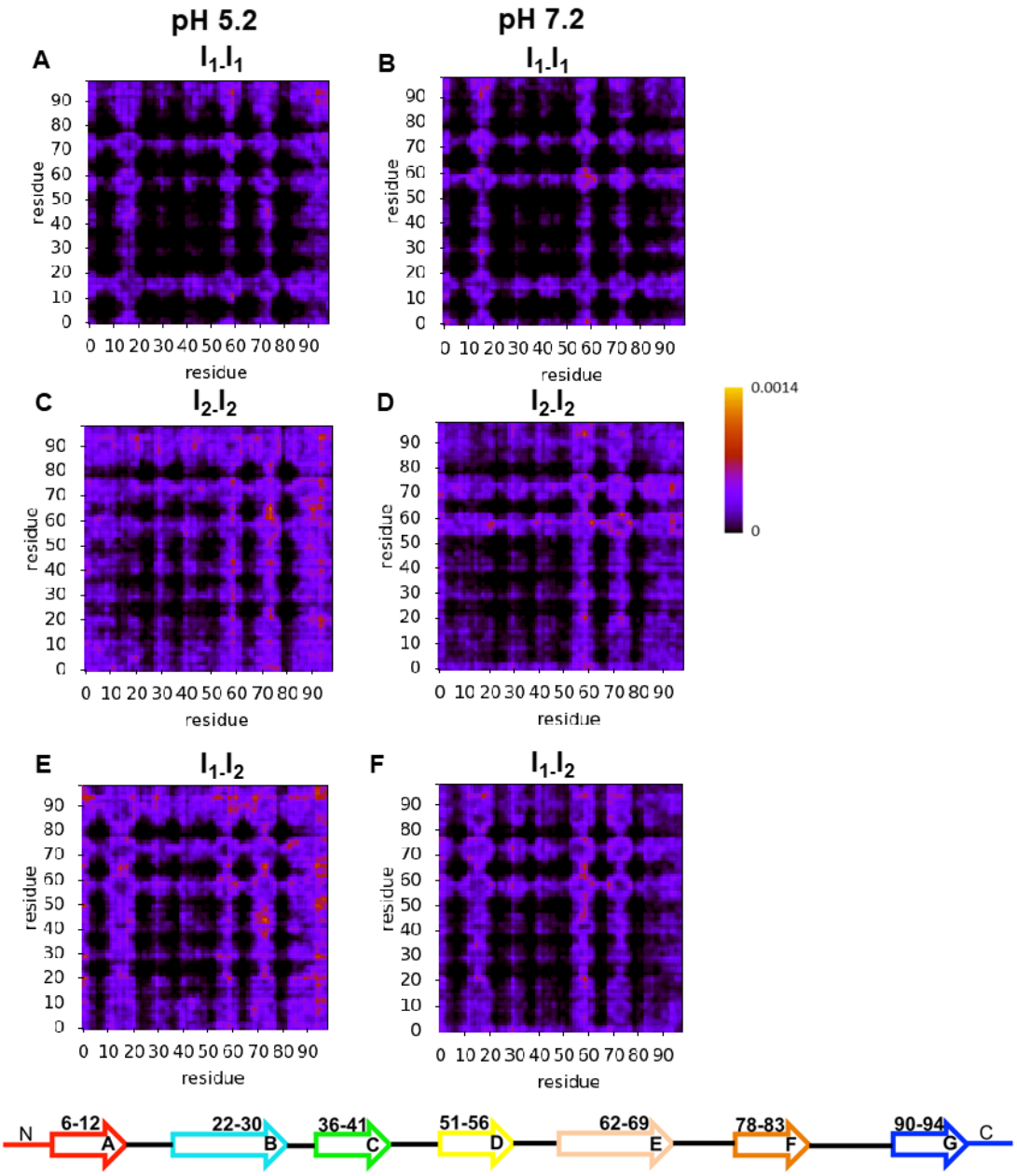


Figure 27. Probability maps for the intermolecular contacts established between monomers of different dimer structures at pH 5.2 (left) and pH 7.2 (right).

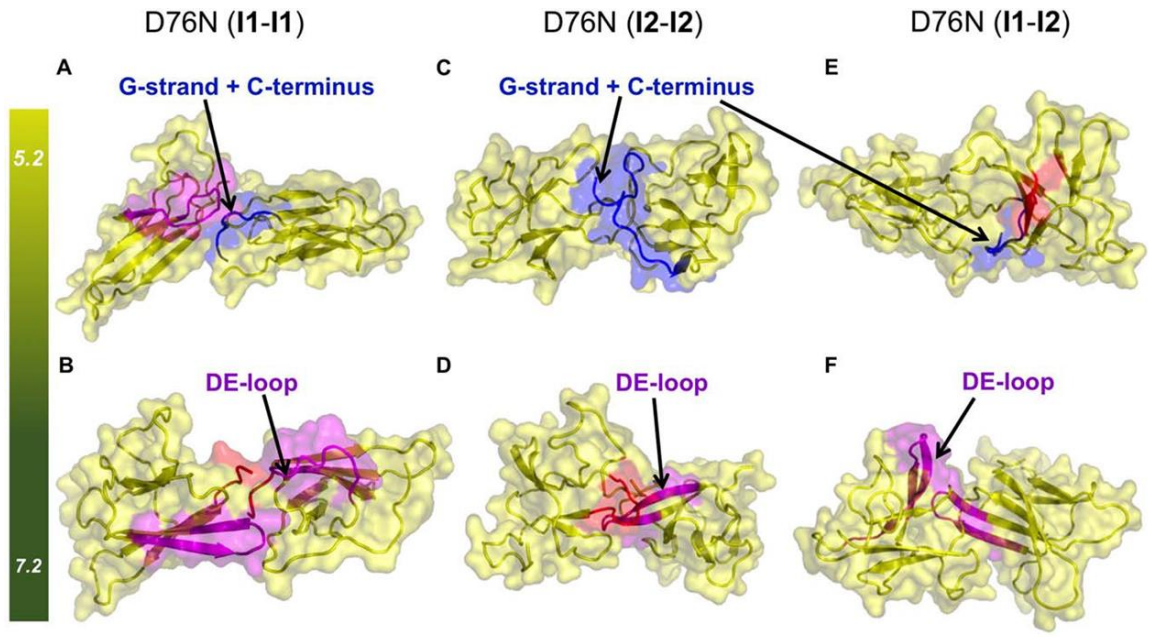


Figure 28. Representative dimer conformations. Three-dimensional representation of representative conformations of I_1-I_1 (A-B) and I_2-I_2 homodimers (C-D), and I_1-I_2 (E-F) heterodimers of D76N at pH 5.2 and 7.2.

The analysis of the IPMs indicates that under acidic conditions (pH 5.2), I_1 monomers associate preferentially via the C-terminus and the adjacent G-strand (Figure 27A and Figure 28A) while the formation of I_1 dimers at physiological pH is driven by the DE-loop (Figure 27B and Figure 28B). A similar association pattern is found for homodimers of the I_2 intermediate, which also associate via the G-strand and C-terminus at pH 5.2 (Figure 27C and Figure 28C). However, in this case, the dimerization interface can also involve the EF-loop of one monomer and the DE-loop and E-strand of the other monomer. Similarly, at physiological pH, the formation of I_2 homodimers is driven by the DE-loop (Figure 27D and Figure 28D).

The analysis of the IPMs of the heterodimers formed by I_1 and I_2 monomers highlights a major role of the G-strand and C-terminus in the dimerization at pH 5.2 (Figure 27E and Figure 28E), with the EF-loop driving an alternative but less likely dimerization pathway. At pH 7.2, the main dimer interface involves the DE-loop of one monomer and the DE-loop and E-strand of the other monomer (Figure 27F and Figure 28F).

To compare these results with those previously reported for the ΔN6 variant (57), we also evaluated the IPMs for the D76N mutant at pH 6.2 (Supplementary Figure 1 and Supplementary Figure 2). We observed that the N-terminal A-strand and the adjoining AB-loop play an essential role in the dimerization of the intermediate states at pH 6.2. A

distinctive role of these structural elements in dimerization appears to exclusive of pH 6.2. A relevant role of this region in β2-m aggregation at physiological pH has been proposed by several groups (59, 101, 103, 108, 109).

By taking into account the shape of the dimers resulting from the docking procedure, we predict that at pH 7.2 tetramerization of I_1-I_1 homodimers, I_2-I_2 homodimers, and I_1-I_2 heterodimers will be driven by "tail-to-tail" intermolecular interactions involving residues located at the C-terminus (I_1-I_1), and C- and N-termini (I_1-I_2, I_2-I_2), while at pH 5.2 dimer association will mostly involve the DE-loop region, in line with experimental data reported by different groups (59, 101, 103, 108).

6.3. Topology-driven aggregation hotspots

In the context of our analysis, the dimerization hot-spots are the residues that establish the largest number of intermolecular contacts upon dimerization, acting as triggers of the aggregation cascade. These are identified by computing the probability of intermolecular interaction per residue within the subset of the 50 most frequent intermolecular interactions. This analysis is important because it provides information that can be tested experimentally through *in vitro* studies.

In the homodimers of intermediate I_1, there is a cluster of residues located on the C-terminus (Trp95 and Arg97) and on the G-strand (Lys91 and Lys94), which are critical for dimerization at pH 5.2. We also highlight the importance of Lys19 (AB-loop), Trp60 (DE-loop), Glu74 and Lys75 (EF-loop). Lys94 and Trp60 are particularly prolific in establishing intermolecular contacts (Figure 29A). At physiological pH, we note a prevalence of hot-spots located on the DE-loop (Phe56, Asp59, Trp60 and Tyr63), with Asp59 clearly assuming a leading role as driver of dimerization (Figure 29B). Other important residues that may assist in the association of I_1 monomers at pH 7.2 are Glu16 and Asn17 (AB-loop), and Lys94 (G-strand).

Interestingly, despite having both termini detached and unstructured, the hot-spots residues triggering the dimerization of I_2 monomers are mostly localized on the DE-loop and upward regions under both pH conditions. Also interesting is the fact that their probabilities are significantly higher than those of the I_1 dimers, suggesting a more homogeneous aggregation pathway for this intermediate. Under acidic conditions (Figure 29C), dimerization of I_2 is essentially driven by Trp60 (DE-loop) and Trp95 (C-terminus). Tyr66 (E-strand), Glu74, Lys75, and Tyr78 (EF-loop) are also important to trigger dimerization. When the pH

is increased to physiological conditions the number of hot-spots located on the DE-loop (Phe56, Lys58, Trp60, Phe62 and Tyr63) increases significantly in line with what happens with I_1, and Trp60 assumes a clearly dominant role. Other important residues include Phe70 (E-strand), Glu74, Lys75, and Tyr78 (EF-loop), and Trp95 (C-terminus) (Figure 29D).

In the heterodimers formed by I_1 and I_2 monomers, residues Asp59 and Trp60 (DE-loop), Tyr66 (E-strand), Glu74 (EF-loop), Lys94 (G-strand), and Trp95 (C-terminus) are essential for dimerization under acidic conditions (Figure 29E). At pH 7.2, Trp60 (DE-loop) assumes a leading role, followed by Phe70 (E-strand) Tyr78 (EF-loop) and Trp95 (C-terminus) (Figure 29F). We observe an increase in the importance of the DE-loop in dimerization concomitantly with a decrease in the importance of the C-terminus and adjoining region with increasing pH.

To compare these results with those reported previously for the ΔN6 variant (57) we also evaluated the aggregation hot-spots for the D76N mutant at pH 6.2 (Supplementary Figure 2). We observe that the AB-loop (through residues Arg12, His13, Glu16, and Lys19) plays an essential role in the dimerization of both intermediates (this behavior is exclusive of pH 6.2). Additionally, Tyr10 at the end of the unstructured and detached strand A becomes important in homodimerization of I_2. This observation is in line with our previous findings for the ΔN6 variant at pH 6.2 (57), in which the detached and unstructured strand A (eg. Tyr10) and the adjacent AB-loop (eg. His13) of the aggregation-prone intermediate are essential for dimerization. However, the important role played by the C-terminus (and adjacent strand G) in the dimerization of both I_1 and I_2 intermediates, is not recapitulated by the intermediate state populated by ΔN6. This is likely the result of the unstructured and detached C-terminus, which is exclusively featured by both D76N intermediates. On the other hand, the important role played by the FG-loop (eg. His84) in the dimerization of the ΔN6 intermediate is not observed in the D76N mutant.

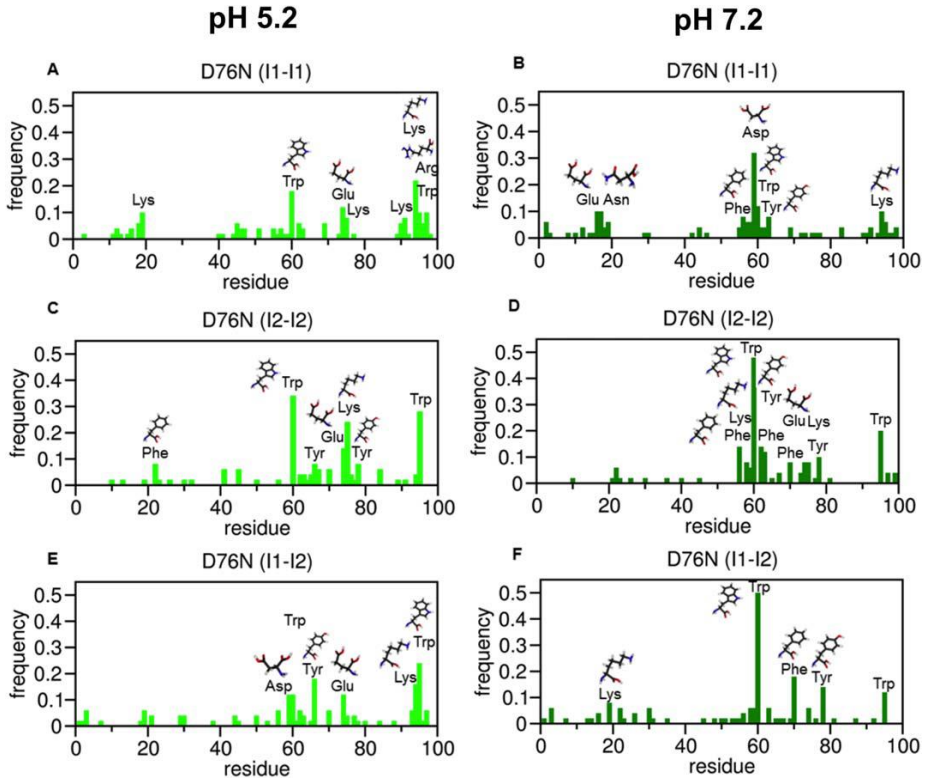


Figure 29. Dimerization hot-spots. Intermolecular contact probability per residue evaluated in the ensemble of the 50 most frequent intermolecular contacts formed in homodimers of I_1 and I_2, and heterodimers of I_1 and I_2 monomers of D76N at pH 5.2 (left) and 7.2 (right).

6.4. HADDOCK dimer prediction

To gauge the quality of the results obtained within the MC-ED, we used the HADDOCK docking method (188, 238), which, as outlined in chapter 4, is based on a more complete scoring function. The starting conformations used in HADDOCK are the monomers forming the representative dimers reported in Figure 28. We used as active restraints the dimerization hot-spots (i.e. residues that trigger the aggregation cascade) that were identified through *in vitro* experiments (e.g. Glu16, Lys19, Phe56, Asp59, Trp60, Phe62 and Tyr63) (59, 101, 103, 108) as well as those predicted in the context of the present analysis. We verified that the dimer conformations corresponding to the best HADDOCK scores have interfaces similar to the ones obtained within the scope of the MC-ED (Figure 30), which confirms a key role played by shape complementarity in protein-protein association (130, 221).

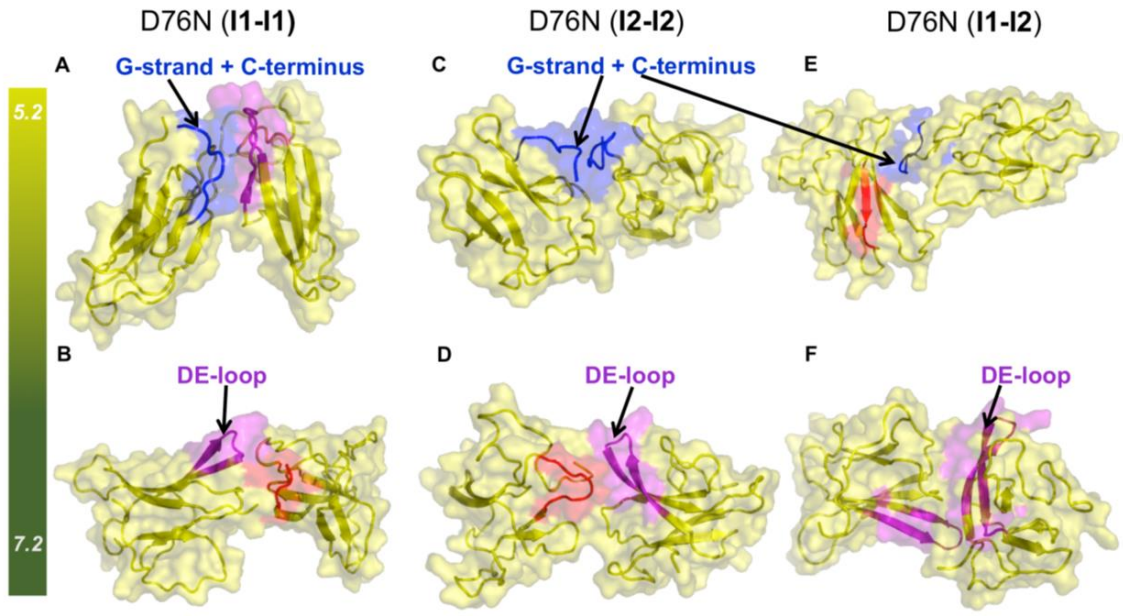


Figure 30. Three-dimensional structure of representative conformations of D76N-I_1 and D76N-I_2 homodimers, and D76N-I_1-I_2 heterodimers at two different pH values (5.2 and 7.2) obtained by the HADDOCK docking method. The active restraints used were the dimerization hotspots identified through *in vitro* experiments as well as those predicted by MC-ED.

6.5. From intermolecular interactions to dimer interface

After exploring the role of shape complementarity in β2-m dimerization we went on to explore how dimer structure is modulated by the different types of intermolecular interactions, namely hydrophobic and electrostatic interactions, and hydrogen bonds, which are recognized driving forces of protein-protein association (130-132). We focused our analysis on the intermediate state I_2 populated by the D76N mutant and on its association at physiological pH. Operationally, we conducted a set of MC-ED simulations with a cost function that optimizes I_2-I_2 dimers for shape and hydrophobic complementarity, and conducted another set of MC-ED simulations driven either by a cost function that optimizes I_2-I_2 dimers for shape and electrostatic complementarity, or by a cost function optimizing I_2-I_2 dimers for shape complementarity and hydrogen bonds (i.e. a specific type of electrostatic interaction). For comparative purposes a set of simulations with the complete cost function was also performed. The data analysis was the same as that reported in the previous section, which includes the determination of structural regions and residues that are more likely to trigger the dimerization phase. At this point it is important to mention a technical detail in the evaluation of the IPMs. For the purpose of evaluating intermolecular contacts in a dimer whose interface was exclusively optimized for hydrophobic complementarity we only

consider contacts whose interaction distance is within the cut-off of the corresponding interaction well. The same applies to the other considered interaction types.

We start by analyzing the density curves for the binding energy of the dimers generated with each considered cost function (Figure 31). We observed that the cost function with the hydrophobic interactions generates the broadest distribution of the binding energy, which suggests that this type of interactions creates dimers which are structurally more heterogeneous. On the other hand, the cost functions that consider electrostatic interactions (including the one that considers specifically hydrogen bonds) generates narrow distributions of the binding energy with the mode (which represents the most likely binding energy within the ensemble of dimers) centered in lower values than that of the hydrophobic interactions. This indicates that the electrostatic interactions may generate more structurally homogeneous dimers. The set of simulations performed with the complete cost function generates a broad distribution for the binding energy, which suggests that, while the hydrophobic interactions contribute to the broadness of the distribution, the electrostatic interactions (including the hydrogen bonds) have an additive effect on dimer's binding energy, contributing to lower the binding energies of this distribution in comparison with the other distributions.

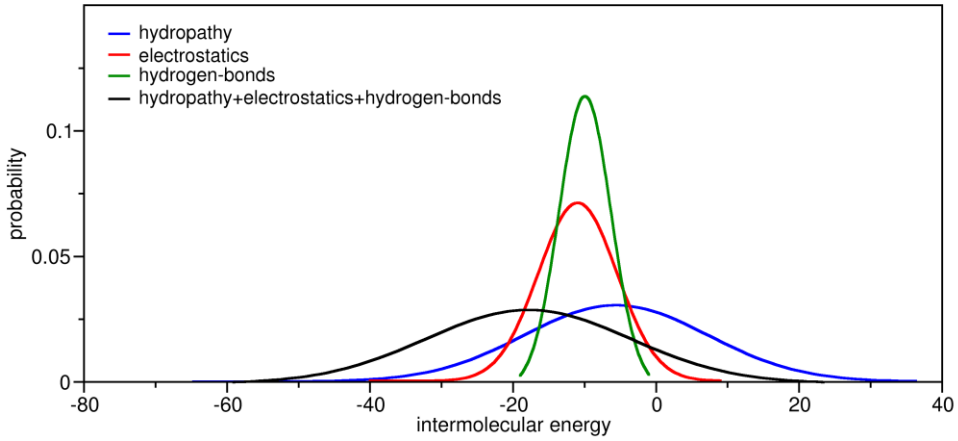


Figure 31. Probability density functions for the intermolecular energy of the I_2 homodimers at pH 7.2 generated by MC-ED simulations with each individual cost function contribution. It is also represented the probability density function for the intermolecular energy of the dimers generated by the correspondent MC-ED simulation with the complete cost function.

The analysis of the IPMs and of the difference probability maps (i.e. maps representing the differences in intermolecular contact probability measured with respect to the IPM obtained exclusively for shape complementarity) (Figure 32) indicates that the

hydropathic contribution drives the formation of interfacial regions with bulky hydrophobic residues in the dimer's interface which include the BC-, the DE- and the EF-loops (Figure 32A). The BC- and EF-loops assume a particularly more prominent role in interfaces optimized for hydrophobic complementarity than in those optimized exclusively for shape complementarity, in which the DE-loop, which is rich in bulky residues, is the predominant region (Figure 32B). The electrostatic contribution favours the presence in the interfaces of regions rich in charged residues like the EF-loop, the CD-loop and the C-terminus (Figure 32C, Figure 32D and Figure 33). In turn, the hydrogen bonds contribution favours the presence of interfacial regions with polar residues such as the CD-loop, the EF-loop and the C-terminus (Figure 32E), which are clearly more prevalent in hydrogen bond driving dimer's interfaces. Perhaps, not surprising, the latter show a strikingly markedly decrease of the presence of the DE-loop relative to the other cost functions (Figure 32F).

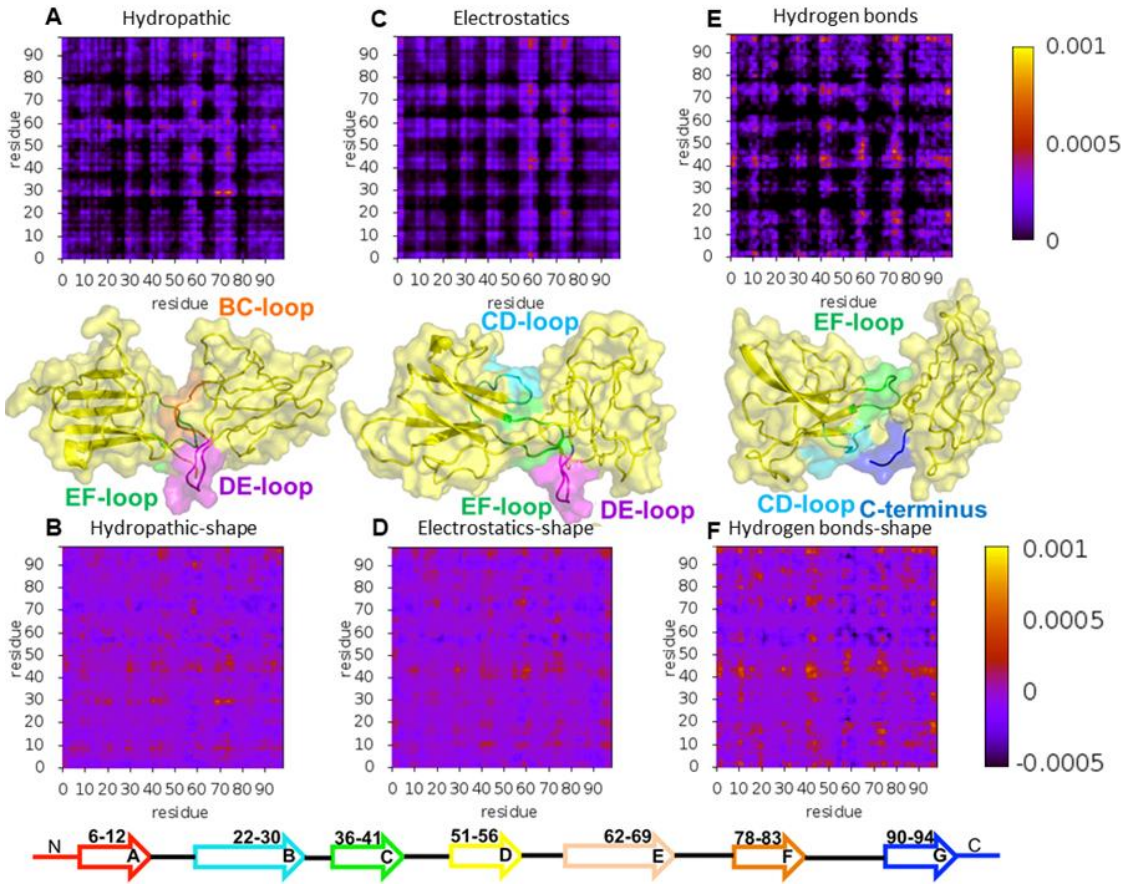


Figure 32. Intermolecular probability maps (IPMs) for the dimer ensembles generated by the docking simulations with each individual energy contribution (ensemble of I_2 of D76N at pH 7.2) and difference IPMs of the latter in relation to the original shape-based MC-ED IPMs. Representative

dimer conformations of the ensembles generated with each individual contribution of the new MC-ED cost function are also represented.

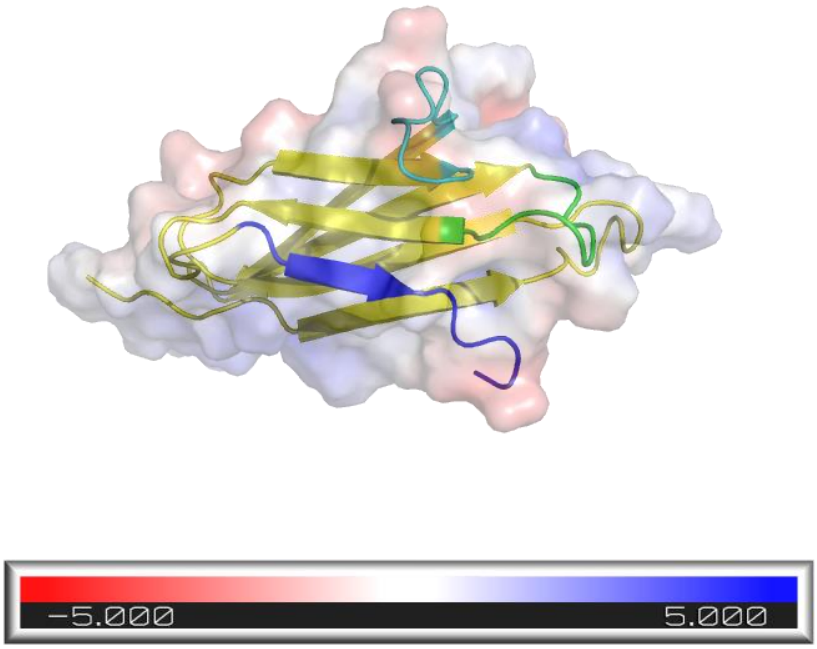


Figure 33. Representation of the surface electrostatic potential of the D76N mutant of β2-m with blue corresponding to positively charged regions and red corresponding to negatively charged regions as computed with the APBS plugin of PyMOL. The CD-loop, the EF-loop and the C-terminal region are highlighted in cyan, green and blue, respectively.

Finally the analysis of the dimerization hotspots highlights the following observations:

1) In the interfaces optimized for hydrophobic complementarity the hotspot residues are Tyr10, Phe22, Phe30, Phe56, Trp60, Phe62, Phe70, Lys75 and Trp95 (Figure 34A). By comparing these results with those obtained for shape complementarity we note an increase in the frequency of hydrophobic residues such as Tyr10 (A-strand), Phe22 (B-strand), Phe62 (DE-loop), Phe70 (EF-loop) and, most strikingly, Phe30 (BC-loop) (Figure 34A and Figure 36A). This frequency increase ranges from 0.04 (Phe22 and Phe70) to 0.2 (Phe30). Nevertheless, we still observe a prevalent role of DE-loop residues (e.g. Phe56, Trp60, Phe62) (although not so marked) (Figure 34A and Figure 36A), which reflects the fact that these residues are simultaneously bulky and hydrophobic.

2) In the interfaces optimized for electrostatic complementarity the hotspots are Arg3 (N-terminus), Phe22, (B-strand), Arg45 (CD-loop), Trp60 (DE-loop), Phe62 (DE-loop), Lys75 (EF-loop), Trp95 (C-terminus) and Arg97 (C-terminus) (Figure 34B). By comparing the results with those obtained for shape complementarity we highlight an increase in frequency of the charged residues Arg3 (N-terminus), Arg45 (CD-loop), Lys75 (EF-loop) and Arg97 (C-terminus), ranging from 0.06 (Lys75) to 0.14 (Arg97) (Figure 34B and Figure 36A).

3) In the interface optimized for hydrogen bonds the aggregation hot-spots are Arg3 (N-terminus), Arg12 (A-strand), Asn17 (AB-loop), His31 (BC-loop), Asn42 (CD-loop), Arg45 (CD-loop), Asp59 (DE-loop), Glu74 (EF-loop), Lys-75 (EF-loop) and Arg97 (C-terminus) (Figure 34C). A comparison of these results with those obtained for shape complementarity indicate an increase in the number of polar residues identified as hotspots such as Arg12 (A-strand), Asn17 (AB-loop), His31 (BC-loop), Asn42 (CD-loop), Arg45 (CD-loop), Asp-59 (DE-loop) and Arg97 (C-terminus), and an increase in the frequency of Glu74 (EF-loop) of 0.04 (Figure 34C and Figure 36A).

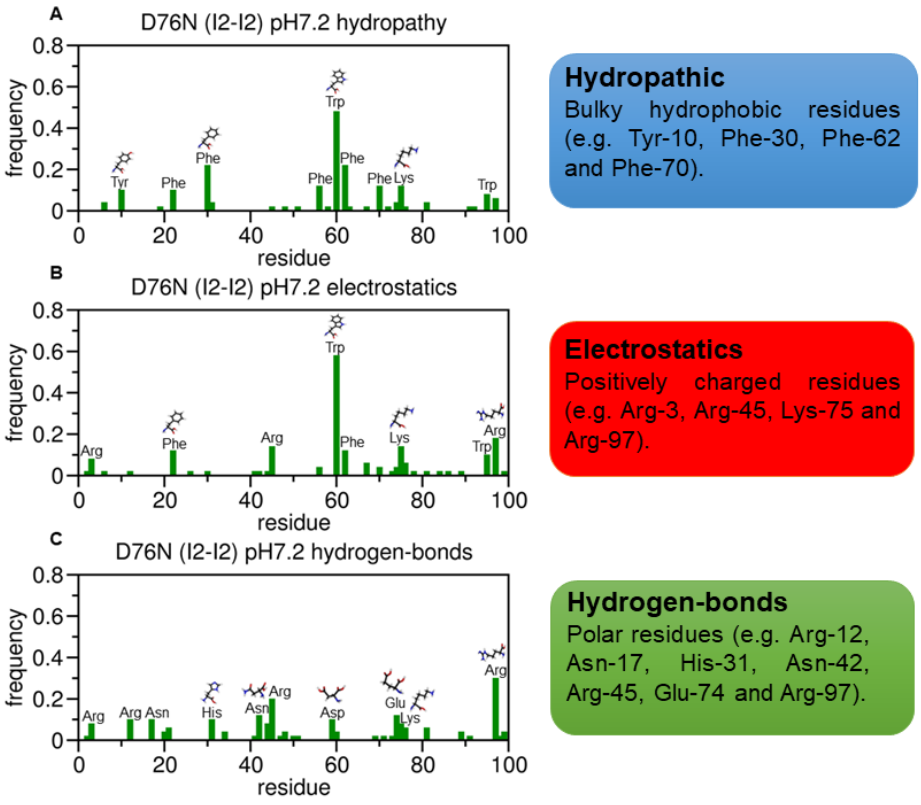


Figure 34. Dimerization hot-spots in homodimers of I_2 at pH 7.2 obtained with MC-ED simulations using individually each contribution of the new cost function: hydropathic (A), electrostatics (B) and hydrogen bonds (C).

Finally we compared the IPM of the conformational ensemble generated by the MC-ED with the complete cost function with that generated by the MC-ED with the original cost function, uniquely based on shape complementarity (Figure 35). The IPM corresponding to the new cost function is computed by verifying if each atom pair meets the interaction criteria of each considered interaction type and, if so, by assigning it to the corresponding interaction type(s) and counting it as an intermolecular contact. When a given atom pair is involved in more than one type of interaction (e.g. in an electrostatic (coulombic) interaction and in a hydrogen bond), it is counted only once. We note that, although the two IPMs exhibit an overall similar pattern with a high prevalence of the DE- and EF-loops in dimer's interface, there is a marked increase in the presence of the CD-loop (Figure 35A and Figure 35B) with the new cost function. This likely reflects the contributions of the electrostatic interactions and hydrogen bonds in the new cost function while the overall similar pattern likely reflects the major influence of shape complementarity on protein-protein association. Additionally, the hydrophobic effect, essential in protein-protein association, is indirectly captured by the original shape-based cost function as most of the hydrophobic residues have bulky side-chains (e.g. aromatic residues), which could partially account for the similarity in IPMs.

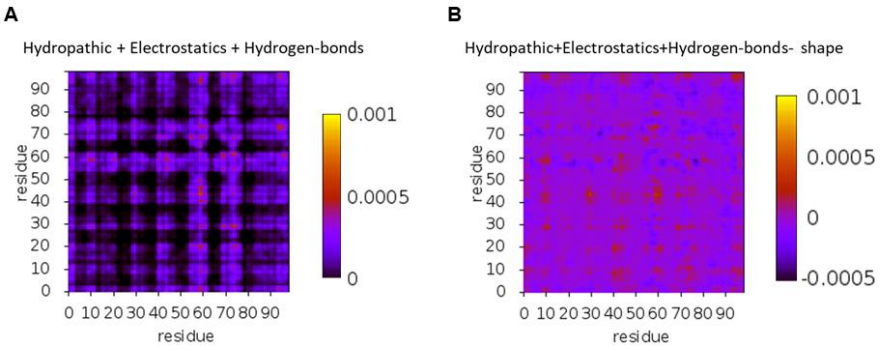


Figure 35. Intermolecular probability map (IPM) for the ensemble of D76N I_2 dimers generated by the MC-ED with the complete cost function and difference IPM of the latter in relation to the original shape-based MC-ED IPM.

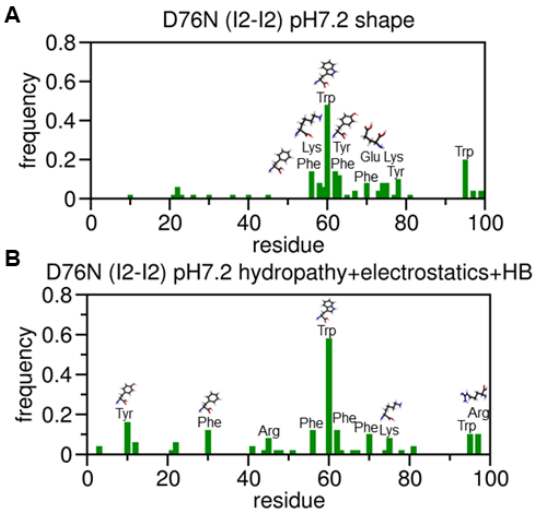


Figure 36. Dimerization hot-spots in homodimers of I_2 at pH 7.2 obtained with MC-ED simulations using the original version of the method (A) and the new version of the method with the complete cost function (B).

The analysis outlined above indicates that the new cost function is physically consistent. Indeed, each type of intermolecular interaction gives rise to an ensemble of dimers with interfaces predominantly stabilized by residues with physicochemical properties consistent with the establishment of that particular type of interaction. Additionally, the profile of hotspots of the ensemble of dimers generated with the complete cost function (Figure 36B) indicates that the three types of considered interactions are involved in the stabilization of the interface, showing that the new cost function is correctly capturing the different types of intermolecular interactions relevant to protein-protein association, at least at a qualitative level.

6.6. Integrated analysis on the β2-m dimerization mechanism

6.6.1 Dimer stability under different pH conditions

We started by computing the PDF for the binding energy of dimers whose monomers are representative of the intermediate states (I, I_1 and I_2) extracted from CpHMD trajectories at several pH conditions. For D76N, we considered acidic pH 5.2 to gain insight into how pH may modulate the dimerization pattern of this mutant, although it is not a biologically relevant pH. In the case of ΔN6, however, we considered the slightly acidic pH 6.2, which is biologically relevant since it occurs at the inflamed joints. For comparative purposes we also computed the PDF for the binding energy of dimers formed by monomers representing the native state of the D76N mutant under different pH conditions. We further evaluated the PDF for the number of intermolecular contacts, which shows that the dimers produced with the

deployed docking exhibit a similar degree of compactness, with interfaces reaching ~5K intermolecular atomic contacts (and 250-300 atomic clashes on average).

Under acidic conditions (pH 5.2) the PDF for the binding energy is conserved across the dimers formed by monomers of D76N intermediate states (Figure 37A). The mode of the distribution is reasonably the same (E~-18) for dimers of the intermediate I_2 (i.e. I_2-I_1 and I_2-I_2 complexes) and a little higher (E~-16) for homodimers of I_1.

At pH 7.2 there are noticeable differences in the PDFs (Figure 37C). First, the PDF corresponding to I_1-I_1dimers shifts towards higher binding energies and its mode is now E~-13. This stability loss may be linked to the deprotonation of His84, which is coupled to a smaller detachment of the C-terminal region. The loss of C-terminal mobility implies that this region becomes less available to participate in intermolecular interactions. The PDF of I_2-I_1 dimers fairly conserves the mode (E~-17) while I_2-I_2 dimers get slightly more stable (E~-19) and clearly more stable than I_1-I_1 dimers at physiological pH. Furthermore, in this case, the tails of the distribution extend towards lower energy values with higher probability than at pH 5.2. Taking the binding energy alone as a proxy of dimer stability, one can predict that, at physiological pH, homodimers of I_2 will be the most stable. Since dimers must be stable enough to oligomerize further, it is also likely that homo- and heterodimers of I_2 are more prone to aggregate than I_1 homodimers and are, therefore, the key species in D76N aggregation. We note however that, if dimers are too stable, they are likely to remain soluble. Therefore, the most stable dimers, i.e. those pertaining to the tails of the distributions, are not necessarily the ones that will grow into fibrils.

It is interesting to compare the behavior of D76N with ΔN6, which populates the intermediate I. At pH 7.2 and 6.2, I-I dimers have binding energies (E~-19) similar to D76N I_2-I_2 dimers, suggesting similar aggregation potential for these two β2-m intermediates.

Finally, we computed the PDF for the binding energy of heterodimers formed by the native state of ΔN6 and the native state of wt, which are the species involved in the 'prion-like hypothesis', at pH 6.2 (Figure 37B). The distribution is strikingly shifted towards higher energies with the mode located at E~-6, indicating that these dimers will be weakly bound. This observation is in line with experimental evidence based on NMR measurements reported by Radford and co-workers (239), who, nonetheless, argued that such a weak binding can still induce conformational changes in the wt β2-m protein as well as on ΔN6 itself, while a

stronger binding may block the conformational alterations characteristic of a prion mechanism.

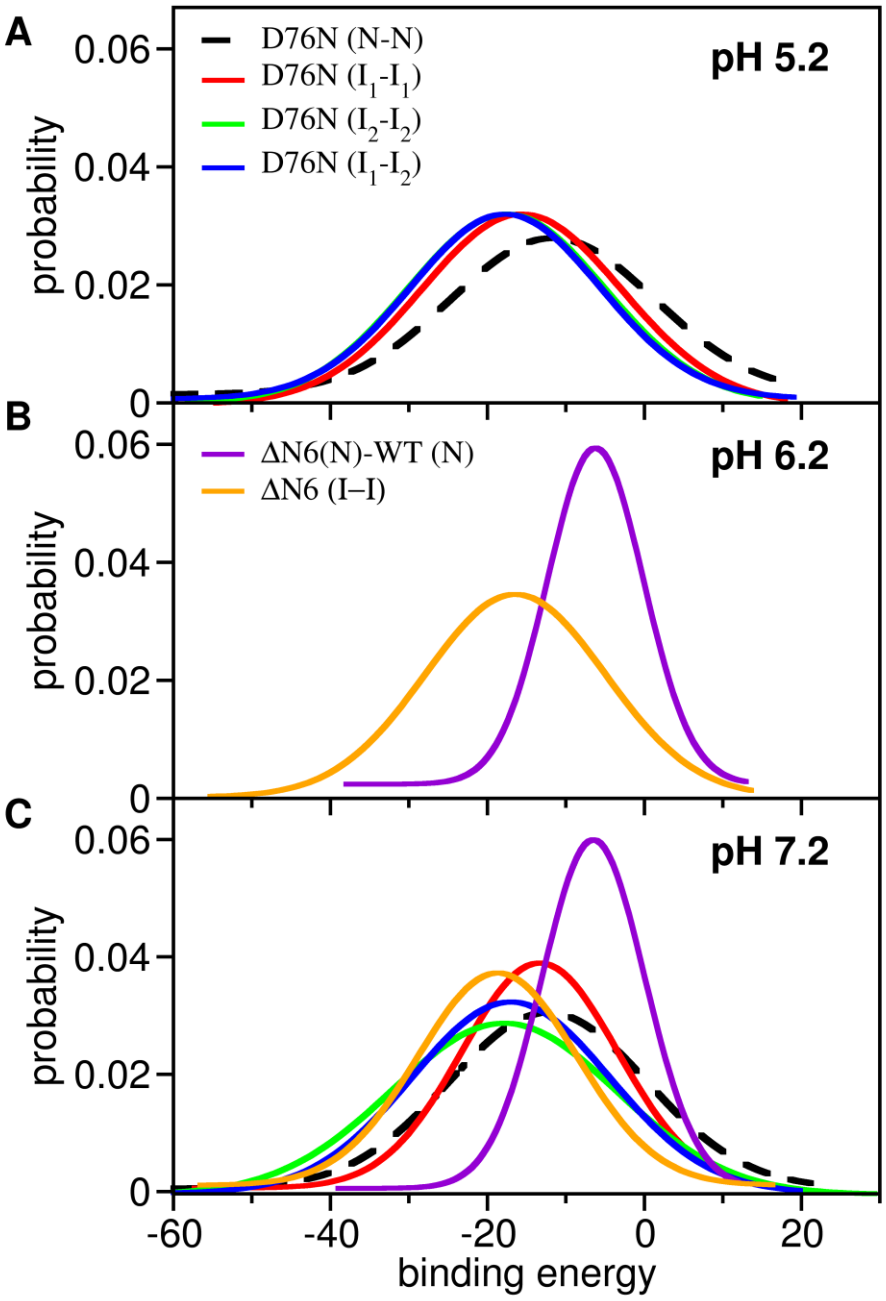


Figure 37. Probability density curves for the distribution of intermolecular energy evaluated in each considered ensemble of dimers at acidic pH (A), slightly acidic pH (B) and neutral pH (C).

6.6.2. Structure of D76N dimers under different pH conditions and dimerization hot spots

To get insight into the structural features of the D76N dimer's interface and identify which regions of the monomers are more likely involved in the formation of dimers at acidic and neutral pH, we compute the IPMs. To properly count contacts within a dimer, it is important to recall that the binding energy now contains contributions from three interaction potentials. A representative dimer conformation is reported together with the IPM (Figure 38).

The analysis of the IPMs (Figure 38) indicates that the DE-loop and EF-loop behave as adhesion zones in the association of the two considered intermediate states at pH 7.2 (Figure 38A-C). Their importance, however, is more evident in the formation of I_2-I_2 dimers at physiological pH. It is possible that the detachment of both the N- and C- terminal regions from the protein's core in the I_2 intermediate state facilitates (and enhances) the movement of the DE- and EF-loops, in line with observations reported in (73). Since this enhancement is stronger at pH 7.2 (Table 4), the loops will more likely establish intermolecular interactions at physiological conditions. The intermolecular interactions involving these loops become less likely at acidic pH but their fingerprint is still noticeable in the IPMs (Figure 38E), with Phe70 (EF-loop) behaving as a hotspot residue (Figure 39B). The leading hotspot residue at physiological pH is clearly Trp60 (DE-loop), whose role as an interaction hub sharply decreases as the pH is lowered to 5.2 (Figure 39B). Indeed, under acidic conditions the dimerization of I_2 is majorly triggered by Arg3 (N-terminus), followed by two clusters of residues located on the DE-loop and adjoining D-strand (His51, Phe56 and Trp60), and, to a lesser extent, on the EF-loop and adjoining E-strand (Tyr67, Phe70 and Lys75) (Figure 39B). We also pinpoint the participation of Arg3 (N-terminus), Tyr10 and Arg12 (A-strand) in the association pattern of homo- and heterodimers, particularly at physiologic pH (Figure 39A-C).

Under acidic pH, the C-terminus gains relevance as an adhesion zone in the heterodimers and, more strikingly, in homodimers of I_1 (Figure 38D), possibly due to an increased detachment of the C-terminus, which is coupled with increased protonation of His84 (FG-loop) (pKa~5.2). The AB loop also stands out as an important structural element in I_1-I_1 dimerization, establishing preferential interaction with the EF-loop and AB-loop of the other monomer, as well as with the C-terminus. At pH 5.2, I_1 monomers associate mainly through Trp95 and Arg97 (C-terminus), followed by His13 and Lys19 (both at the AB-loop) (Figure 39A). The latter are also leading hotspots in dimerization of the heterodimers (Figure

39C), where His51 (D-strand) also acts as hotspot because of its increased protonation at acidic pH (pKa~6.5). We stress that a charged histidine can establish ionic interactions with carboxylic acids, on top of hydrogen bonds.

It is interesting to compare these results with those obtained with the cost function that optimizes dimers exclusively for shape complementarity. The DE- and EF-loops were already rather important adhesion zones in the dimerization of I_2 at acidic and physiological pH. However, the C-terminus and the (unstructured) C-terminal region (C-terminus and G strand) played a strikingly dominant role in I_2 dimerization at acidic pH, which is substantially suppressed when interactions other than packing are considered. Indeed, the electrostatic interactions involving polar and charged residues of regions such as the N-terminal A-strand are likely accountable for the observed differences. As for the dimerization of I_1, the AB-loop was already an important driver of monomer association also at physiological pH as well as the C-terminus at acidic pH, similarly to the current results.

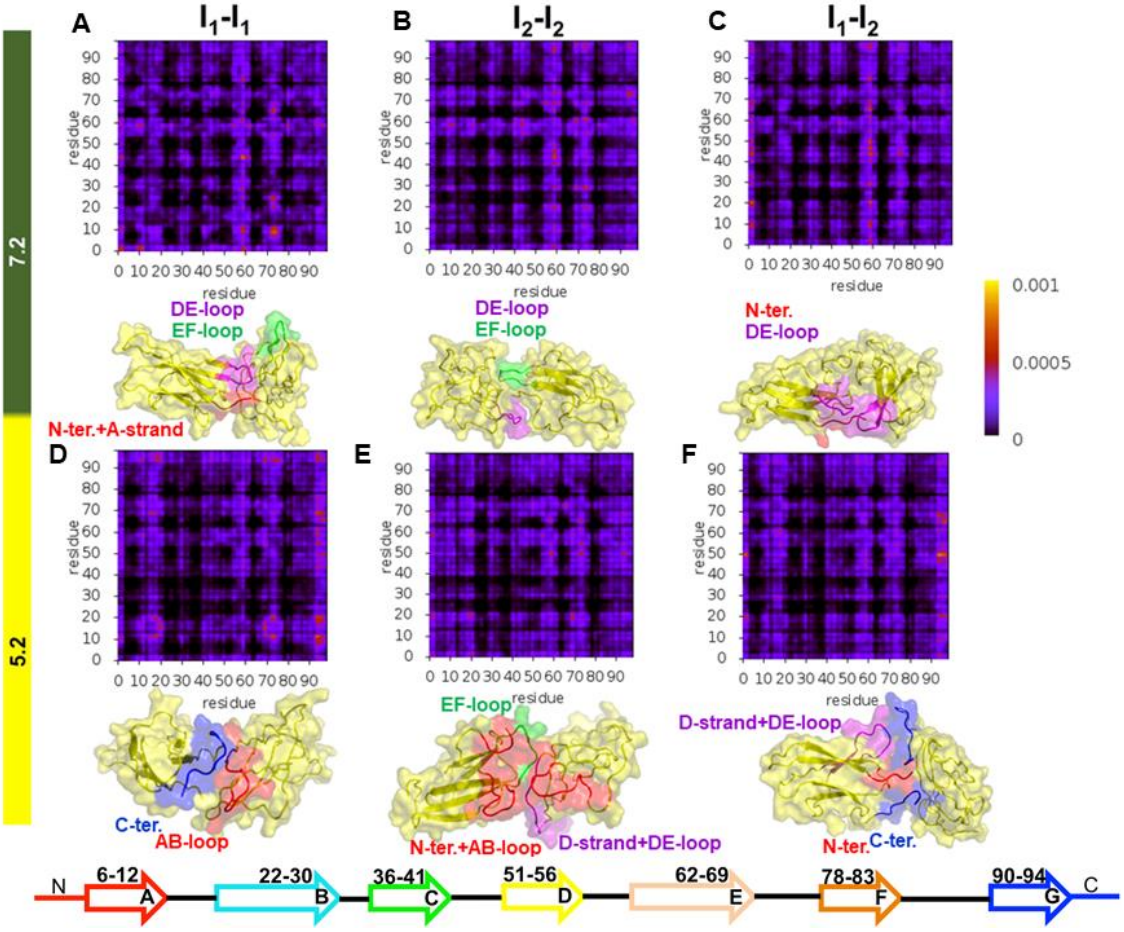


Figure 38. Structural regions involved in D76N dimerization. Probability maps for intermolecular contacts forming at the interface of dimers of the intermediate states populated by the D76N mutant,

and three-dimensional representation of representative dimer conformations, i.e. a conformation with energy matching the mode of the PDFs.

It is worth noting the increased relevance of the N-terminal region in dimerization observed with the new MC-ED as compared with the results obtained with the original MC-ED. This may result from electrostatic interactions (including hydrogen bonds) involving polar and charged residues such as Arg3, Arg12, His13 and Lys19 that were not taken into account in the original version of the MC-ED, solely based on shape complementarity.

Based on the structure of the dimers generated by the docking procedure, we predict that a likely pathway for further aggregation (e.g. tetramerization) of I_1-I_1 and I_1-I_2 homodimers at pH 5.2 will be through interactions involving the DE- and EF-loops. In contrast, formation of tetramers from the same dimers at pH 7.2 may proceed through "tail to tail" interactions involving the two termini. The I_2 homodimers can tetramerize through interactions involving the C-termini at pH 5.2 while at physiological pH dimer association may likely proceed by either of the two termini. From these observations, emerges the evidence that either of the dimers of the D76N intermediates generated by our MC-ED algorithm have feasible pathways to further oligomerize and to then give rise to amyloid fibrils.

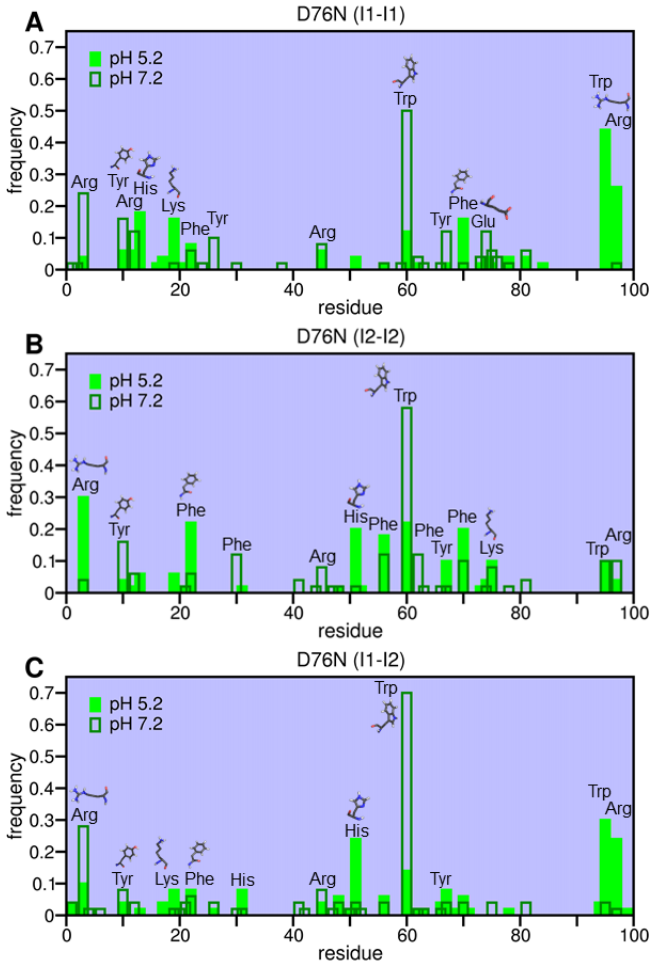


Figure 39. Dimerization hot-spots. Intermolecular interaction probability per residue evaluated in the ensemble of the 50 most frequent intermolecular interactions formed in homodimers of I_1 (A) and I_2 (B), and heterodimers of I_1 and I_2 (C) monomers of D76N at pH 5.2 and 7.2.

6.6.3. Structure of ΔN6 dimers under different pH conditions and dimerization hot spots

Here, we extend the analysis to the ΔN6 variant. We perform a comparative analysis of the dimerization phase of the intermediate I with that triggering a prion-like templating mechanism (Figure 40). According to the latter, the conversion of wt β2-m into an aggregation prone conformer is induced by bimolecular collision between the wt protein and the ΔN6 mutant. Therefore, we investigate the structure of dimers formed by the native structure of the wt protein and the native structure of ΔN6.

The analysis of the IPMs reveals that at pH 7.2 the homodimers of the intermediate state I populated by ΔN6 (Figure 40A) associate through the DE-loop and BC-loop, and, to a lesser extent, via the FG loop and the C terminus. This dimer arrangement allows for further oligomerization through the unstructured and detached A-strands. When the pH is lowered to

6.2 the N-terminal region (comprising the A-strand and AB-loop) becomes an important adhesion zone (Figure 40C), in part due to its higher mobility (Table 5). We also computed the IPMs for the D76N intermediates at pH 6.2 (Supplementary Figure 3) for comparison purposes with the ΔN6 intermediate. We observe that the N-terminal region, including the A-strand and the adjoining AB-loop, have a similarly important role in dimerization at this slightly acidic pH (Supplementary Figure 3 and Supplementary Figure 4). This could be due to an increase in the protonation of His13 located at the beginning of the AB-loop because of the proximity of the pKa of this residue side-chain to the environmental pH 6.2 (Table 1). This protonation event is coupled with the increased detachment of the A-strand from the protein core (Table 3, Table 4 and Table 5), which renders it more available to establish intermolecular interactions with neighboring monomers, thus triggering the dimerization process. The dimers with interfaces involving the A-strand have straightforward pathways to further oligomerize through the involvement of other regions of the monomer and/or by a free A-strand from the other monomer. The DE-loop conserves its importance and the interactions involving the FG-loop gain relevance. The major difference between the current IPMs and those obtained before (57), i.e. with a cost function that optimizes shape complementarity, is a more evident fingerprint for the BC-loop presumably due to the role of electrostatic interactions established by the His31. The role of electrostatic interactions (including hydrogen bonds) can be observed in the increased frequency of polar and/or charged residues identified as hotspots. It is interesting to note that Trp60 is an important aggregation hotspot at both pH values considered (Figure 41A), but its role as an interaction hub in ΔN6 is significantly downgraded when compared with the results for D76N. Phe30 (BC-loop) conserves an essential role as hotspot when the pH is lowered, while His84 and Thr86 (FG loop) saw an enhancement at acidic pH. The hotspot character of Arg97 (C-terminus) is enhanced at physiological pH, while that of Tyr10 (A-strand) clearly stands out at pH 6.2, presumably because of the increased mobility of the N-terminal region.

The IPMs for the heterodimers formed by the native state of wt β2-m and the native state of ΔN6 reveal an important role of the DE-loop (especially at pH 6.2) and of the CD-loop (more pronounced at pH 7.2) in the dimerization process (Figure 40B and D). Under physiological pH, the F-strand and the FG-loop also participate in the association process, although to a less extent. These results are in line with those reported by Radford and co-workers (239), that claimed the involvement of the DE-loop, BC-loop and FG-loop in the interfaces of the heterodimers of ΔN6 and wt β2-m. Trp60 stands out again as an aggregation

hotspot (especially at pH 6.2), and Arg45 (CD-loop) at both physiological and pH 6.2 (Figure 41B). Arg81 (F-strand) is also an important linker at pH 7.2 followed by Arg97 (C terminus), as well as Arg3 (N terminus) at pH 6.2. It is likely that the intermolecular interactions between the positively charged arginine residues may contribute to destabilize the interfacial region.

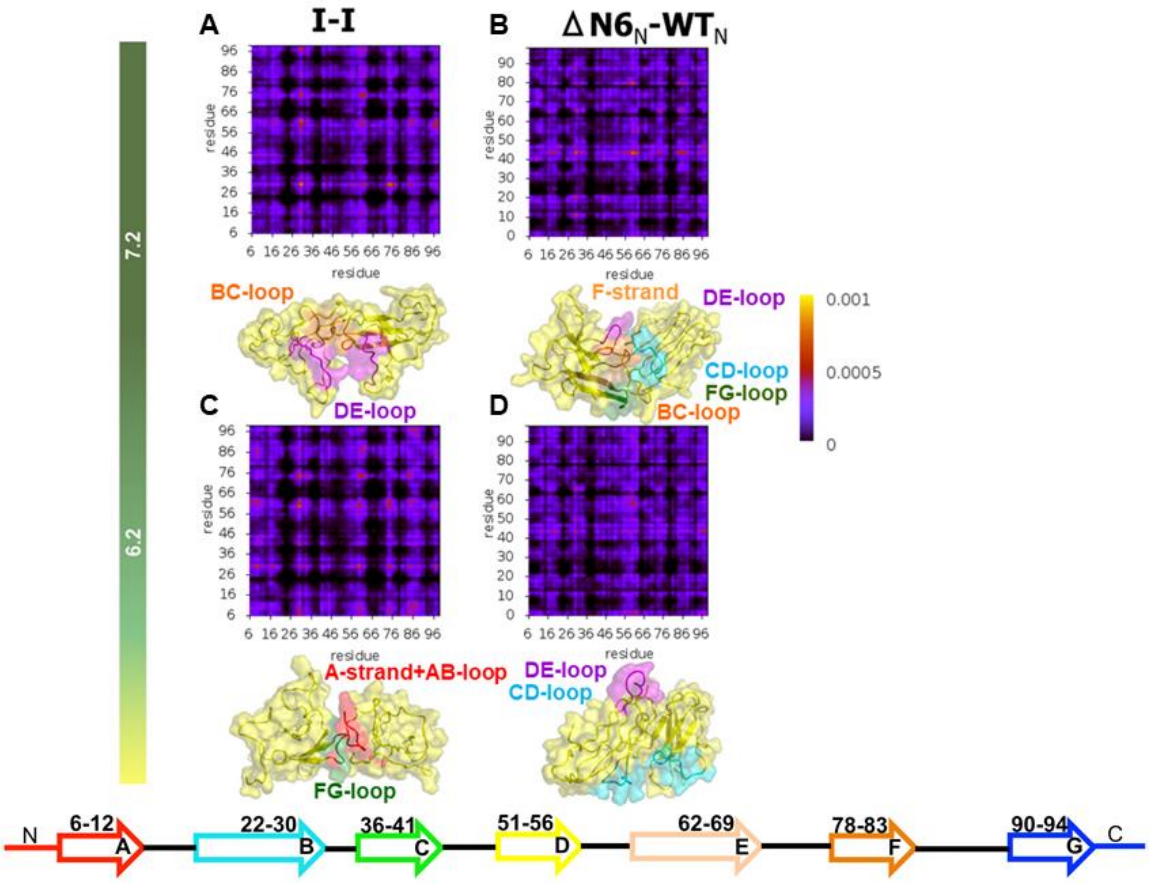


Figure 40. Structural regions involved in ΔN6 dimerization. Probability maps for the intermolecular interactions established between monomers of different ΔN6 dimer structures at pH 7.2 (A-B) and pH 6.2 (C-D).

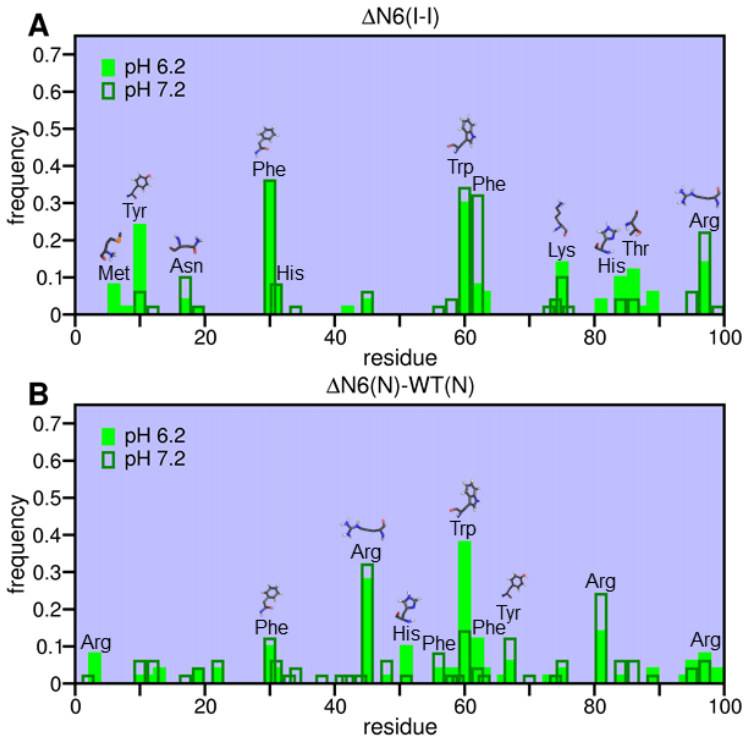


Figure 41. Dimerization hot-spots. Intermolecular interaction probability per residue evaluated in the ensemble of the 50 most frequent intermolecular interactions formed in homodimers of ΔN6 (I) (A) and heterodimers of ΔN6 (N) and WT (N) (B) monomers at pH 6.2 and 7.2.

6.6.4. Integrative analysis on the dimerization hotspots from the two β2-m variants

After studying the dimerization of both β2-m variants as models of β2-m aggregation, we performed a comparative analysis of their dimerization hotspots in order to identify which ones are conserved in the dimerization process (i.e. have a frequency of at least 0.06 in both variants at any of the considered pHs) (Figure 42A and B).

We observe a stronger role of N-terminal residues (Arg3, Tyr10 and Arg12) and a weaker role of BC-loop residues in the interfaces of the D76N mutant dimers relatively to the ΔN6 dimers (Figure 42A and B). In contrast, residue Phe30 of the BC loop is significantly more involved in the formation of intermolecular interactions in ΔN6 dimers than in D76N dimers, particularly at physiological pH (Figure 42B). This is likely the result of the higher displacement and mobility of the BC-loop region in the ΔN6 variant at physiological pH (Table 3, Table 4 and Table 5). The higher relevance of the N-terminal region in the dimerization of the D76N mutant could be rationalized on the basis of the higher detachment of this region from the protein core in the I_2 intermediate of D76N, rendering it more prone to participate in intermolecular interactions with neighboring monomers (Table 4 and Table 5).

The FG-loop is more represented in the interfaces of the dimers of ΔN6 than in the D76N dimers, particularly at the slightly acidic pH 6.2, wherein His84 and Thr86 have some importance in monomer association. Interestingly, Trp60 participates in more intermolecular contacts in the dimers of the D76N intermediates - in which it is the main interaction hub - than it does in the dimers of the ΔN6 intermediate, in which Phe30 is the main interaction hub.

A comparative analysis of the hotspots profiles of the two β2-m variants reveals that Tyr10 (A-strand), Phe30 and His31 (BC-loop), Arg45 (CD-loop), Trp60 and Phe62 (DE-loop), Lys75 (EF-loop), and Trp95 and Arg97 (C-terminus) are the essential residues for β2-m dimerization. Indeed, these residues have a fundamental role in the dimerization of both β2-m variants, used here as model systems of β2-m, and, as such, it is likely that they may be involved in the early phase of the β2-m aggregation mechanism. Interestingly, there is already experimental evidence pointing to the involvement of some of these residues in β2-m dimerization. Indeed, Phe30 and His31 was observed to be part of the interface of a ΔN6 nanobody-trapped domain-swapped dimer (68), while Arg45 was present in the interface of the DCIM50 homodimers (109). Trp60 is widely recognized as a critical residue in β2-m aggregation, either by structural direct evidence (59, 101, 109) or by indirect evidence showing the abrogation of aggregation upon Trp60 mutation. Indeed, several studies with Trp60 mutants revealed an increase in the conformational stability of the protein (81) as well as less propensity to form oligomers (81) and amyloid fibrils (82). Moreover, several studies provided direct evidence of the participation of Trp60 in interfaces of β2-m dimers (68, 101), wherein it mediates hydrophobic interactions important for dimer's stability. Phe62 was also identified as an interacting partner in the interface of the DimC33 covalent homodimer obtained by the mutation of the serine at position 33 to a cysteine (101). Besides corroborating previous experimental findings, we also predicted the involvement of new residues in β2-m dimerization such as Tyr10, Lys75, Trp95 and Arg97, thus providing new testable predictions to guide the research on β2-m amyloidogenesis and therapeutic strategies.

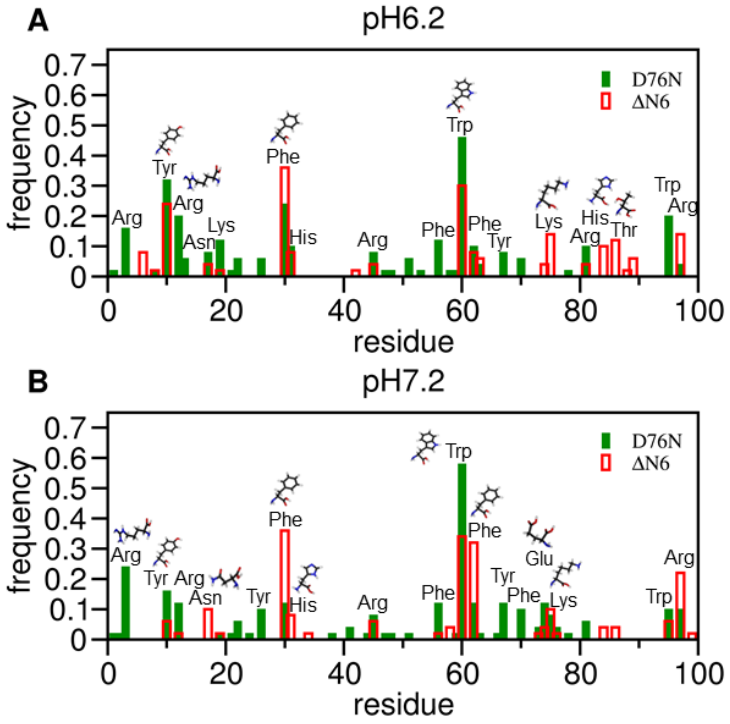


Figure 42. Dimerization hot-spots. Comparison of the intermolecular interaction probability per residue evaluated in the ensemble of the 50 most frequent intermolecular interactions formed in the dimer structures generated by the intermediates of both variants of β2-microglobulin, the ΔN6 structural variant and the D76N mutant, at pH 6.2 (A) and pH 7.2 (B).

6.7. Insights into the tetramerization stage of β2-m aggregation

Here we study the tetramerization phase of D76N at physiological pH and of ΔN6 at a slightly acidic pH. We focus our analysis on the D76N mutant at physiological pH and on the ΔN6 at a slightly acidic pH because these pH conditions are the biologically relevant conditions for the respective variants. Because the simulations (i.e. protein-protein docking and structure relaxation with molecular dynamics) are significantly time consuming, we restricted our analysis to the most aggregation-prone intermediate state of D76N, namely I_2, and to the intermediate I of ΔN6.

As mentioned in the introductory section, there is experimental evidence according to which the aggregation pathway of wt β2-m proceeds exclusively by the formation of even-numbered oligomers (soluble tetramers and hexamers) formed through the addition of dimeric units (113). Assuming that the aggregation pathways of the D76N and ΔN6 conserves this parity, we studied the formation of D76N and ΔN6 tetramers by docking dimers of I_2 and I at pH 7.2 and pH 6.2, respectively. Ensembles of 1000 tetramers were generated. We started by computing the PDF for the binding energy of tetramers formed from the homodimers of the I_2

intermediate, i.e. the most aggregation-prone intermediate of D76N mutant, at physiological pH, and from the homodimers of I at a slightly acidic pH. The PDF for the binding energy indicates that tetramers are significantly less stable (E~-10) than the homodimers of I_2 (E~-19), suggesting that dimers are the most likely dominant species in the initial phase of D76N aggregation (Figure 43A). The same holds for the ΔN6 intermediate, with the tetramers being markedly less stable (E~-5) than the homodimers of the ΔN6 intermediate (E~-19) and, significantly, less stable than the tetramers of the I_2 intermediate of D76N (Figure 44A). This indicates that tetramerization is less favored in the ΔN6 variant than in the D76N mutant, which could rationalize the lower *in vitro* aggregation propensity of ΔN6 in relation to the D76N mutant. We also evaluated the PDF for the number of intermolecular contacts, which shows that the dimer-dimer interfaces exhibit a similar degree of compactness than the monomer-monomer interfaces in the dimers, both with ~4-5K intermolecular atomic contacts, despite having a lower number of atomic clashes (~200).

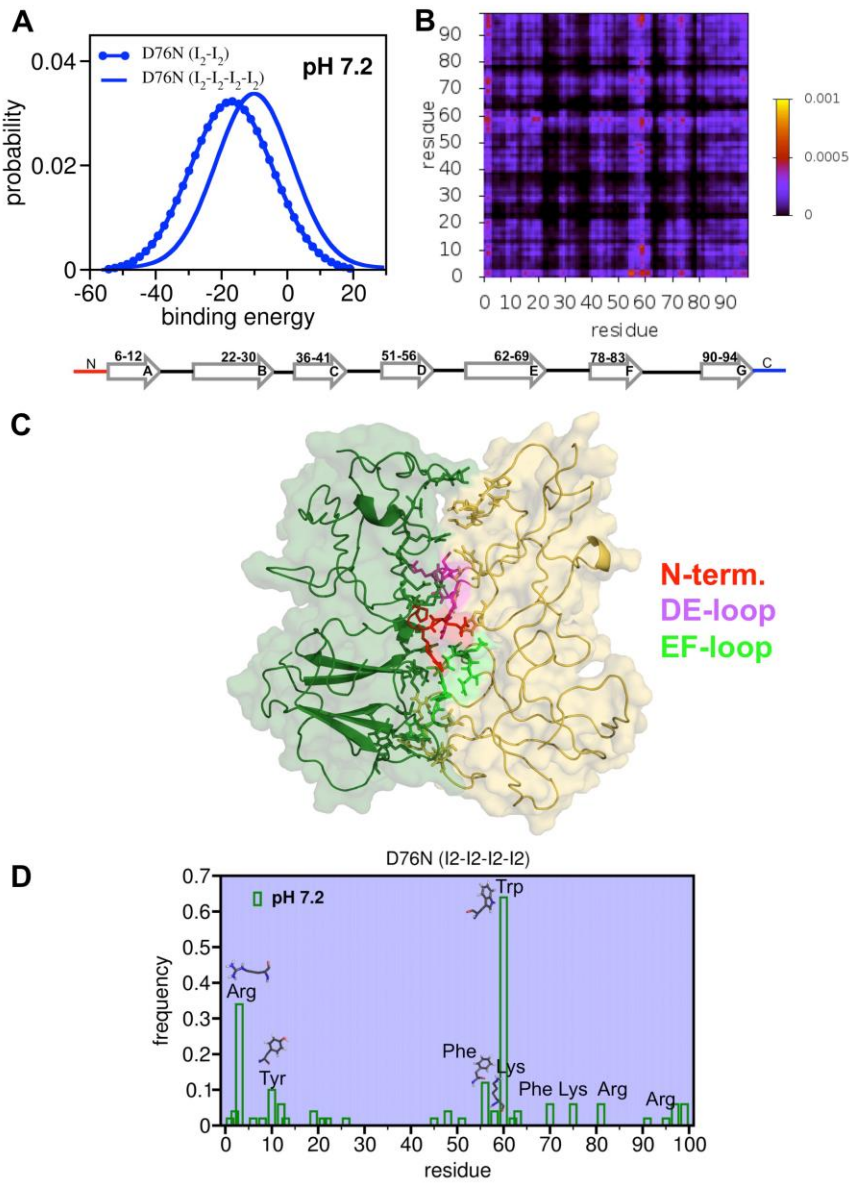


Figure 43. Tetramerization of D76N. (A) PDF for the binding energy of tetramers formed by dimers of I_2; (B) Probability map for intermolecular contacts formed at the interface of tetramers; (C) Representative tetramer conformation in which the dimers are colored green and yellow and the residues that mediate interfacial interactions are represented with sticks; and (D) Tetramerization hot-spots.

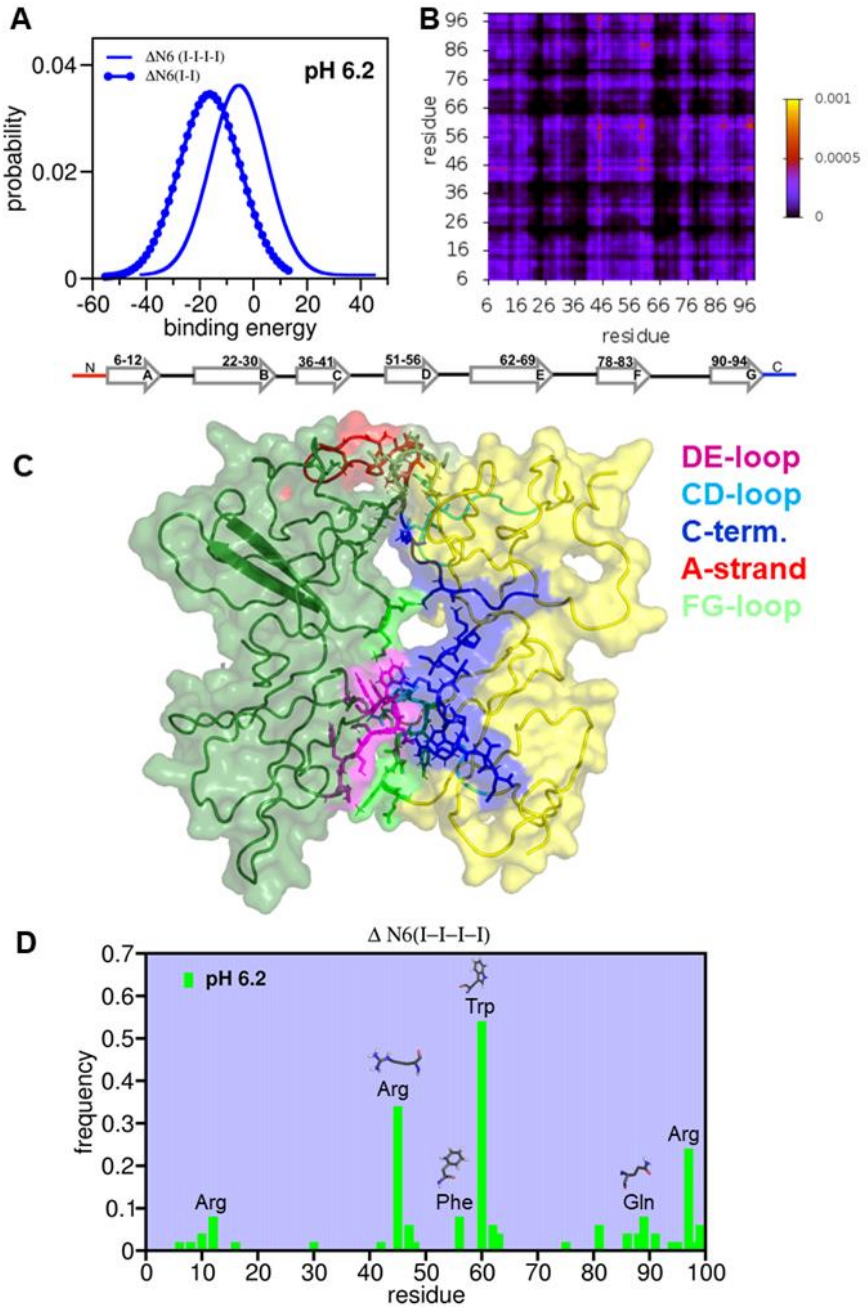


Figure 44. Tetramerization of ΔN6. (A) PDF for the binding energy of tetramers formed by dimers of I; (B) Probability map for intermolecular contacts formed at the interface of tetramers; (C) Representative tetramer conformation in which the dimers are colored green and yellow and the residues that mediate interfacial interactions are represented with sticks; and (D) Tetramerization hot-spots.

The analysis of the IPM (Figure 43B) for the intermolecular contacts of the D76N mutant suggests that the DE-loop together with N-terminal region (N-terminus and A-strand), the EF-loop and the C-terminus are the most important adhesion zones in the D76N tetramer. In the ΔN6 tetramer, the CD-loop, the DE-loop and the C-terminus are the main adhesion

zones (Figure 44B). In line with this observation we find that intermolecular interactions in the D76N tetramer are most likely mediated by Arg3 (N-terminus), Tyr10 (A-strand), Phe56 (D-strand), Trp60 (DE-loop), and, to a lesser extent, by Lys58 (DE-loop) and Arg97 (C-terminus) (Figure 43D). In the ΔN6 tetramers, Trp60 (DE-loop) and Phe56 (D-strand) are also important in establishing intermolecular interactions in tetramer's interface, as well as Arg12 (A-strand), Arg45 (CD-loop), Gln89 (FG-loop) and Arg97 (C-terminus) (Figure 44D). It is likely that intermolecular interactions between the positively charged arginine residues contribute to destabilize the tetramer's interface, rationalizing the higher binding energies of ΔN6 tetramers comparatively to those of D76N mutant. These results corroborate experimental findings that showed the presence of Phe56 and Trp60 in interfaces of β2-m tetramers. The participation of Phe56 and Trp60 in the tetramer's interface is supported by the presence of these residues in the interface of the tetramers formed from the disulfide-linked homodimers DCIM20 and DCIM50 (109). Phe56, as well as Lys58, Gln89 and Arg97, are present in the interface of the wt β2-m tetramer in the presence of Cu^{2+} structurally characterized by Vachet and co-workers using covalent labelling and mass spectrometry in combination with MD (115). Particularly, Lys58 forms a hydrogen bond with Gln89, while Glu50 forms a salt bridge with Arg97. Interestingly, while this study also supports an important role for the D- and G- strands in the establishment of the tetramer's interface, they preclude the participation of the N-terminus. However, this observation may result from the fact that the residues located on the N-terminus remain Cu^{2+} binding sites and therefore steric hindrance precludes their participation in the tetramer's interface. We expected the participation of the N- and/or C-terminal regions in tetramerization of the I_2 and I intermediates at pH 7.2 and pH 6.2, respectively, as most of the homodimers of these intermediates have one or both of them available to further oligomerization, as previously highlighted in sections 6.6.2 and 6.6.3.

6.8. Key findings

In the present study, we used an array of molecular simulation methods to explore the early stage of the aggregation pathway of the D76N mutant of protein β2-m. The exploration of the folding space with DMD simulations predicts the occurrence of two aggregation-prone intermediate states, I_1 and I_2. The intermediate I_1 has a well-preserved core and a C-terminus unstructured and detached from the protein's core, while the I_2 intermediate presents a well-preserved core and both termini unstructured and detached from the protein's core.

We found that I_2 (the intermediate that is exclusively populated by the mutant form) is considerably more prone to aggregate than I_1 (the intermediate which is also populated by the wt) forming homodimers with higher steric complementarity.

The identification of the so-called topology aggregation hot-spots carried out in this study reiterates the importance of residues such as Trp60 and Phe62 and predicts the involvement of new residues such as Lys75 and Trp95 in the aggregation process.

The energy-structure mapping carried out here for the dimers of the I_2 intermediate of D76N at pH 7.2 reveals that the new MC-ED cost function is correctly capturing the main driving forces of protein-protein association as the simulations with each energy contribution generate dimer's interfaces predominantly constituted by residues with physicochemical properties prone to the type of interaction being considered. Furthermore, when using the new cost function, the interfaces included residues with different physicochemical characteristics, reflecting the different kinds of intermolecular interactions included in the cost function.

The analysis of the density curves for the binding energies of the dimers generated from the different conformational states of the two variants indicates that I_2 is, along with the ΔN6 intermediate, the most aggregation-prone intermediate of β2-microglobulin.

The analysis of the IPMs suggests that the DE-loop and the EF-loop are the essential regions in D76N dimerization at neutral pH while the N- and C-terminal regions have a prominent role at acidic conditions. In I_2 homodimers there is a decrease in the relevance of the C-terminus and an increase in the relevance of the DE-loop and EF-loop at acidic pH when compared to the homodimers of I_1 and to the heterodimers. This could be rationalized on the basis that besides having an unstructured C-terminus (like I_1), I_2 has also an unstructured N-terminus, which renders the central DE- and EF-loops more mobile and available to participate in intermolecular interactions at dimer interfaces, which is in agreement with recently reported experimental data. Additionally, the strikingly dominant role of the C-terminus and adjacent unstructured G strand in I_2 dimerization at acidic pH observed with the original MC-ED is substantially suppressed when interactions other than packing are taken into account. This, together with the increased relevance of the N-terminal region in D76N dimerization, likely reflects the role of polar and charged residues of regions such as the N-terminal strand-A in the establishment of electrostatic interactions upon dimerization.

The analysis of the IPMs indicates that the N-terminal A-strand is essential in ΔN6 dimerization, in agreement with its unstructured and detached character. Additionally, there is a more evident fingerprint for the BC-loop with the new version of the MC-ED than in the original based only on steric complementarity, presumably due to the role of electrostatic interactions established by the His31.

A prionlike mechanism appears to be a less efficient pathway for β2-m amyloidogenesis than those mediated by the ΔN6 and D76N intermediates although it is possible that the higher binding energies of these heterodimers are a prerequisite for the conformational conversion mechanism as suggested by an NMR study by Radford and colleagues. Additionally, this NMR study revealed the participation of the DE-loop, BC-loop and FG-loop in the interfaces of the heterodimers of ΔN6 and wt β2-m, regions that we predicted to be involved in the formation of these dimers.

An integrative analysis of the dimerization hotspots from the two β2-m variants reveals that Phe30 and His31 (BC-loop), Arg45 (CD-loop), and Trp60 and Phe62 (DE-loop) are essential residues for β2-microglobulin dimerization. These predictions corroborate experimental results supporting the involvement of the BC-loop and DE-loop, particularly of Phe30, His31, Trp60 and Phe62, in β2-microglobulin dimerization. We also predicted the involvement of new residues in β2-m dimerization such as Tyr10, Lys75, Trp95 and Arg97, thus providing new testable predictions to guide the research on β2-microglobulin amyloidogenesis and therapeutic strategies.

We also get first insights into the tetramerization phase of β2-m aggregation, namely of the I_2 intermediate of D76N mutant and of the I intermediate of ΔN6. We verified that the tetramerization of ΔN6 is less favourable than that of D76N as observed in the PDFs for the tetramer's binding energy, which contributes to explain the lower *in vitro* amyloidogenicity of ΔN6 in relation to D76N. We propose that the N- and C-terminal regions and the DE-loop have an important role in tetramer formation, with residues Trp60 (DE-loop), Arg3 (N-terminus), Phe56 (D-strand), Tyr10 (A-strand), Arg97 (C-terminus), Arg45 (CD-loop) and, to a lesser extent, Gln89 (FG-loop) and Lys58 (DE-loop) mediating interactions in tetramer's interface.

7. Conclusions & Future Work

Solving the aggregation mechanism of protein β2-microglobulin (β2-m) is a task of paramount importance given its role as causative agent of dialysis related amyloidosis (DRA), a conformational disorder that affects more than 90% of people doing long-term hemodialysis worldwide (240). Unfortunately, the wt form does not aggregate *de novo* under *in vitro* physiological conditions, and over the years researchers have been exploring engineered or naturally occurring model systems to gain insight into the fibrillogenesis mechanism of the parent species. This project focused on the truncated mutant ΔN6, whose biological significance is not clear, and on the single point mutant D76N found in one French family, which aggregates in several visceral organs causing a systemic amyloidosis. The results reported here help gain insight into the fibrillogenesis mechanism of the parent species, but they do not entail an exclusive role of the truncated species in the actual fibrillogenesis pathway of the full-length wt protein, nor do they seek to reduce the latter to the aggregation pathway of the D76N mutant. Indeed, it is likely that aggregation of the full length wt form is strictly dependent on unique environmental conditions occurring in the osteoarticular system of dialysis patients, and, therefore, the latter should be identified and mimicked both *in vitro* as well as in simulations in order to draw a more accurate picture of wt β2-m aggregation in DRA. The goal of the present study is to provide mechanistic insights, hypotheses and testable theoretical predictions on the early dimerization phase of two model systems (ΔN6 and D76N) that aggregate *in vitro* under physiological unseeded conditions.

In particular, we focus our analysis on the self-association process of intermediate states for folding with aggregation potential that were identified in simulation studies (57, 116) framed on structure-based models for protein folding, i.e. that highlight the topological features of this self-assembly process. A distinctive structural trait of these intermediate states is the existence of one (in the intermediate I of ΔN6 and in the intermediate I_1 of D76N) or two (in the I_2 intermediate of D76N) unstructured terminal regions. The importance of unstructured terminal regions in the aggregation mechanism of β2-m (33, 34, 59, 61, 73, 82, 91, 241-243) and in the aggregation of other model systems (244-251) has been acknowledged in several studies.

This study presents a novelty in relation to our previous contributions (57, 220) in one fundamental point: the docking procedure we deploy to explore the dimerization phase of β2-m uses a cost function that extends beyond packing interactions, which account for shape

complementarity, by also including electrostatic and hydrogen bond interactions, as well as interactions between hydrophobic atoms that seek to modulate the hydrophobic effect.

While the original version of the Monte Carlo ensemble docking (MC-ED) algorithm predicted a direct role (i.e. via the establishment of intermolecular contacts) of the unstructured terminal regions of all intermediate states in triggering aggregation, the novel version indicates that the unfolding and detachment of the terminal regions from the core may increase the mobility and solvent exposure of other structural elements that now appear as sticky regions, namely the DE-loop (in the dimerization of both β2-m variants), the EF-loop (in the dimerization of the D76N mutant) and the BC-loop (in the dimerization of the ΔN6 variant). In particular, the new cost function highlights a clearly more important role for the DE-loop and the EF-loop in the dimerization of the I_2 intermediate (D76N) at pH 5.2. Overall, the DE-, EF- and BC-loops dominate at physiological pH and the terminal regions at acidic pH. Interestingly, the strikingly dominant role of the C-terminus and adjacent unstructured G strand in the dimerization of I_2 at acidic pH that was previously observed is substantially suppressed when electrostatic interactions (e.g. those involving polar and charged residues of the N-terminus and A-strand) are also included in the cost function. The novel cost function indicates a relevant role for the A-strand in the dimerization of ΔN6 at the slightly acidic pH 6.2 and a clear fingerprint for the BC-loop, which was not so visible with the original version of the method, most likely due to the electrostatic interactions established by the His31.

Here, we also analysed for the first time the interfaces resulting from intermolecular interactions between the native state of ΔN6 and the native state of wt β2-m, which would underlie a "prion-like" mechanism for β2-m amyloidogenesis. The dimers we obtained are the most unstable of all dimers studied here, featuring relatively high binding energies in agreement with experimental data reported by Radford and co-workers (239). Our results support the involvement of the DE-loop, BC-loop and FG-loop in the interfaces of the heterodimers of ΔN6 and wt β2-m, also in line with experimental data (239).

The results of extensive simulations carried out in the present study are in line with experimental data supporting an essential role for Phe30 and His31 (BC-loop), Arg45 (CD-loop), and Trp60 and Phe62 (DE-loop) in β2-m dimerization. Additionally, they predict novel hotspot residues such as Tyr10 (A-strand), Lys75 (EF-loop), and Trp95 and Arg97 (C-terminus).

Finally, by studying the dimerization of dimers of the intermediate I_2 populated by the D76N mutant and of the intermediate I of ΔN6, we obtained first glimpses into the tetramerization interface of β2-m. We verified that the tetramerization of ΔN6 is less favourable than that of D76N as observed in the PDFs for the tetramer's binding energy, which contributes to explain the lower *in vitro* amyloidogenicity of ΔN6 in relation to D76N. We predict that the N- and C-terminal regions and the DE-loop have an important role in the structure of the interface of the tetramer, and we propose that the formation of the latter may be mediated by interactions involving Trp60 (DE-loop), Arg3 (N-terminus), Phe56 (D-strand), Tyr 10 (A-strand), Arg97 (C-terminus), Arg45 (CD-loop) and, to a lesser extent, Gln89 (FG-loop) and Lys58 (DE-loop).

An outstanding question in amyloid disease concerns the mechanism(s) of cytotoxicity. The classical amyloid hypothesis, according to which the toxic species is the amyloid fibril itself, is gradually evolving into the view that the oligomers produced along the amyloid cascade are the primary toxic species while fibrils may be toxic, inert or even protective (17). This assumption rests on growing evidence that pre-fibrilar oligomers have the potential to disrupt the permeability of cellular membranes (through the formation of ion channels, pores or non-selective permeation of lipid bilayers), eventually causing cell death (15). Thus, conformational states that will not evolve into amyloids may actually play a critical role in amyloid disease due to their cytotoxic effects. While researchers are still taking the first steps towards understanding the mechanism of toxicity of oligomers it is becoming widely accepted that membrane disruption and permeabilization is likely the major cause of cell impairment and death. However, difficulty in obtaining highly pure samples of non-fibrillar aggregates that are sufficiently long-lived for biophysical studies has significantly hindered progress in the field. As such computational simulations may be a useful alternative to address this challenge.

A recent study by Chiti and co-workers on protein HypF-N (252), which is not associated with amyloid disease, indicate that toxicity and the ability to trigger apoptosis are only associated with the less compact, less stable and more hydrophobic assemblies. Inspired by these results, we will explore the relation between the structural properties of oligomers and the amount of physical disruption (i.e. the introduction of membrane defects) resulting from membrane-oligomers interactions, thus providing a seminal contribution to the problem of cytotoxicity in β2-m amyloidosis with potential impact on therapeutic strategies. To achieve this goal, we will use classical MD simulations to evaluate the cytotoxic potential of

selected dimer and tetramer conformations by studying the effects of their interaction with lipid bilayers. We will also investigate putative conformational changes occurring in the dimers and tetramers resulting from interactions with the membrane. We will be mostly interested in exploring in detail the relation between the structural properties of oligomers and the amount of physical disruption (i.e. the introduction of membrane defects) due to membrane-bound (and membrane-inserted) oligomers. Physical disruption will induce changes in membrane permeation resulting in leakage of cell contents, eventually leading to cell death. We will evaluate the structural changes induced by specific oligomeric states in model membranes. Since these simulations are highly demanding one needs to carefully select the model dimers and tetramers that will be investigated. With the experimentally invoked hypothesis that the more hydrophobic, less stable dimers grow into mature fibrils whereas the less hydrophobic, more stable dimers seem to aggregate into stable protofibrils with no further evolution (253), we will pre-select the latter as model probes. Furthermore, since we are interested in structures that are able to sustain mechanical strain (as a result of their insertion within membranes), and energetic stability does not necessarily imply mechanic stability, we will place an additional selection criteria: besides being the less hydrophobic and the more energetically stable structures, the selected oligomers should be also the most mechanically stable structures in our ensembles. Thus, we will further scrutinize the pre-selected oligomers with steered MD (254) simulations mimicking pulling experiments with atomic force microscopy. For each protein variant the selected dimer (and tetramer) will be tested for cytotoxicity. This amounts to place the oligomer in contact with the membrane. Both the unrestrained and the steered atomistic MD simulations will be performed using the GROMACS software package. We will use the GROMOS 54A7 force field, an excellent protein force field that has now a very good phosphatidylcholine lipid parameterization (255).

Bibliography

1. Jackson SE. How do small single-domain proteins fold? Folding & design. 1998;3(4):R81-91. Epub 1998/08/26. doi: 10.1016/S1359-0278(98)00033-9. PubMed PMID: 9710577.
2. Neudecker P, Robustelli P, Cavalli A, Walsh P, Lundstrom P, Zarrine-Afsar A, et al. Structure of an intermediate state in protein folding and aggregation. Science. 2012;336(6079):362-6. Epub 2012/04/21. doi: 10.1126/science.1214203. PubMed PMID: 22517863.
3. Chiti F, Dobson CM. Protein misfolding, functional amyloid, and human disease. Annual review of biochemistry. 2006;75:333-66. Epub 2006/06/08. doi: 10.1146/annurev.biochem.75.101304.123901. PubMed PMID: 16756495.
4. Gomes CM, Faísca PFN. Protein Folding: An Introduction: Springer International Publishing; 2019.
5. Harris J, Milton, NGM. Introduction and Technical Survey: Protein Aggregation and Fibrillogenesis. In: Harris J, editor. Protein Aggregation and Fibrillogenesis in Cerebral and Systemic Amyloid Disease. Dordrecht: Springer Netherlands; 2012. p. 3-25.
6. Sipe JD, Benson MD, Buxbaum JN, Ikeda S, Merlini G, Saraiva MJ, et al. Amyloid fibril protein nomenclature: 2010 recommendations from the nomenclature committee of the International Society of Amyloidosis. Amyloid. 2010;17(3-4):101-4. Epub 2010/11/03. doi: 10.3109/13506129.2010.526812. PubMed PMID: 21039326.
7. Uversky VN, Fink A. Protein Misfolding, Aggregation and Conformational Diseases: Part A: Protein Aggregation and Conformational Diseases: Springer US; 2006.
8. Cohen SI, Vendruscolo M, Dobson CM, Knowles TP. From macroscopic measurements to microscopic mechanisms of protein aggregation. Journal of molecular biology. 2012;421(2-3):160-71. Epub 2012/03/13. doi: 10.1016/j.jmb.2012.02.031. PubMed PMID: 22406275.
9. Morriss-Andrews A, Shea JE. Computational studies of protein aggregation: methods and applications. Annual review of physical chemistry. 2015;66:643-66. Epub 2015/02/05. doi: 10.1146/annurev-physchem-040513-103738. PubMed PMID: 25648485.
10. Thompson AJ, Barrow CJ. Protein conformational misfolding and amyloid formation: characteristics of a new class of disorders that include Alzheimer's and Prion diseases. Current medicinal chemistry. 2002;9(19):1751-62. Epub 2002/10/09. PubMed PMID: 12369885.
11. Moreno-Gonzalez I, Soto C. Misfolded protein aggregates: mechanisms, structures and potential for disease transmission. Seminars in cell & developmental biology. 2011;22(5):482-7. Epub 2011/05/17. doi: 10.1016/j.semcdb.2011.04.002. PubMed PMID: 21571086; PubMed Central PMCID: PMC3175247.
12. Corlin BD HH. β2-Microglobulin Amyloidosis. In: Harris R, editor. Protein Aggregation and Fibrillogenesis in Cerebral and Systemic Amyloid Disease. Dordrecht: Springer Netherlands; 2012. p. 517-40.
13. Kiss E, Keusch G, Zanetti M, Jung T, Schwarz A, Schocke M, et al. Dialysis-related amyloidosis revisited. AJR American journal of roentgenology. 2005;185(6):1460-7. Epub 2005/11/24. doi: 10.2214/AJR.04.1309. PubMed PMID: 16303998.
14. Herrup K. The case for rejecting the amyloid cascade hypothesis. Nature neuroscience. 2015;18(6):794-9. Epub 2015/05/27. doi: 10.1038/nn.4017. PubMed PMID: 26007212.
15. Bucciantini M, Rigacci S, Stefani M. Amyloid Aggregation: Role of Biological Membranes and the Aggregate-Membrane System. The journal of physical chemistry letters. 2014;5(3):517-27. Epub 2014/02/06. doi: 10.1021/jz4024354. PubMed PMID: 26276603.
16. Leal SS, Botelho HM, Gomes CM. Metal ions as modulators of protein conformation and misfolding in neurodegeneration. Coordin Chem Rev. 2012;256(19-20):2253-70. doi: 10.1016/j.ccr.2012.04.004. PubMed PMID: ISI:000308837100013.

17. Stefani M. Structural features and cytotoxicity of amyloid oligomers: Implications in Alzheimer's disease and other diseases with amyloid deposits. Prog Neurobiol. 2012;99(3):226-45. doi: 10.1016/j.pneurobio.2012.03.002. PubMed PMID: ISI:000312172400004.
18. Hartl FU, Bracher A, Hayer-Hartl M. Molecular chaperones in protein folding and proteostasis. Nature. 2011;475(7356):324-32. Epub 2011/07/22. doi: 10.1038/nature10317. PubMed PMID: 21776078.
19. Esposito G. CA, Bellotti V. et al. Pathological Self-Aggregation of b2-Microglobulin: A Challenge for Protein Biophysics. 2012. In: Protein Aggregation and Fibrillogenesis in Cerebral and Systemic Amyloid Disease [Internet]. Springer NetherlandsSubcellular Biochemistry.
20. Eichner T, Radford SE. Understanding the complex mechanisms of beta2-microglobulin amyloid assembly. The FEBS journal. 2011;278(20):3868-83. Epub 2011/05/21. doi: 10.1111/j.1742-4658.2011.08186.x. PubMed PMID: 21595827; PubMed Central PMCID: PMC3229708.
21. Becker JW, Reeke GN, Jr. Three-dimensional structure of beta 2-microglobulin. Proceedings of the National Academy of Sciences of the United States of America. 1985;82(12):4225-9. Epub 1985/06/01. PubMed PMID: 3889925; PubMed Central PMCID: PMC397969.
22. Smith DP, Radford SE. Role of the single disulphide bond of beta(2)-microglobulin in amyloidosis in vitro. Protein science : a publication of the Protein Society. 2001;10(9):1775-84. Epub 2001/08/22. doi: 10.1110/ps.4901. PubMed PMID: 11514668; PubMed Central PMCID: PMC2253195.
23. Yamamoto K, Yagi H, Ozawa D, Sasahara K, Naiki H, Goto Y. Thiol compounds inhibit the formation of amyloid fibrils by beta 2-microglobulin at neutral pH. Journal of molecular biology. 2008;376(1):258-68. Epub 2007/12/25. doi: 10.1016/j.jmb.2007.12.002. PubMed PMID: 18155723.
24. Hasegawa K, Ohhashi Y, Yamaguchi I, Takahashi N, Tsutsumi S, Goto Y, et al. Amyloidogenic synthetic peptides of beta2-microglobulin--a role of the disulfide bond. Biochemical and biophysical research communications. 2003;304(1):101-6. Epub 2003/04/23. PubMed PMID: 12705891.
25. Kameda A, Hoshino M, Higurashi T, Takahashi S, Naiki H, Goto Y. Nuclear magnetic resonance characterization of the refolding intermediate of beta2-microglobulin trapped by non-native prolyl peptide bond. Journal of molecular biology. 2005;348(2):383-97. Epub 2005/04/07. doi: 10.1016/j.jmb.2005.02.050. PubMed PMID: 15811375.
26. Bellotti V, Gallieni M, Giorgetti S, Brancaccio D. Dynamic of beta(2)-microglobulin fibril formation and reabsorption: the role of proteolysis. Seminars in dialysis. 2001;14(2):117-22. Epub 2001/03/27. PubMed PMID: 11264779.
27. Gejyo F, Yamada T, Odani S, Nakagawa Y, Arakawa M, Kunitomo T, et al. A new form of amyloid protein associated with chronic hemodialysis was identified as beta 2-microglobulin. Biochemical and biophysical research communications. 1985;129(3):701-6. Epub 1985/06/28. PubMed PMID: 3893430.
28. Gejyo F, Odani S, Yamada T, Honma N, Saito H, Suzuki Y, et al. Beta 2-microglobulin: a new form of amyloid protein associated with chronic hemodialysis. Kidney international. 1986;30(3):385-90. Epub 1986/09/01. PubMed PMID: 3537446.
29. Floege J, Ketteler M. beta2-microglobulin-derived amyloidosis: an update. Kidney international Supplement. 2001;78:S164-71. Epub 2001/02/13. doi: 10.1046/j.1523-1755.2001.59780164.x. PubMed PMID: 11169004.
30. Giorgetti S, Rossi A, Mangione P, Raimondi S, Marini S, Stoppini M, et al. Beta2-microglobulin isoforms display an heterogeneous affinity for type I collagen. Protein science : a publication of the Protein Society. 2005;14(3):696-702. Epub 2005/02/04. doi: 10.1110/ps.041194005. PubMed PMID: 15689502; PubMed Central PMCID: PMC2279294.
31. van Ypersele de Strihou C, Jadoul M, Malghem J, Maldague B, Jamart J. Effect of dialysis membrane and patient's age on signs of dialysis-related amyloidosis. The Working Party on Dialysis Amyloidosis. Kidney international. 1991;39(5):1012-9. Epub 1991/05/01. PubMed PMID: 2067196.
32. Zingraff JJ, Noel LH, Bardin T, Atienza C, Zins B, Drueke TB, et al. Beta 2-microglobulin amyloidosis in chronic renal failure. The New England journal of medicine. 1990;323(15):1070-1. Epub 1990/10/11. doi: 10.1056/NEJM199010113231514. PubMed PMID: 2215569.

33. Corazza A, Pettirossi F, Viglino P, Verdone G, Garcia J, Dumy P, et al. Properties of some variants of human beta2-microglobulin and amyloidogenesis. The Journal of biological chemistry. 2004;279(10):9176-89. Epub 2003/12/09. doi: 10.1074/jbc.M310779200. PubMed PMID: 14660575.
34. Morgan CJ, Gelfand M, Atreya C, Miranker AD. Kidney dialysis-associated amyloidosis: a molecular role for copper in fiber formation. Journal of molecular biology. 2001;309(2):339-45. Epub 2001/05/24. doi: 10.1006/jmbi.2001.4661. PubMed PMID: 11371157.
35. Piazza R, Pierno M, Iacopini S, Mangione P, Esposito G, Bellotti V. Micro-heterogeneity and aggregation in beta2-microglobulin solutions: effects of temperature, pH, and conformational variant addition. European biophysics journal : EBJ. 2006;35(5):439-45. Epub 2006/03/08. doi: 10.1007/s00249-006-0051-0. PubMed PMID: 16520957.
36. Relini A, De Stefano S, Torrassa S, Cavalleri O, Rolandi R, Gliozzi A, et al. Heparin strongly enhances the formation of beta2-microglobulin amyloid fibrils in the presence of type I collagen. The Journal of biological chemistry. 2008;283(8):4912-20. Epub 2007/12/07. doi: 10.1074/jbc.M702712200. PubMed PMID: 18056266.
37. Pal-Gabor H, Gombos L, Micsonai A, Kovacs E, Petrik E, Kovacs J, et al. Mechanism of lysophosphatidic acid-induced amyloid fibril formation of beta(2)-microglobulin in vitro under physiological conditions. Biochemistry. 2009;48(24):5689-99. Epub 2009/05/13. doi: 10.1021/bi900356r. PubMed PMID: 19432419.
38. Hasegawa K, Tsutsumi-Yasuhara S, Ookoshi T, Ohhashi Y, Kimura H, Takahashi N, et al. Growth of beta(2)-microglobulin-related amyloid fibrils by non-esterified fatty acids at a neutral pH. The Biochemical journal. 2008;416(2):307-15. Epub 2008/07/22. doi: 10.1042/BJ20080543. PubMed PMID: 18637792.
39. Relini A, Canale C, De Stefano S, Rolandi R, Giorgetti S, Stoppini M, et al. Collagen plays an active role in the aggregation of beta2-microglobulin under physiopathological conditions of dialysis-related amyloidosis. The Journal of biological chemistry. 2006;281(24):16521-9. Epub 2006/04/08. doi: 10.1074/jbc.M513827200. PubMed PMID: 16601119.
40. Valleix S, Gillmore JD, Bridoux F, Mangione PP, Dogan A, Nedelec B, et al. Hereditary systemic amyloidosis due to Asp76Asn variant beta2-microglobulin. The New England journal of medicine. 2012;366(24):2276-83. Epub 2012/06/15. doi: 10.1056/NEJMoa1201356. PubMed PMID: 22693999.
41. Stoppini M, Bellotti V. Systemic amyloidosis: lessons from beta2-microglobulin. The Journal of biological chemistry. 2015;290(16):9951-8. Epub 2015/03/10. doi: 10.1074/jbc.R115.639799. PubMed PMID: 25750126; PubMed Central PMCID: PMC4400370.
42. Kelly JW. The alternative conformations of amyloidogenic proteins and their multi-step assembly pathways. Current opinion in structural biology. 1998;8(1):101-6. doi: Doi 10.1016/S0959-440x(98)80016-X. PubMed PMID: ISI:000072384400014.
43. Chiti F, Mangione P, Andreola A, Giorgetti S, Stefani M, Dobson CM, et al. Detection of two partially structured species in the folding process of the amyloidogenic protein beta 2-microglobulin. Journal of molecular biology. 2001;307(1):379-91. Epub 2001/03/13. doi: 10.1006/jmbi.2000.4478. PubMed PMID: 11243826.
44. Chiti F, De Lorenzi E, Grossi S, Mangione P, Giorgetti S, Caccialanza G, et al. A partially structured species of beta 2-microglobulin is significantly populated under physiological conditions and involved in fibrillogenesis. The Journal of biological chemistry. 2001;276(50):46714-21. Epub 2001/10/09. doi: 10.1074/jbc.M107040200. PubMed PMID: 11590172.
45. Jahn TR, Radford SE. Folding versus aggregation: polypeptide conformations on competing pathways. Archives of biochemistry and biophysics. 2008;469(1):100-17. Epub 2007/06/26. doi: 10.1016/j.abb.2007.05.015. PubMed PMID: 17588526; PubMed Central PMCID: PMC2706318.
46. Jahn TR, Parker MJ, Homans SW, Radford SE. Amyloid formation under physiological conditions proceeds via a native-like folding intermediate. Nature structural & molecular biology. 2006;13(3):195-201. Epub 2006/02/24. doi: 10.1038/nsmb1058. PubMed PMID: 16491092.
47. Thirumalai D, Klimov DK, Dima RI. Emerging ideas on the molecular basis of protein and peptide aggregation. Current opinion in structural biology. 2003;13(2):146-59. Epub 2003/05/03. PubMed PMID: 12727507.

48. Rennella E, Corazza A, Giorgetti S, Fogolari F, Viglino P, Porcari R, et al. Folding and fibrillogenesis: clues from beta2-microglobulin. Journal of molecular biology. 2010;401(2):286-97. Epub 2010/06/19. doi: 10.1016/j.jmb.2010.06.016. PubMed PMID: 20558175.
49. Armen RS, Daggett V. Characterization of two distinct beta2-microglobulin unfolding intermediates that may lead to amyloid fibrils of different morphology. Biochemistry. 2005;44(49):16098-107. Epub 2005/12/08. doi: 10.1021/bi050731h. PubMed PMID: 16331970.
50. Liu K, Cho HS, Lashuel HA, Kelly JW, Wemmer DE. A glimpse of a possible amyloidogenic intermediate of transthyretin. Nature structural biology. 2000;7(9):754-7. Epub 2000/08/31. doi: 10.1038/78980. PubMed PMID: 10966644.
51. Rennella E, Cutuil T, Schanda P, Ayala I, Gabel F, Forge V, et al. Oligomeric states along the folding pathways of beta2-microglobulin: kinetics, thermodynamics, and structure. Journal of molecular biology. 2013;425(15):2722-36. Epub 2013/05/08. doi: 10.1016/j.jmb.2013.04.028. PubMed PMID: 23648836.
52. Armen RS, DeMarco ML, Alonso DOV, Daggett V. Pauling and Corey's alpha-pleated sheet structure may define the prefibrillar amyloidogenic intermediate in amyloid disease. Proceedings of the National Academy of Sciences of the United States of America. 2004;101(32):11622-7. doi: 10.1073/pnas.0401781101. PubMed PMID: ISI:000223276700019.
53. Chong SH, Hong J, Lim S, Cho S, Lee J, Ham S. Structural and Thermodynamic Characteristics of Amyloidogenic Intermediates of beta-2-Microglobulin. Scientific reports. 2015;5:13631. Epub 2015/09/09. doi: 10.1038/srep13631. PubMed PMID: 26348154; PubMed Central PMCID: PMC4562173.
54. Fogolari F, Corazza A, Varini N, Rotter M, Gumral D, Codutti L, et al. Molecular dynamics simulation of beta(2)-microglobulin in denaturing and stabilizing conditions. Proteins. 2011;79(3):986-1001. Epub 2011/02/03. doi: 10.1002/prot.22940. PubMed PMID: 21287627.
55. Torbeev V, Ebert MO, Dolenc J, Hilvert D. Substitution of proline32 by alpha-methylproline preorganizes beta2-microglobulin for oligomerization but not for aggregation into amyloids. Journal of the American Chemical Society. 2015;137(7):2524-35. Epub 2015/01/31. doi: 10.1021/ja510109p. PubMed PMID: 25633201.
56. Estacio SG, Shakhnovich EI, Faisca PF. Assessing the effect of loop mutations in the folding space of beta2-microglobulin with molecular dynamics simulations. International journal of molecular sciences. 2013;14(9):17256-78. Epub 2013/08/27. doi: 10.3390/ijms140917256. PubMed PMID: 23975166; PubMed Central PMCID: PMC3794727.
57. Estacio SG, Krobath H, Vila-Vicosa D, Machuqueiro M, Shakhnovich EI, Faisca PF. A simulated intermediate state for folding and aggregation provides insights into DeltaN6 beta2-microglobulin amyloidogenic behavior. PLoS computational biology. 2014;10(5):e1003606. Epub 2014/05/09. doi: 10.1371/journal.pcbi.1003606. PubMed PMID: 24809460; PubMed Central PMCID: PMC4014404.
58. Eakin CM, Berman AJ, Miranker AD. A native to amyloidogenic transition regulated by a backbone trigger. Nature structural & molecular biology. 2006;13(3):202-8. Epub 2006/02/24. doi: 10.1038/nsmb1068. PubMed PMID: 16491088.
59. Calabrese MF, Eakin CM, Wang JM, Miranker AD. A regulatable switch mediates self-association in an immunoglobulin fold. Nature structural & molecular biology. 2008;15(9):965-71. Epub 2009/01/28. PubMed PMID: 19172750; PubMed Central PMCID: PMC2680708.
60. Corazza A, Rennella E, Schanda P, Mimmi MC, Cutuil T, Raimondi S, et al. Native-unlike long-lived intermediates along the folding pathway of the amyloidogenic protein beta2-microglobulin revealed by real-time two-dimensional NMR. The Journal of biological chemistry. 2010;285(8):5827-35. Epub 2009/12/24. doi: 10.1074/jbc.M109.061168. PubMed PMID: 20028983; PubMed Central PMCID: PMC2820808.
61. Mukaiyama A, Nakamura T, Makabe K, Maki K, Goto Y, Kuwajima K. The molten globule of beta(2)-microglobulin accumulated at pH 4 and its role in protein folding. Journal of molecular biology. 2013;425(2):273-91. Epub 2012/11/17. doi: 10.1016/j.jmb.2012.11.002. PubMed PMID: 23154171.

62. Mukaiyama A, Nakamura T, Makabe K, Maki K, Goto Y, Kuwajima K. Native-state heterogeneity of beta(2)-microglobulin as revealed by kinetic folding and real-time NMR experiments. Journal of molecular biology. 2013;425(2):257-72. Epub 2012/11/17. doi: 10.1016/j.jmb.2012.11.004. PubMed PMID: 23154167.
63. Ma B, Nussinov R. Molecular dynamics simulations of the unfolding of beta(2)-microglobulin and its variants. Protein engineering. 2003;16(8):561-75. Epub 2003/09/12. PubMed PMID: 12968074.
64. Fang P-S, Zhao J-H, Liu H-L, Liu K-T, Chen J-T, Tsai W-B, et al. Molecular dynamics simulations to investigate the relationship between the structural stability and amyloidogenesis of the wild-type and N-terminal hexapeptide deletion ΔN6 β2-microglobulin. Molecular Simulation. 2009;35(9):755-65. doi: 10.1080/08927020902818005.
65. Hall Z, Schmidt C, Politis A. Uncovering the Early Assembly Mechanism for Amyloidogenic beta2-Microglobulin Using Cross-linking and Native Mass Spectrometry. The Journal of biological chemistry. 2016;291(9):4626-37. Epub 2015/12/15. doi: 10.1074/jbc.M115.691063. PubMed PMID: 26655720; PubMed Central PMCID: PMC4813486.
66. Esposito G, Michelutti R, Verdone G, Viglino P, Hernandez H, Robinson CV, et al. Removal of the N-terminal hexapeptide from human beta2-microglobulin facilitates protein aggregation and fibril formation. Protein science : a publication of the Protein Society. 2000;9(5):831-45. Epub 2000/06/13. doi: 10.1110/ps.9.5.831. PubMed PMID: 10850793; PubMed Central PMCID: PMC2144642.
67. Eichner T, Kalverda, A.P., Thompson, G.S., Homans, S.W., Radford, S.E. Conformational conversion during amyloid formation at atomic resolution. Mol Cell. 2011;41(2):161-72. Epub 2011/01/21. doi: 10.1016/j.molcel.2010.11.028. PubMed Central PMCID: PMCPMC3029554.
68. Domanska K, Vanderhaegen S, Srinivasan V, Pardon E, Dupeux F, Marquez JA, et al. Atomic structure of a nanobody-trapped domain-swapped dimer of an amyloidogenic beta2-microglobulin variant. Proceedings of the National Academy of Sciences of the United States of America. 2011;108(4):1314-9. Epub 2011/01/12. doi: 10.1073/pnas.1008560108. PubMed PMID: 21220305; PubMed Central PMCID: PMC3029709.
69. Heegaard NH, Jorgensen TJ, Cheng L, Schou C, Nissen MH, Trapp O. Interconverting conformations of variants of the human amyloidogenic protein beta2-microglobulin quantitatively characterized by dynamic capillary electrophoresis and computer simulation. Analytical chemistry. 2006;78(11):3667-73. Epub 2006/06/02. doi: 10.1021/ac060194m. PubMed PMID: 16737222.
70. Heegaard NH, Jorgensen TJ, Rozlosnik N, Corlin DB, Pedersen JS, Tempesta AG, et al. Unfolding, aggregation, and seeded amyloid formation of lysine-58-cleaved beta 2-microglobulin. Biochemistry. 2005;44(11):4397-407. Epub 2005/03/16. doi: 10.1021/bi047594t. PubMed PMID: 15766269.
71. Corlin DB, Sen JW, Ladefoged S, Lund GB, Nissen MH, Heegaard NH. Quantification of cleaved beta2-microglobulin in serum from patients undergoing chronic hemodialysis. Clinical chemistry. 2005;51(7):1177-84. Epub 2005/05/14. doi: 10.1373/clinchem.2005.049544. PubMed PMID: 15890888.
72. Heegaard NH, Roepstorff P, Melberg SG, Nissen MH. Cleaved beta 2-microglobulin partially attains a conformation that has amyloidogenic features. The Journal of biological chemistry. 2002;277(13):11184-9. Epub 2002/01/22. doi: 10.1074/jbc.M108837200. PubMed PMID: 11801591.
73. Le Marchand T, de Rosa M, Salvi N, Sala BM, Andreas LB, Barbet-Massin E, et al. Conformational dynamics in crystals reveal the molecular bases for D76N beta-2 microglobulin aggregation propensity. Nature communications. 2018;9(1):1658. Epub 2018/04/27. doi: 10.1038/s41467-018-04078-y. PubMed PMID: 29695721; PubMed Central PMCID: PMC5916882.
74. de Rosa M BA, Giorgetti S, Mangione PP, Bolognesi M, Ricagno S. Decoding the Structural Bases of D76N ß2-Microglobulin High Amyloidogenicity through Crystallography and Asn-Scan Mutagenesis. PloS one. 2015;10(12):e0144061. doi: 10.1371/journal.pone.0144061.
75. Leri M, Bemporad F, Oropesa-Nunez R, Canale C, Calamai M, Nosi D, et al. Molecular insights into cell toxicity of a novel familial amyloidogenic variant of beta2-microglobulin. Journal of cellular

and molecular medicine. 2016;20(8):1443-56. Epub 2016/03/19. doi: 10.1111/jcmm.12833. PubMed PMID: 26990223; PubMed Central PMCID: PMC4956941.
76. Chandrasekaran P, Rajasekaran R. A systematic molecular dynamics approach to the structural characterization of amyloid aggregation propensity of beta2-microglobulin mutant D76N. Molecular bioSystems. 2016;12(3):850-9. Epub 2016/01/14. doi: 10.1039/c5mb00759c. PubMed PMID: 26757617.
77. Camilloni C, Sala BM, Sormanni P, Porcari R, Corazza A, De Rosa M, et al. Rational design of mutations that change the aggregation rate of a protein while maintaining its native structure and stability. Scientific reports. 2016;6:25559. Epub 2016/05/07. doi: 10.1038/srep25559. PubMed PMID: 27150430; PubMed Central PMCID: PMC4858664.
78. Gumral D, Fogolari F, Corazza A, Viglino P, Giorgetti S, Stoppini M, et al. Reduction of conformational mobility and aggregation in W60G beta2-microglobulin: assessment by 15N NMR relaxation. Magnetic resonance in chemistry : MRC. 2013;51(12):795-807. Epub 2013/10/19. doi: 10.1002/mrc.4018. PubMed PMID: 24136818.
79. Narang SS, Shuaib S, Goyal D, Goyal B. Assessing the effect of D59P mutation in the DE loop region in amyloid aggregation propensity of beta2-microglobulin: A molecular dynamics simulation study. Journal of cellular biochemistry. 2018;119(1):782-92. Epub 2017/06/29. doi: 10.1002/jcb.26241. PubMed PMID: 28657666.
80. Natalello A, Relini A, Penco A, Halabelian L, Bolognesi M, Doglia SM, et al. Wild type beta-2 microglobulin and DE loop mutants display a common fibrillar architecture. PloS one. 2015;10(3):e0122449. Epub 2015/03/25. doi: 10.1371/journal.pone.0122449. PubMed PMID: 25803608; PubMed Central PMCID: PMC4372401.
81. Santambrogio C, Ricagno S, Colombo M, Barbiroli A, Bonomi F, Bellotti V, et al. DE-loop mutations affect beta2 microglobulin stability, oligomerization, and the low-pH unfolded form. Protein science : a publication of the Protein Society. 2010;19(7):1386-94. Epub 2010/05/28. doi: 10.1002/pro.419. PubMed PMID: 20506535; PubMed Central PMCID: PMC2974830.
82. Esposito G, Ricagno S, Corazza A, Rennella E, Gumral D, Mimmi MC, et al. The controlling roles of Trp60 and Trp95 in beta2-microglobulin function, folding and amyloid aggregation properties. Journal of molecular biology. 2008;378(4):887-97. Epub 2008/04/09. doi: 10.1016/j.jmb.2008.03.002. PubMed PMID: 18395224.
83. Ricagno S, Colombo M, de Rosa M, Sangiovanni E, Giorgetti S, Raimondi S, et al. DE loop mutations affect beta2-microglobulin stability and amyloid aggregation. Biochemical and biophysical research communications. 2008;377(1):146-50. Epub 2008/10/07. doi: 10.1016/j.bbrc.2008.09.108. PubMed PMID: 18835253.
84. Ricagno S, Raimondi S, Giorgetti S, Bellotti V, Bolognesi M. Human beta-2 microglobulin W60V mutant structure: Implications for stability and amyloid aggregation. Biochemical and biophysical research communications. 2009;380(3):543-7. Epub 2009/03/17. doi: 10.1016/j.bbrc.2009.01.116. PubMed PMID: 19284997.
85. Kihara M, Chatani E, Iwata K, Yamamoto K, Matsuura T, Nakagawa A, et al. Conformation of amyloid fibrils of beta2-microglobulin probed by tryptophan mutagenesis. The Journal of biological chemistry. 2006;281(41):31061-9. Epub 2006/08/12. doi: 10.1074/jbc.M605358200. PubMed PMID: 16901902.
86. Blaho DV, Miranker AD. Delineating the conformational elements responsible for Cu(2+)-induced oligomerization of beta-2 microglobulin. Biochemistry. 2009;48(28):6610-7. Epub 2009/06/13. doi: 10.1021/bi900540j. PubMed PMID: 19518133; PubMed Central PMCID: PMC3342574.
87. Rosano C, Zuccotti S, Mangione P, Giorgetti S, Bellotti V, Pettirossi F, et al. beta2-microglobulin H31Y variant 3D structure highlights the protein natural propensity towards intermolecular aggregation. Journal of molecular biology. 2004;335(4):1051-64. Epub 2003/12/31. PubMed PMID: 14698299.
88. Esposito G, Corazza A, Viglino P, Verdone G, Pettirossi F, Fogolari F, et al. Solution structure of beta(2)-microglobulin and insights into fibrillogenesis. Biochimica et biophysica acta.

2005;1753(1):76-84. Epub 2005/08/06. doi: 10.1016/j.bbapap.2005.07.003. PubMed PMID: 16081329.
89. Platt GW, Routledge KE, Homans SW, Radford SE. Fibril growth kinetics reveal a region of beta2-microglobulin important for nucleation and elongation of aggregation. Journal of molecular biology. 2008;378(1):251-63. Epub 2008/03/18. doi: 10.1016/j.jmb.2008.01.092. PubMed PMID: 18342332; PubMed Central PMCID: PMC2627305.
90. Eakin CM, Knight JD, Morgan CJ, Gelfand MA, Miranker AD. Formation of a copper specific binding site in non-native states of beta-2-microglobulin. Biochemistry. 2002;41(34):10646-56. Epub 2002/08/21. PubMed PMID: 12186550.
91. Verdone G, Corazza A, Viglino P, Pettirossi F, Giorgetti S, Mangione P, et al. The solution structure of human beta2-microglobulin reveals the prodromes of its amyloid transition. Protein science : a publication of the Protein Society. 2002;11(3):487-99. Epub 2002/02/16. doi: 10.1110/ps.29002. PubMed PMID: 11847272; PubMed Central PMCID: PMC2373469.
92. Bellotti V, Stoppini M, Mangione P, Sunde M, Robinson C, Asti L, et al. Beta2-microglobulin can be refolded into a native state from ex vivo amyloid fibrils. European journal of biochemistry / FEBS. 1998;258(1):61-7. Epub 1998/12/16. PubMed PMID: 9851692.
93. Stoppini M, Mangione P, Monti M, Giorgetti S, Marchese L, Arcidiaco P, et al. Proteomics of beta2-microglobulin amyloid fibrils. Biochimica et biophysica acta. 2005;1753(1):23-33. Epub 2005/09/13. doi: 10.1016/j.bbapap.2005.07.019. PubMed PMID: 16154394.
94. Monti M, Principe S, Giorgetti S, Mangione P, Merlini G, Clark A, et al. Topological investigation of amyloid fibrils obtained from beta2-microglobulin. Protein science : a publication of the Protein Society. 2002;11(10):2362-9. Epub 2002/09/19. doi: 10.1110/ps.0206902. PubMed PMID: 12237458; PubMed Central PMCID: PMC2373708.
95. Giorgetti S, Stoppini M, Tennent GA, Relini A, Marchese L, Raimondi S, et al. Lysine 58-cleaved beta2-microglobulin is not detectable by 2D electrophoresis in ex vivo amyloid fibrils of two patients affected by dialysis-related amyloidosis. Protein science : a publication of the Protein Society. 2007;16(2):343-9. Epub 2007/01/24. doi: 10.1110/ps.062563507. PubMed PMID: 17242436; PubMed Central PMCID: PMC2203293.
96. Myers SL, Jones S, Jahn TR, Morten IJ, Tennent GA, Hewitt EW, et al. A systematic study of the effect of physiological factors on beta2-microglobulin amyloid formation at neutral pH. Biochemistry. 2006;45(7):2311-21. Epub 2006/02/16. doi: 10.1021/bi052434i. PubMed PMID: 16475820.
97. Mangione PP, Esposito G, Relini A, Raimondi S, Porcari R, Giorgetti S, et al. Structure, folding dynamics, and amyloidogenesis of D76N beta2-microglobulin: roles of shear flow, hydrophobic surfaces, and alpha-crystallin. The Journal of biological chemistry. 2013;288(43):30917-30. Epub 2013/09/10. doi: 10.1074/jbc.M113.498857. PubMed PMID: 24014031; PubMed Central PMCID: PMC3829406.
98. de Rosa M, Halabelian L, Barbiroli A, Bolognesi M, Bellotti V, Ricagno S. An Asp to Asn mutation is a toxic trigger in beta-2 microglobulin: structure and biophysics. Amyloid. 2017;24(sup1):15-6. Epub 2017/04/25. doi: 10.1080/13506129.2016.1272450. PubMed PMID: 28434301.
99. Eichner T, Radford SE. A generic mechanism of beta2-microglobulin amyloid assembly at neutral pH involving a specific proline switch. Journal of molecular biology. 2009;386(5):1312-26. Epub 2009/05/20. PubMed PMID: 19452600.
100. Fabian H, Gast K, Laue M, Misselwitz R, Uchanska-Ziegler B, Ziegler A, et al. Early stages of misfolding and association of beta2-microglobulin: insights from infrared spectroscopy and dynamic light scattering. Biochemistry. 2008;47(26):6895-906. Epub 2008 Jun 10. doi: 10.1021/bi800279y.
101. Halabelian L, Relini A, Barbiroli A, Penco A, Bolognesi M, Ricagno S. A covalent homodimer probing early oligomers along amyloid aggregation. Scientific reports. 2015;5:14651. Epub 2015/10/01. doi: 10.1038/srep14651. PubMed PMID: 26420657; PubMed Central PMCID: PMC4588566.

102. Benyamini H, Gunasekaran K, Wolfson H, Nussinov R. Fibril modelling by sequence and structure conservation analysis combined with protein docking techniques: beta(2)-microglobulin amyloidosis. Biochimica et biophysica acta. 2005;1753(1):121-30. Epub 2005/08/19. doi: 10.1016/j.bbapap.2005.07.012. PubMed PMID: 16107326.
103. Mendoza VL, Antwi K, Baron-Rodriguez MA, Blanco C, Vachet RW. Structure of the preamyloid dimer of beta-2-microglobulin from covalent labeling and mass spectrometry. Biochemistry. 2010;49(7):1522-32. Epub 2010/01/22. doi: 10.1021/bi901748h. PubMed PMID: 20088607; PubMed Central PMCID: PMC2848472.
104. Liu C, Sawaya MR, Eisenberg D. beta(2)-microglobulin forms three-dimensional domain-swapped amyloid fibrils with disulfide linkages. Nature structural & molecular biology. 2011;18(1):49-55. Epub 2010/12/07. doi: 10.1038/nsmb.1948. PubMed PMID: 21131979; PubMed Central PMCID: PMC3058263.
105. Benyamini H, Gunasekaran K, Wolfson H, Nussinov R. Beta2-microglobulin amyloidosis: insights from conservation analysis and fibril modelling by protein docking techniques. Journal of molecular biology. 2003;330(1):159-74. Epub 2003/06/24. PubMed PMID: 12818210.
106. Fogolari F, Corazza A, Viglino P, Zuccato P, Pieri L, Faccioli P, et al. Molecular dynamics simulation suggests possible interaction patterns at early steps of beta2-microglobulin aggregation. Biophysical journal. 2007;92(5):1673-81. Epub 2006/12/13. doi: 10.1529/biophysj.106.098483. PubMed PMID: 17158575; PubMed Central PMCID: PMC1796822.
107. Radford SE, Gosal WS, Platt GW. Towards an understanding of the structural molecular mechanism of beta(2)-microglobulin amyloid formation in vitro. Biochimica et biophysica acta. 2005;1753(1):51-63. Epub 2005/08/16. doi: 10.1016/j.bbapap.2005.07.006. PubMed PMID: 16099226.
108. Srikanth R, Mendoza VL, Bridgewater JD, Zhang G, Vachet RW. Copper binding to beta-2-microglobulin and its pre-amyloid oligomers. Biochemistry. 2009;48(41):9871-81. Epub 2009/09/17. doi: 10.1021/bi901172y. PubMed PMID: 19754160; PubMed Central PMCID: PMC2766818.
109. Colombo M, de Rosa M, Bellotti V, Ricagno S, Bolognesi M. A recurrent D-strand association interface is observed in beta-2 microglobulin oligomers. The FEBS journal. 2012;279(6):1131-43. Epub 2012/02/01. doi: 10.1111/j.1742-4658.2012.08510.x. PubMed PMID: 22289140.
110. Chen Y, Dokholyan NV. A single disulfide bond differentiates aggregation pathways of beta2-microglobulin. Journal of molecular biology. 2005;354(2):473-82. Epub 2005/10/26. doi: 10.1016/j.jmb.2005.09.075. PubMed PMID: 16242719.
111. Smith DP, Radford SE, Ashcroft AE. Elongated oligomers in beta2-microglobulin amyloid assembly revealed by ion mobility spectrometry-mass spectrometry. Proceedings of the National Academy of Sciences of the United States of America. 2010;107(15):6794-8. Epub 2010/03/31. doi: 10.1073/pnas.0913046107. PubMed PMID: 20351246; PubMed Central PMCID: PMC2872402.
112. Smith DP, Woods LA, Radford SE, Ashcroft AE. Structure and dynamics of oligomeric intermediates in beta2-microglobulin self-assembly. Biophysical journal. 2011;101(5):1238-47. Epub 2011/09/06. doi: 10.1016/j.bpj.2011.07.023. PubMed PMID: 21889462; PubMed Central PMCID: PMC3164137.
113. Antwi K, Mahar M, Srikanth R, Olbris MR, Tyson JF, Vachet RW. Cu(II) organizes beta-2-microglobulin oligomers but is released upon amyloid formation. Protein science : a publication of the Protein Society. 2008;17(4):748-59. Epub 2008/02/29. doi: 10.1110/ps.073249008. PubMed PMID: 18305198; PubMed Central PMCID: PMC2271169.
114. White HE, Hodgkinson JL, Jahn TR, Cohen-Krausz S, Gosal WS, Muller S, et al. Globular tetramers of beta(2)-microglobulin assemble into elaborate amyloid fibrils. Journal of molecular biology. 2009;389(1):48-57. Epub 2009/04/07. doi: 10.1016/j.jmb.2009.03.066. PubMed PMID: 19345691; PubMed Central PMCID: PMC2726924.
115. Mendoza VL, Baron-Rodriguez MA, Blanco C, Vachet RW. Structural insights into the pre-amyloid tetramer of beta-2-microglobulin from covalent labeling and mass spectrometry. Biochemistry. 2011;50(31):6711-22. Epub 2011/07/02. doi: 10.1021/bi2004894. PubMed PMID: 21718071; PubMed Central PMCID: PMC3149750.

116. Loureiro RJS, Vila-Vicosa D, Machuqueiro M, Shakhnovich EI, Faisca PFN. A tale of two tails: the importance of unstructured termini in the aggregation pathway of beta2-microglobulin. Proteins. 2017. Epub 2017/07/27. doi: 10.1002/prot.25358. PubMed PMID: 28745031.
117. Shao J, Tanner, S. W., Thompson, N. et al. Clustering Molecular Dynamics Trajectories: 1. Characterizing the Performance of Different Clustering Algorithms. Journal of Chemical Theory and Computation. 2007;3(6):2312–34. doi: 10.1021/ct700119m.
118. Karpen ME, Tobias DJ, Brooks CL, 3rd. Statistical clustering techniques for the analysis of long molecular dynamics trajectories: analysis of 2.2-ns trajectories of YPGDV. Biochemistry. 1993;32(2):412-20. Epub 1993/01/19. PubMed PMID: 8422350.
119. Shenkin PS, Mcdonald DQ. Cluster-Analysis of Molecular-Conformations. Journal of computational chemistry. 1994;15(8):899-916. doi: DOI 10.1002/jcc.540150811. PubMed PMID: ISI:A1994NX40800010.
120. Segaran T. Programming Collective Intelligence O'Reilly Media; 2007.
121. Cormack RM. A review of classification. J R Stat Soc A. 1971;134:321-67.
122. Jain AK, Murty MN, Flynn PJ. Data clustering: A review. Acm Comput Surv. 1999;31(3):264-323. doi: Doi 10.1145/331499.331504. PubMed PMID: ISI:000086365400002.
123. Poncin M, Hartmann B, Lavery R. Conformational sub-states in B-DNA. Journal of molecular biology. 1992;226(3):775-94. Epub 1992/08/05. PubMed PMID: 1507226.
124. Torda AE, Vangunsteren WF. Algorithms for Clustering Molecular-Dynamics Configurations. Journal of computational chemistry. 1994;15(12):1331-40. doi: DOI 10.1002/jcc.540151203. PubMed PMID: ISI:A1994PT19200002.
125. Feig M, Karanicolas J, Brooks CL, 3rd. MMTSB Tool Set: enhanced sampling and multiscale modeling methods for applications in structural biology. Journal of molecular graphics & modelling. 2004;22(5):377-95. Epub 2004/04/22. doi: 10.1016/j.jmgm.2003.12.005. PubMed PMID: 15099834.
126. Lee B, Richards FM. The interpretation of protein structures: estimation of static accessibility. Journal of molecular biology. 1971;55(3):379-400. Epub 1971/02/14. PubMed PMID: 5551392.
127. Tozzini V. Coarse-grained models for proteins. Current opinion in structural biology. 2005;15(2):144-50. Epub 2005/04/20. doi: 10.1016/j.sbi.2005.02.005. PubMed PMID: 15837171.
128. Noid WG. Perspective: Coarse-grained models for biomolecular systems. The Journal of chemical physics. 2013;139(9):090901. Epub 2013/09/14. doi: 10.1063/1.4818908. PubMed PMID: 24028092.
129. Chiu W, Baker ML, Almo SC. Structural biology of cellular machines. Trends in cell biology. 2006;16(3):144-50. Epub 2006/02/07. doi: 10.1016/j.tcb.2006.01.002. PubMed PMID: 16459078.
130. Tsuchiya Y, Kinoshita K, Nakamura H. Analyses of homo-oligomer interfaces of proteins from the complementarity of molecular surface, electrostatic potential and hydrophobicity. Protein engineering, design & selection : PEDS. 2006;19(9):421-9. Epub 2006/07/14. doi: 10.1093/protein/gzl026. PubMed PMID: 16837482.
131. Jones S, Thornton JM. Protein-protein interactions: a review of protein dimer structures. Progress in biophysics and molecular biology. 1995;63(1):31-65. Epub 1995/01/01. PubMed PMID: 7746868.
132. Jones S, Thornton JM. Principles of protein-protein interactions. Proceedings of the National Academy of Sciences of the United States of America. 1996;93(1):13-20. Epub 1996/01/09. PubMed PMID: 8552589; PubMed Central PMCID: PMC40170.
133. Bickerton GR, Higueruelo AP, Blundell TL. Comprehensive, atomic-level characterization of structurally characterized protein-protein interactions: the PICCOLO database. BMC bioinformatics. 2011;12:313. Epub 2011/08/02. doi: 10.1186/1471-2105-12-313. PubMed PMID: 21801404; PubMed Central PMCID: PMC3161047.
134. Tina KG, Bhadra R, Srinivasan N. PIC: Protein Interactions Calculator. Nucleic acids research. 2007;35(Web Server issue):W473-6. Epub 2007/06/23. doi: 10.1093/nar/gkm423. PubMed PMID: 17584791; PubMed Central PMCID: PMC1933215.
135. Cummings MD, Hart TN, Read RJ. Atomic solvation parameters in the analysis of protein-protein docking results. Protein science : a publication of the Protein Society. 1995;4(10):2087-99.

Epub 1995/10/01. doi: 10.1002/pro.5560041014. PubMed PMID: 8535245; PubMed Central PMCID: PMC2142991.
136. Eisenberg D, McLachlan AD. Solvation energy in protein folding and binding. Nature. 1986;319(6050):199-203. Epub 1986/01/16. doi: 10.1038/319199a0. PubMed PMID: 3945310.
137. Wesson L, Eisenberg D. Atomic solvation parameters applied to molecular dynamics of proteins in solution. Protein science : a publication of the Protein Society. 1992;1(2):227-35. Epub 1992/02/01. doi: 10.1002/pro.5560010204. PubMed PMID: 1304905; PubMed Central PMCID: PMC2142195.
138. Fauchere J-L, Pliska V. Hydrophobic parameters pi of amino-acid side chains from the partitioning of N-acetyl-amino-acid amides. Eur J Med Chem. 1983;18(3):369-75.
139. Radzicka A, Wolfenden, R. Comparing the polarities of the amino acids: side-chain distribution coefficients between the vapor phase, cyclohexane, 1-octanol, and neutral aqueous solution. Biochemistry. 1988;27(5):1664–70.
140. Wolfenden R, Andersson L, Cullis PM, Southgate CC. Affinities of amino acid side chains for solvent water. Biochemistry. 1981;20(4):849-55. Epub 1981/02/17. PubMed PMID: 7213619.
141. Schiffer CA, Caldwell JW, Kollman PA, Stroud RM. Protein Structure Prediction with a Combined Solvation Free Energy-Molecular Mechanics Force Field. Molecular Simulation. 1993;10(2-6):121-49. doi: 10.1080/08927029308022162.
142. Kyte J. The basis of the hydrophobic effect. Biophysical chemistry. 2003;100(1-3):193-203. Epub 2003/03/21. PubMed PMID: 12646366.
143. Gitlin I, Carbeck JD, Whitesides GM. Why are proteins charged? Networks of charge-charge interactions in proteins measured by charge ladders and capillary electrophoresis. Angew Chem Int Ed Engl. 2006;45(19):3022-60. Epub 2006/04/19. doi: 10.1002/anie.200502530. PubMed PMID: 16619322.
144. Dill KA. Dominant forces in protein folding. Biochemistry. 1990;29(31):7133-55. Epub 1990/08/07. PubMed PMID: 2207096.
145. Bashford D, Karplus M. pKa's of ionizable groups in proteins: atomic detail from a continuum electrostatic model. Biochemistry. 1990;29(44):10219-25. Epub 1990/11/06. PubMed PMID: 2271649.
146. Tanford C. The Interpretation Of Hydrogen Ion Titration Curves Of Proteins. Advances in Protein Chemistry. 1962;17:69-165.
147. Dominy BN, Minoux H, Brooks CL, 3rd. An electrostatic basis for the stability of thermophilic proteins. Proteins. 2004;57(1):128-41. Epub 2004/08/25. doi: 10.1002/prot.20190. PubMed PMID: 15326599.
148. Honig B, Nicholls A. Classical electrostatics in biology and chemistry. Science. 1995;268(5214):1144-9. Epub 1995/05/26. PubMed PMID: 7761829.
149. Nicholls A HB. A rapid finite difference algorithm, utilizing successive over-relaxation to solve the Poisson–Boltzmann equation. J Comput Chem. 1991;12(4):435-45. doi: https://doi.org/10.1002/jcc.540120405.
150. Smith GR, Sternberg MJ. Prediction of protein-protein interactions by docking methods. Current opinion in structural biology. 2002;12(1):28-35. Epub 2002/02/13. PubMed PMID: 11839486.
151. Urbanc B, Borreguero JM, Cruz L, Stanley HE. Ab initio discrete molecular dynamics approach to protein folding and aggregation. Methods in enzymology. 2006;412:314-38. Epub 2006/10/19. doi: 10.1016/S0076-6879(06)12019-4. PubMed PMID: 17046666.
152. Yun S, Urbanc B, Cruz L, Bitan G, Teplow DB, Stanley HE. Role of electrostatic interactions in amyloid beta-protein (A beta) oligomer formation: a discrete molecular dynamics study. Biophysical journal. 2007;92(11):4064-77. Epub 2007/02/20. doi: 10.1529/biophysj.106.097766. PubMed PMID: 17307823; PubMed Central PMCID: PMC1868995.
153. Luisi DL, Snow CD, Lin JJ, Hendsch ZS, Tidor B, Raleigh DP. Surface salt bridges, double-mutant cycles, and protein stability: an experimental and computational analysis of the interaction of the Asp 23 side chain with the N-terminus of the N-terminal domain of the ribosomal protein l9.

Biochemistry. 2003;42(23):7050-60. Epub 2003/06/11. doi: 10.1021/bi027202n. PubMed PMID: 12795600.
154. Biedermann F, Schneider HJ. Experimental Binding Energies in Supramolecular Complexes. Chemical reviews. 2016;116(9):5216-300. Epub 2016/05/04. doi: 10.1021/acs.chemrev.5b00583. PubMed PMID: 27136957.
155. Kortemme T, Morozov AV, Baker D. An orientation-dependent hydrogen bonding potential improves prediction of specificity and structure for proteins and protein-protein complexes. Journal of molecular biology. 2003;326(4):1239-59. Epub 2003/02/19. PubMed PMID: 12589766.
156. Sheu SY, Yang DY, Selzle HL, Schlag EW. Energetics of hydrogen bonds in peptides. Proceedings of the National Academy of Sciences of the United States of America. 2003;100(22):12683-7. Epub 2003/10/16. doi: 10.1073/pnas.2133366100. PubMed PMID: 14559970; PubMed Central PMCID: PMC240678.
157. Brändén CI, Tooze J. Introduction to Protein Structure: Garland Pub.; 1999.
158. Baker EN, Hubbard RE. Hydrogen bonding in globular proteins. Progress in biophysics and molecular biology. 1984;44(2):97-179. Epub 1984/01/01. PubMed PMID: 6385134.
159. Stickle DF, Presta LG, Dill KA, Rose GD. Hydrogen bonding in globular proteins. Journal of molecular biology. 1992;226(4):1143-59. Epub 1992/08/20. PubMed PMID: 1518048.
160. Jeffrey GA. An Introduction to Hydrogen Bonding. Oxford: Oxford University Press; 1997.
161. Xu D, Tsai CJ, Nussinov R. Hydrogen bonds and salt bridges across protein-protein interfaces. Protein engineering. 1997;10(9):999-1012. Epub 1998/02/17. PubMed PMID: 9464564.
162. Xu D, Lin SL, Nussinov R. Protein binding versus protein folding: the role of hydrophilic bridges in protein associations. Journal of molecular biology. 1997;265(1):68-84. Epub 1997/01/10. doi: 10.1006/jmbi.1996.0712. PubMed PMID: 8995525.
163. Tsai CJ, Lin SL, Wolfson HJ, Nussinov R. Studies of protein-protein interfaces: a statistical analysis of the hydrophobic effect. Protein science : a publication of the Protein Society. 1997;6(1):53-64. Epub 1997/01/01. doi: 10.1002/pro.5560060106. PubMed PMID: 9007976; PubMed Central PMCID: PMC2143524.
164. Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, et al. The Protein Data Bank. Nucleic acids research. 2000;28(1):235-42. Epub 1999/12/11. doi: 10.1093/nar/28.1.235. PubMed PMID: 10592235; PubMed Central PMCID: PMC102472.
165. Janin J. Protein-protein docking tested in blind predictions: the CAPRI experiment. Molecular bioSystems. 2010;6(12):2351-62. Epub 2010/08/21. doi: 10.1039/c005060c. PubMed PMID: 20725658.
166. Dutta S, Berman HM. Large macromolecular complexes in the Protein Data Bank: a status report. Structure. 2005;13(3):381-8. Epub 2005/03/16. doi: 10.1016/j.str.2005.01.008. PubMed PMID: 15766539.
167. Vakser IA. Protein-protein docking: from interaction to interactome. Biophysical journal. 2014;107(8):1785-93. Epub 2014/11/25. doi: 10.1016/j.bpj.2014.08.033. PubMed PMID: 25418159; PubMed Central PMCID: PMC4213718.
168. 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. Epub 2002/05/10. doi: 10.1002/prot.10115. PubMed PMID: 12001221.
169. Vajda S, Kozakov D. Convergence and combination of methods in protein-protein docking. Current opinion in structural biology. 2009;19(2):164-70. Epub 2009/03/31. doi: 10.1016/j.sbi.2009.02.008. PubMed PMID: 19327983; PubMed Central PMCID: PMC2763924.
170. Ritchie DW. Recent progress and future directions in protein-protein docking. Current protein & peptide science. 2008;9(1):1-15. Epub 2008/03/14. PubMed PMID: 18336319.
171. Vakser IA, Kundrotas P. Predicting 3D structures of protein-protein complexes. Current pharmaceutical biotechnology. 2008;9(2):57-66. Epub 2008/04/09. PubMed PMID: 18393862.
172. Wei G, Mousseau N, Derreumaux P. Computational simulations of the early steps of protein aggregation. Prion. 2007;1(1):3-8. Epub 2007/01/01. PubMed PMID: 19164927; PubMed Central PMCID: PMC2633700.

173. Saladin A. PC. Protein-Protein Docking. In: M. Z, editor. Protein-protein complexes: Analysis, Modeling and Drug Design. Bremen: Imperial College Press; 2010. p. 147-77.
174. Chen R, Weng Z. Docking unbound proteins using shape complementarity, desolvation, and electrostatics. Proteins. 2002;47(3):281-94. Epub 2002/04/12. PubMed PMID: 11948782.
175. Eisenstein M, Katchalski-Katzir E. On proteins, grids, correlations, and docking. Comptes rendus biologies. 2004;327(5):409-20. Epub 2004/07/17. PubMed PMID: 15255472.
176. Mandell JG, Roberts VA, Pique ME, Kotlovyi V, Mitchell JC, Nelson E, et al. Protein docking using continuum electrostatics and geometric fit. Protein engineering. 2001;14(2):105-13. Epub 2001/04/12. PubMed PMID: 11297668.
177. Heifetz A, Katchalski-Katzir E, Eisenstein M. Electrostatics in protein-protein docking. Protein science : a publication of the Protein Society. 2002;11(3):571-87. Epub 2002/02/16. PubMed PMID: 11847280; PubMed Central PMCID: PMC2373470.
178. Vakser IA, Aflalo C. Hydrophobic docking: a proposed enhancement to molecular recognition techniques. Proteins. 1994;20(4):320-9. Epub 1994/12/01. doi: 10.1002/prot.340200405. PubMed PMID: 7731951.
179. Berchanski A, Shapira B, Eisenstein M. Hydrophobic complementarity in protein-protein docking. Proteins. 2004;56(1):130-42. Epub 2004/05/27. doi: 10.1002/prot.20145. PubMed PMID: 15162493.
180. Katchalski-Katzir E, Shariv I, Eisenstein M, Friesem AA, Aflalo C, Vakser IA. 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. Epub 1992/03/15. PubMed PMID: 1549581; PubMed Central PMCID: PMC48623.
181. Kozakov D, Brenke R, Comeau SR, Vajda S. PIPER: an FFT-based protein docking program with pairwise potentials. Proteins. 2006;65(2):392-406. Epub 2006/08/26. doi: 10.1002/prot.21117. PubMed PMID: 16933295.
182. Chen R, Li L, Weng Z. ZDOCK: an initial-stage protein-docking algorithm. Proteins. 2003;52(1):80-7. Epub 2003/06/05. doi: 10.1002/prot.10389. PubMed PMID: 12784371.
183. Schneidman-Duhovny D, Inbar Y, Polak V, Shatsky M, Halperin I, Benyamini H, et al. Taking geometry to its edge: fast unbound rigid (and hinge-bent) docking. Proteins. 2003;52(1):107-12. Epub 2003/06/05. doi: 10.1002/prot.10397. PubMed PMID: 12784375.
184. Schneidman-Duhovny D, Inbar Y, Nussinov R, Wolfson HJ. Geometry-based flexible and symmetric protein docking. Proteins. 2005;60(2):224-31. Epub 2005/06/28. doi: 10.1002/prot.20562. PubMed PMID: 15981269.
185. Terashi G, Takeda-Shitaka M, Kanou K, Iwadate M, Takaya D, Umeyama H. The SKE-DOCK server and human teams based on a combined method of shape complementarity and free energy estimation. Proteins. 2007;69(4):866-72. Epub 2007/09/14. doi: 10.1002/prot.21772. PubMed PMID: 17853449.
186. Fernandez-Recio J, Totrov M, Abagyan R. ICM-DISCO docking by global energy optimization with fully flexible side-chains. Proteins. 2003;52(1):113-7. Epub 2003/06/05. doi: 10.1002/prot.10383. PubMed PMID: 12784376.
187. Gray JJ, Moughon S, Wang C, Schueler-Furman O, Kuhlman B, Rohl CA, et al. Protein-protein docking with simultaneous optimization of rigid-body displacement and side-chain conformations. Journal of molecular biology. 2003;331(1):281-99. Epub 2003/07/24. PubMed PMID: 12875852.
188. Dominguez C, Boelens R, Bonvin AM. HADDOCK: a protein-protein docking approach based on biochemical or biophysical information. Journal of the American Chemical Society. 2003;125(7):1731-7. Epub 2003/02/13. doi: 10.1021/ja026939x. PubMed PMID: 12580598.
189. Kozakov D, Clodfelter KH, Vajda S, Camacho CJ. Optimal clustering for detecting near-native conformations in protein docking. Biophysical journal. 2005;89(2):867-75. Epub 2005/05/24. doi: 10.1529/biophysj.104.058768. PubMed PMID: 15908573; PubMed Central PMCID: PMC1366636.
190. Lorenzen S, Zhang Y. Identification of near-native structures by clustering protein docking conformations. Proteins. 2007;68(1):187-94. Epub 2007/04/03. doi: 10.1002/prot.21442. PubMed PMID: 17397057.

191. Wiehe K, Pierce B, Mintseris J, Tong WW, Anderson R, Chen R, et al. ZDOCK and RDOCK performance in CAPRI rounds 3, 4, and 5. Proteins. 2005;60(2):207-13. Epub 2005/06/28. doi: 10.1002/prot.20559. PubMed PMID: 15981263.
192. Zacharias M. ATTRACT: protein-protein docking in CAPRI using a reduced protein model. Proteins. 2005;60(2):252-6. Epub 2005/06/28. doi: 10.1002/prot.20566. PubMed PMID: 15981270.
193. Schueler-Furman O, Wang C, Baker D. Progress in protein-protein docking: atomic resolution predictions in the CAPRI experiment using RosettaDock with an improved treatment of side-chain flexibility. Proteins. 2005;60(2):187-94. Epub 2005/06/28. doi: 10.1002/prot.20556. PubMed PMID: 15981249.
194. Wang C, Schueler-Furman O, Andre I, London N, Fleishman SJ, Bradley P, et al. RosettaDock in CAPRI rounds 6-12. Proteins. 2007;69(4):758-63. Epub 2007/08/03. doi: 10.1002/prot.21684. PubMed PMID: 17671979.
195. Chaudhury S, Gray JJ. Conformer selection and induced fit in flexible backbone protein-protein docking using computational and NMR ensembles. Journal of molecular biology. 2008;381(4):1068-87. Epub 2008/07/22. doi: 10.1016/j.jmb.2008.05.042. PubMed PMID: 18640688; PubMed Central PMCID: PMC2573042.
196. Janin J, Henrick K, Moult J, Eyck LT, Sternberg MJ, Vajda S, et al. CAPRI: a Critical Assessment of PRedicted Interactions. Proteins. 2003;52(1):2-9. Epub 2003/06/05. doi: 10.1002/prot.10381. PubMed PMID: 12784359.
197. Gabb HA, Jackson RM, Sternberg MJ. Modelling protein docking using shape complementarity, electrostatics and biochemical information. Journal of molecular biology. 1997;272(1):106-20. Epub 1997/09/23. doi: 10.1006/jmbi.1997.1203. PubMed PMID: 9299341.
198. Tovchigrechko A, Vakser IA. GRAMM-X public web server for protein-protein docking. Nucleic acids research. 2006;34(Web Server issue):W310-4. Epub 2006/07/18. doi: 10.1093/nar/gkl206. PubMed PMID: 16845016; PubMed Central PMCID: PMC1538913.
199. Grosdidier S, Fernandez-Recio J. Protein-protein docking and hot-spot prediction for drug discovery. Current pharmaceutical design. 2012;18(30):4607-18. Epub 2012/06/02. PubMed PMID: 22650255.
200. Ritchie DW, Kemp GJL. Protein docking using spherical polar Fourier correlations. Proteins-Structure Function and Genetics. 2000;39(2):178-94. doi: Doi 10.1002/(Sici)1097-0134(20000501)39:2<178::Aid-Prot8>3.3.Co;2-Y. PubMed PMID: ISI:000086061500008.
201. Garzon JI, Lopez-Blanco JR, Pons C, Kovacs J, Abagyan R, Fernandez-Recio J, et al. FRODOCK: a new approach for fast rotational protein-protein docking. Bioinformatics. 2009;25(19):2544-51. Epub 2009/07/22. doi: 10.1093/bioinformatics/btp447. PubMed PMID: 19620099; PubMed Central PMCID: PMC2800348.
202. Connolly ML. Solvent-accessible surfaces of proteins and nucleic acids. Science. 1983;221(4612):709-13. Epub 1983/08/19. PubMed PMID: 6879170.
203. Connolly ML. Analytical Molecular-Surface Calculation. J Appl Crystallogr. 1983;16(Oct):548-58. doi: Doi 10.1107/S0021889883010985. PubMed PMID: ISI:A1983RL07000018.
204. Norel R, Lin SL, Wolfson HJ, Nussinov R. Shape complementarity at protein-protein interfaces. Biopolymers. 1994;34(7):933-40. Epub 1994/07/01. doi: 10.1002/bip.360340711. PubMed PMID: 8054472.
205. Inbar Y, Schneidman-Duhovny D, Halperin I, Oron A, Nussinov R, Wolfson HJ. Approaching the CAPRI challenge with an efficient geometry-based docking. Proteins. 2005;60(2):217-23. Epub 2005/06/28. doi: 10.1002/prot.20561. PubMed PMID: 15981251.
206. Schneidman-Duhovny D, Nussinov R, Wolfson HJ. Automatic prediction of protein interactions with large scale motion. Proteins. 2007;69(4):764-73. Epub 2007/09/22. doi: 10.1002/prot.21759. PubMed PMID: 17886339.
207. Schneidman-Duhovny D, Inbar Y, Nussinov R, Wolfson HJ. PatchDock and SymmDock: servers for rigid and symmetric docking. Nucleic acids research. 2005;33(Web Server issue):W363-7. Epub 2005/06/28. doi: 10.1093/nar/gki481. PubMed PMID: 15980490; PubMed Central PMCID: PMC1160241.

208. Inbar Y, Benyamini H, Nussinov R, Wolfson HJ. Protein structure prediction via combinatorial assembly of sub-structural units. Bioinformatics. 2003;19 Suppl 1:i158-68. Epub 2003/07/12. PubMed PMID: 12855452.
209. Inbar Y, Benyamini H, Nussinov R, Wolfson HJ. Prediction of multimolecular assemblies by multiple docking. Journal of molecular biology. 2005;349(2):435-47. Epub 2005/05/14. doi: 10.1016/j.jmb.2005.03.039. PubMed PMID: 15890207.
210. Qin S, Zhou HX. A holistic approach to protein docking. Proteins. 2007;69(4):743-9. Epub 2007/09/07. doi: 10.1002/prot.21752. PubMed PMID: 17803232.
211. Kanamori E, Murakami Y, Tsuchiya Y, Standley DM, Nakamura H, Kinoshita K. Docking of protein molecular surfaces with evolutionary trace analysis. Proteins. 2007;69(4):832-8. Epub 2007/09/07. doi: 10.1002/prot.21737. PubMed PMID: 17803239.
212. Bernauer J, Aze J, Janin J, Poupon A. A new protein-protein docking scoring function based on interface residue properties. Bioinformatics. 2007;23(5):555-62. Epub 2007/01/24. doi: 10.1093/bioinformatics/btl654. PubMed PMID: 17237048.
213. Zacharias M. Protein-protein docking with a reduced protein model accounting for side-chain flexibility. Protein science : a publication of the Protein Society. 2003;12(6):1271-82. Epub 2003/05/23. doi: 10.1110/ps.0239303. PubMed PMID: 12761398; PubMed Central PMCID: PMC2323887.
214. Moreira IS, Fernandes PA, Ramos MJ. Protein-protein docking dealing with the unknown. Journal of computational chemistry. 2010;31(2):317-42. Epub 2009/05/23. doi: 10.1002/jcc.21276. PubMed PMID: 19462412.
215. Wang C, Bradley P, Baker D. Protein-protein docking with backbone flexibility. Journal of molecular biology. 2007;373(2):503-19. Epub 2007/09/11. doi: 10.1016/j.jmb.2007.07.050. PubMed PMID: 17825317.
216. de Vries SJ, van Dijk M, Bonvin AM. The HADDOCK web server for data-driven biomolecular docking. Nature protocols. 2010;5(5):883-97. Epub 2010/05/01. doi: 10.1038/nprot.2010.32. PubMed PMID: 20431534.
217. van Dijk AD, de Vries SJ, Dominguez C, Chen H, Zhou HX, Bonvin AM. Data-driven docking: HADDOCK's adventures in CAPRI. Proteins. 2005;60(2):232-8. Epub 2005/06/28. doi: 10.1002/prot.20563. PubMed PMID: 15981252.
218. de Vries SJ, van Dijk AD, Krzeminski M, van Dijk M, Thureau A, Hsu V, et al. HADDOCK versus HADDOCK: new features and performance of HADDOCK2.0 on the CAPRI targets. Proteins. 2007;69(4):726-33. Epub 2007/09/07. doi: 10.1002/prot.21723. PubMed PMID: 17803234.
219. Janin J. The targets of CAPRI rounds 3-5. Proteins-Structure Function and Bioinformatics. 2005;60(2):170-5. doi: 10.1002/prot.20553. PubMed PMID: ISI:000230249000003.
220. Krobath H, Estacio SG, Faisca PF, Shakhnovich EI. Identification of a conserved aggregation-prone intermediate state in the folding pathways of Spc-SH3 amyloidogenic variants. Journal of molecular biology. 2012;422(5):705-22. Epub 2012/06/26. doi: 10.1016/j.jmb.2012.06.020. PubMed PMID: 22727745.
221. Li Y, Zhang X, Cao D. The role of shape complementarity in the protein-protein interactions. Scientific reports. 2013;3:3271. Epub 2013/11/21. doi: 10.1038/srep03271. PubMed PMID: 24253561; PubMed Central PMCID: PMC3834541.
222. Seeliger D, de Groot BL. Atomic contacts in protein structures. A detailed analysis of atomic radii, packing, and overlaps. Proteins. 2007;68(3):595-601. Epub 2007/05/19. doi: 10.1002/prot.21447. PubMed PMID: 17510956.
223. Urbanc B, Cruz L, Yun S, Buldyrev SV, Bitan G, Teplow DB, et al. In silico study of amyloid beta-protein folding and oligomerization. Proceedings of the National Academy of Sciences of the United States of America. 2004;101(50):17345-50. Epub 2004/12/08. doi: 10.1073/pnas.0408153101. PubMed PMID: 15583128; PubMed Central PMCID: PMC536046.
224. Teixeira VH, Cunha CA, Machuqueiro M, Oliveira AS, Victor BL, Soares CM, et al. On the use of different dielectric constants for computing individual and pairwise terms in poisson-boltzmann

studies of protein ionization equilibrium. The journal of physical chemistry B. 2005;109(30):14691-706. Epub 2006/07/21. doi: 10.1021/jp052259f. PubMed PMID: 16852854.
225. Ding F, Buldyrev SV, Dokholyan NV. Folding Trp-cage to NMR resolution native structure using a coarse-grained protein model. Biophysical journal. 2005;88(1):147-55. doi: 10.1529/biophysj.104.046375. PubMed PMID: ISI:000226090900016.
226. Lesser GJ, Rose GD. Hydrophobicity of amino acid subgroups in proteins. Proteins. 1990;8(1):6-13. Epub 1990/01/01. doi: 10.1002/prot.340080104. PubMed PMID: 2217164.
227. Van Der Spoel D, Lindahl E, Hess B, Groenhof G, Mark AE, Berendsen HJ. GROMACS: fast, flexible, and free. Journal of computational chemistry. 2005;26(16):1701-18. Epub 2005/10/08. doi: 10.1002/jcc.20291. PubMed PMID: 16211538.
228. Varadi M, Kosol S, Lebrun P, Valentini E, Blackledge M, Dunker AK, et al. pE-DB: a database of structural ensembles of intrinsically disordered and of unfolded proteins. Nucleic acids research. 2014;42(Database issue):D326-35. Epub 2013/11/01. doi: 10.1093/nar/gkt960. PubMed PMID: 24174539; PubMed Central PMCID: PMC3964940.
229. Stefanis L. α-Synuclein in Parkinson's Disease. Cold Spring Harb Perspect Med. 2012;2(2).
230. Srinivasan J, Cheatham TE, Cieplak P, Kollman PA, Case DA. Continuum solvent studies of the stability of DNA, RNA, and phosphoramidate - DNA helices. Journal of the American Chemical Society. 1998;120(37):9401-9. doi: Doi 10.1021/Ja981844+. PubMed PMID: ISI:000076117200001.
231. Kollman PA, Massova I, Reyes C, Kuhn B, Huo SH, Chong L, et al. Calculating structures and free energies of complex molecules: Combining molecular mechanics and continuum models. Accounts Chem Res. 2000;33(12):889-97. doi: 10.1021/ar000033j. PubMed PMID: ISI:000166180700009.
232. Genheden S, Ryde U. The MM/PBSA and MM/GBSA methods to estimate ligand-binding affinities. Expert opinion on drug discovery. 2015;10(5):449-61. Epub 2015/04/04. doi: 10.1517/17460441.2015.1032936. PubMed PMID: 25835573; PubMed Central PMCID: PMC4487606.
233. Kuhn B, Kollman PA. Binding of a diverse set of ligands to avidin and streptavidin: an accurate quantitative prediction of their relative affinities by a combination of molecular mechanics and continuum solvent models. Journal of medicinal chemistry. 2000;43(20):3786-91. Epub 2000/10/06. PubMed PMID: 11020294.
234. Nicholls A, Honig B. A Rapid Finite-Difference Algorithm, Utilizing Successive over-Relaxation to Solve the Poisson-Boltzmann Equation. Journal of computational chemistry. 1991;12(4):435-45. doi: DOI 10.1002/jcc.540120405. PubMed PMID: ISI:A1991FG56800004.
235. Sitkoff D, Sharp KA, Honig B. Accurate Calculation of Hydration Free-Energies Using Macroscopic Solvent Models. J Phys Chem-Us. 1994;98(7):1978-88. doi: Doi 10.1021/J100058a043. PubMed PMID: ISI:A1994MX85400043.
236. R JSL, Vila-Vicosa D, Machuqueiro M, Shakhnovich EI, P FNF. The Early Phase of beta2m Aggregation: An Integrative Computational Study Framed on the D76N Mutant and the DeltaN6 Variant. Biomolecules. 2019;9(8). Epub 2019/08/17. doi: 10.3390/biom9080366. PubMed PMID: 31416179.
237. Natalello A, Mangione PP, Giorgetti S, Porcari R, Marchese L, Zorzoli I, et al. Co-fibrillogenesis of Wild-type and D76N beta2-Microglobulin: THE CRUCIAL ROLE OF FIBRILLAR SEEDS. The Journal of biological chemistry. 2016;291(18):9678-89. Epub 2016/02/28. doi: 10.1074/jbc.M116.720573. PubMed PMID: 26921323; PubMed Central PMCID: PMC4850305.
238. van Zundert GC, Rodrigues JP, Trellet M, Schmitz C, Kastritis PL, Karaca E, et al. The HADDOCK2.2 Web Server: User-Friendly Integrative Modeling of Biomolecular Complexes. Journal of molecular biology. 2016;428(4):720-5. Epub 2015/09/28. doi: 10.1016/j.jmb.2015.09.014. PubMed PMID: 26410586.
239. Karamanos TK, Kalverda AP, Thompson GS, Radford SE. Visualization of transient protein-protein interactions that promote or inhibit amyloid assembly. Molecular cell. 2014;55(2):214-26. Epub 2014/07/02. doi: 10.1016/j.molcel.2014.05.026. PubMed PMID: 24981172; PubMed Central PMCID: PMC4104025.

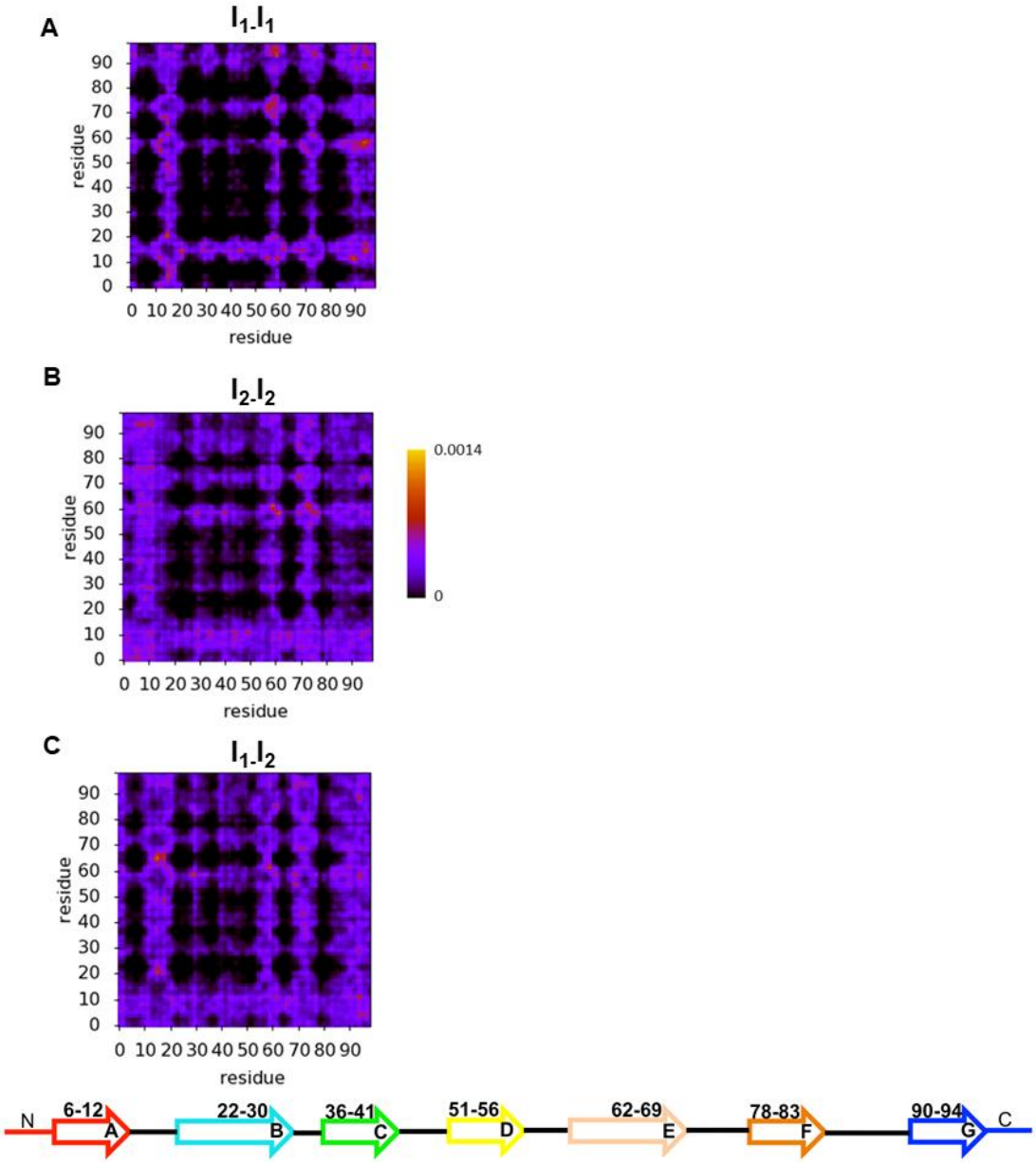
240. Jadoul M, Garbar C, Noel H, Sennesael J, Vanholder R, Bernaert P, et al. Histological prevalence of beta 2-microglobulin amyloidosis in hemodialysis: a prospective post-mortem study. Kidney international. 1997;51(6):1928-32. Epub 1997/06/01. PubMed PMID: 9186884.
241. Motomiya Y, Ando Y, Haraoka K, Sun X, Morita H, Amano I, et al. Studies on unfolded β2-microglobulin at C-terminal in dialysis-related amyloidosis. Kidney Int. 2005;67(1):314–20. doi: 10.1111/j.1523-1755.2005.00084.x.
242. McParland VJ, Kad NM, Kalverda AP, Brown A, Kirwin-Jones P, Hunter MG, et al. Partially unfolded states of beta(2)-microglobulin and amyloid formation in vitro. Biochemistry. 2000;39(30):8735-46. Epub 2000/07/29. PubMed PMID: 10913285.
243. McParland VJ, Kalverda AP, Homans SW, Radford SE. Structural properties of an amyloid precursor of beta(2)-microglobulin. Nature structural biology. 2002;9(5):326-31. Epub 2002/04/23. doi: 10.1038/nsb791. PubMed PMID: 11967566.
244. Mompean M, Chakrabartty A, Buratti E, Laurents DV. Electrostatic Repulsion Governs TDP-43 C-terminal Domain Aggregation. PLoS biology. 2016;14(4):e1002447. Epub 2016/04/21. doi: 10.1371/journal.pbio.1002447. PubMed PMID: 27096426; PubMed Central PMCID: PMC4838238.
245. Zou Y, Sun Y, Zhu Y, Ma B, Nussinov R, Zhang Q. Critical Nucleus Structure and Aggregation Mechanism of the C-terminal Fragment of Copper-Zinc Superoxide Dismutase Protein. ACS chemical neuroscience. 2016;7(3):286-96. Epub 2016/01/28. doi: 10.1021/acschemneuro.5b00242. PubMed PMID: 26815332.
246. Jarrett JT, Berger EP, Jr. LP. The C-terminus of the beta protein is critical in amyloidogenesis. Ann N Y Acad Sci. 1993;695:144-8. doi: 10.1111/j.1749-6632.1993.tb23043.x.
247. Zheng X, Jia L, Hu B, Sun Y, Zhang Y, Gao X, et al. The C-terminal amyloidogenic peptide contributes to self-assembly of Avibirnavirus viral protease. Scientific reports. 2015;5:14794. Epub 2015/10/07. doi: 10.1038/srep14794. PubMed PMID: 26440769; PubMed Central PMCID: PMC4594098.
248. Patino MM, Liu JJ, Glover JR, Lindquist S. Support for the prion hypothesis for inheritance of a phenotypic trait in yeast. Science. 1996;273(5275):622-6. Epub 1996/08/02. PubMed PMID: 8662547.
249. Beland M, Roucou X. The prion protein unstructured N-terminal region is a broad-spectrum molecular sensor with diverse and contrasting potential functions. Journal of neurochemistry. 2012;120(6):853-68. Epub 2011/12/08. doi: 10.1111/j.1471-4159.2011.07613.x. PubMed PMID: 22145935.
250. Smaoui MR, Mazza-Anthony C, Waldispuhl J. Investigating Mutations to Reduce Huntingtin Aggregation by Increasing Htt-N-Terminal Stability and Weakening Interactions with PolyQ Domain. Computational and mathematical methods in medicine. 2016;2016:6247867. Epub 2017/01/18. doi: 10.1155/2016/6247867. PubMed PMID: 28096892; PubMed Central PMCID: PMC5206856.
251. Baias M, Smith PE, Shen K, Joachimiak LA, Zerko S, Kozminski W, et al. Structure and Dynamics of the Huntingtin Exon-1 N-Terminus: A Solution NMR Perspective. Journal of the American Chemical Society. 2017;139(3):1168-76. Epub 2017/01/14. doi: 10.1021/jacs.6b10893. PubMed PMID: 28085263.
252. Tatini F, Pugliese AM, Traini C, Niccoli S, Maraula G, Ed Dami T, et al. Amyloid-beta oligomer synaptotoxicity is mimicked by oligomers of the model protein HypF-N. Neurobiology of aging. 2013;34(9):2100-9. Epub 2013/04/23. doi: 10.1016/j.neurobiolaging.2013.03.020. PubMed PMID: 23601807.
253. Campioni S, Mannini B, Zampagni M, Pensalfini A, Parrini C, Evangelisti E, et al. A causative link between the structure of aberrant protein oligomers and their toxicity. Nature chemical biology. 2010;6(2):140-7. Epub 2010/01/19. doi: 10.1038/nchembio.283. PubMed PMID: 20081829.
254. Park S, Schulten K. Calculating potentials of mean force from steered molecular dynamics simulations. The Journal of chemical physics. 2004;120(13):5946-61. Epub 2004/07/23. doi: 10.1063/1.1651473. PubMed PMID: 15267476.
255. Poger D, Mark AE. On the Validation of Molecular Dynamics Simulations of Saturated and cis-Monounsaturated Phosphatidylcholine Lipid Bilayers: A Comparison with Experiment. J Chem Theory Comput. 2010;6(1):325-36. Epub 2010/01/12. doi: 10.1021/ct900487a. PubMed PMID: 26614341.

Supplementary information

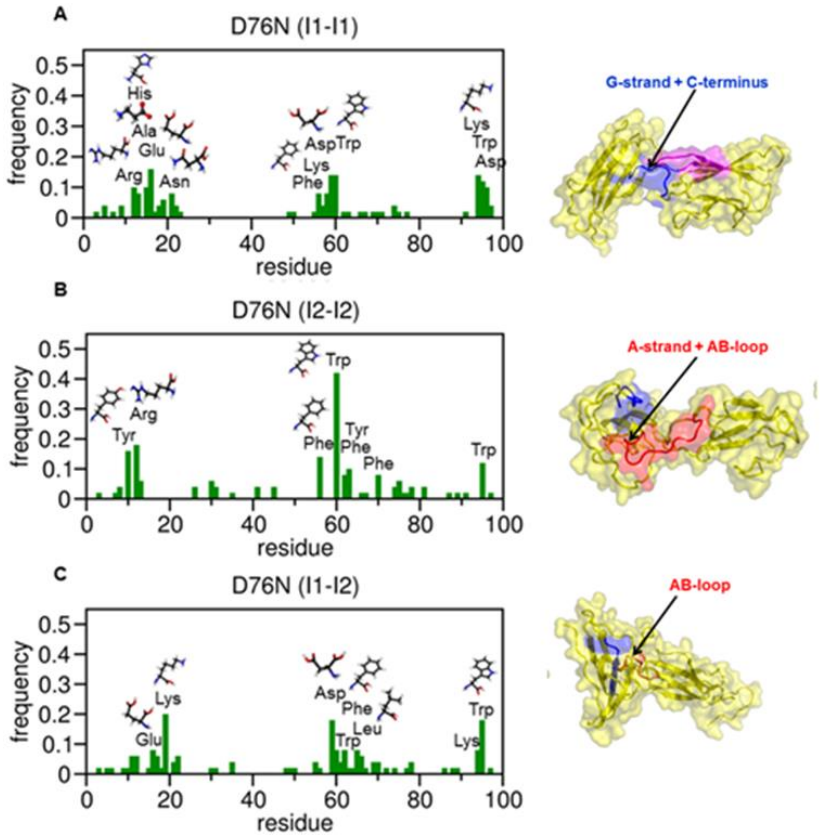
Residue	Atom	SASA
Ala	CB	71.9
Arg	CB	25.9
	CG	23.9
	CD	29.6
	NE	13.3
	CZ	2.2
	NH1	58.6
	NH2	63.4
Asn	CB	32.1
	CG	3.0
	OD1	31.1
	ND2	59.1
Asp	CB	33.5
	CG	5.1
	OD1	38.7
	OD2	40.9
Cys	CB	38.5
	SG	65.0
Gln	CB	26.6
	CG	30.9
	CD	3.7
	OE1	34.2
	NE2	60.0
Glu	CB	26.8
	CG	33.2
	CD	5.5
	OE1	41.4
	OE2	41.5
His	CB	34.0
	CG	1.6
	ND1	11.4
	CE1	51.6
	NE2	30.5
	CD2	33.0
Ile	CB	11.0
	CG1	37.0
	CG2	58.4
	CD1	43.7
Leu	CB	23.1
	CG	9.6
	CD1	63.4
	CD2	61.7

Lys	CB	25.9
	CG	22.6
	CD	28.0
	CE	36.0
	NZ	74.6
Met	CB	24.0
	CG	29.6
	SD	36.4
	CE	74.8
Phe	CB	29.3
	CG	0.6
	CD1	22.1
	CE1	36.7
	CZ	37.7
	CE2	36.1
	CD2	21.9
Pro	CB	39.1
	CG	44.3
	CD	27.6
Ser	CB	44.3
	OG	41.5
Thr	CB	18.0
	OG1	33.5
	CG2	63.1
Trp	CB	30.1
	CG	1.3
	CD1	30.2
	NE1	29.8
	CE2	3.3
	CZ2	38.5
	CH2	37.9
	CZ3	34.1
	CE3	21.5
	CD2	2.2
Tyr	CB	29.2
	CG	0.4
	CD1	22.6
	CE1	34.8
	CZ	2.6
	OH	52.9
	CE2	34.1
	CD2	21.5
Val	CB	11.1
	CG1	57.4
	CG2	59.9

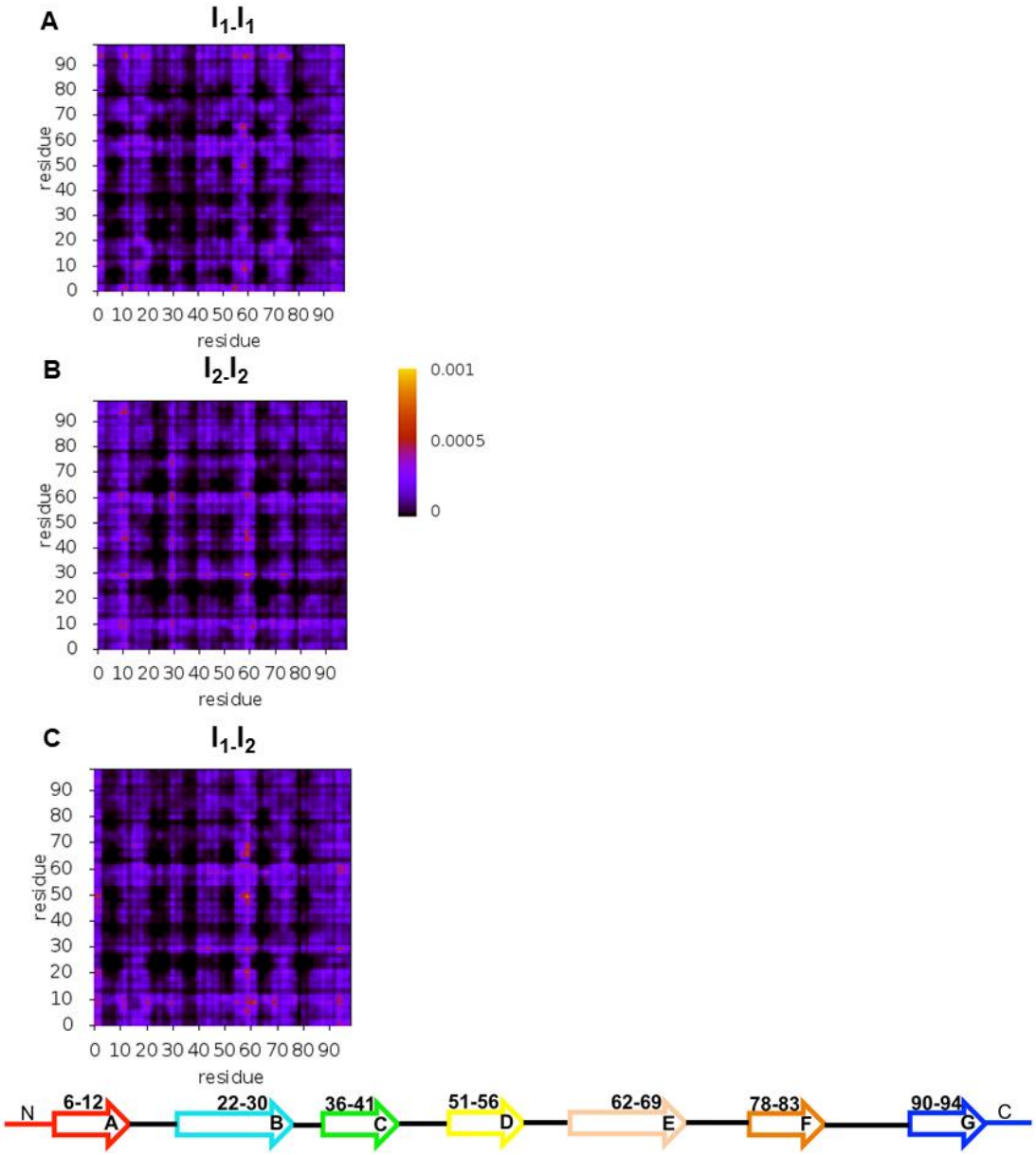
Supplementary Table 1. Solvent accessible surface areas of the different atom types in proteins obtained by Lesser *et al.* (226) used in the calculation of the hydropathic energies in the MC-ED docking protocol.



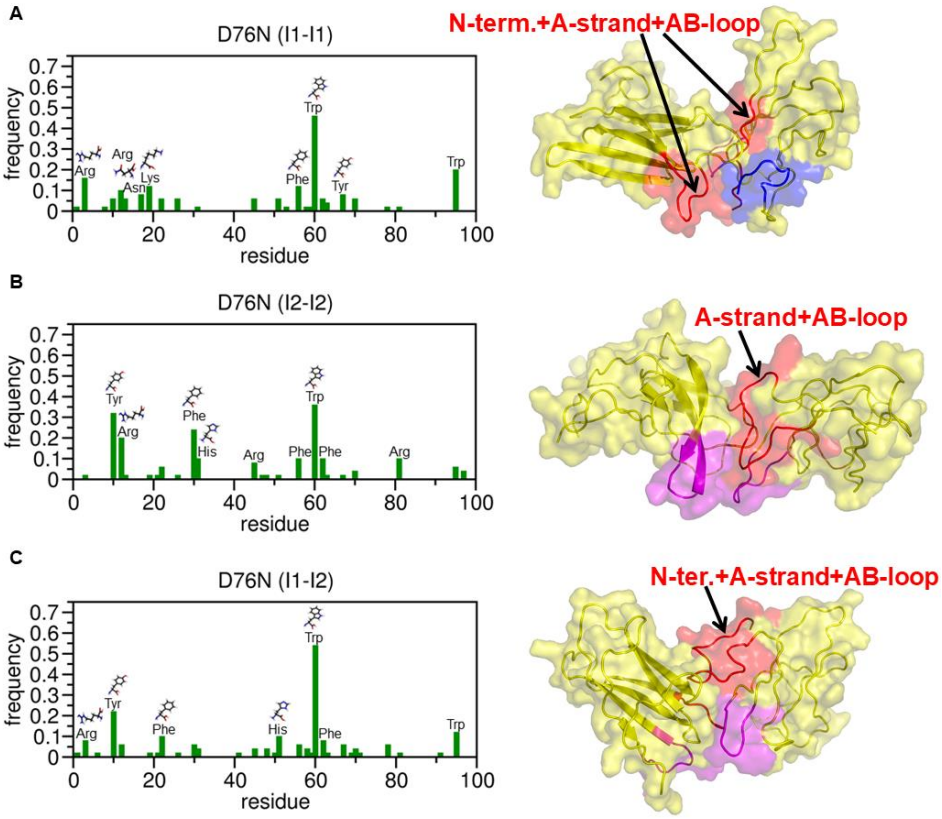
Supplementary Figure 1. Probability maps for the intermolecular contacts established between monomers of different dimer structures of D76N at pH 6.2 produced with the original version of the MC-ED.



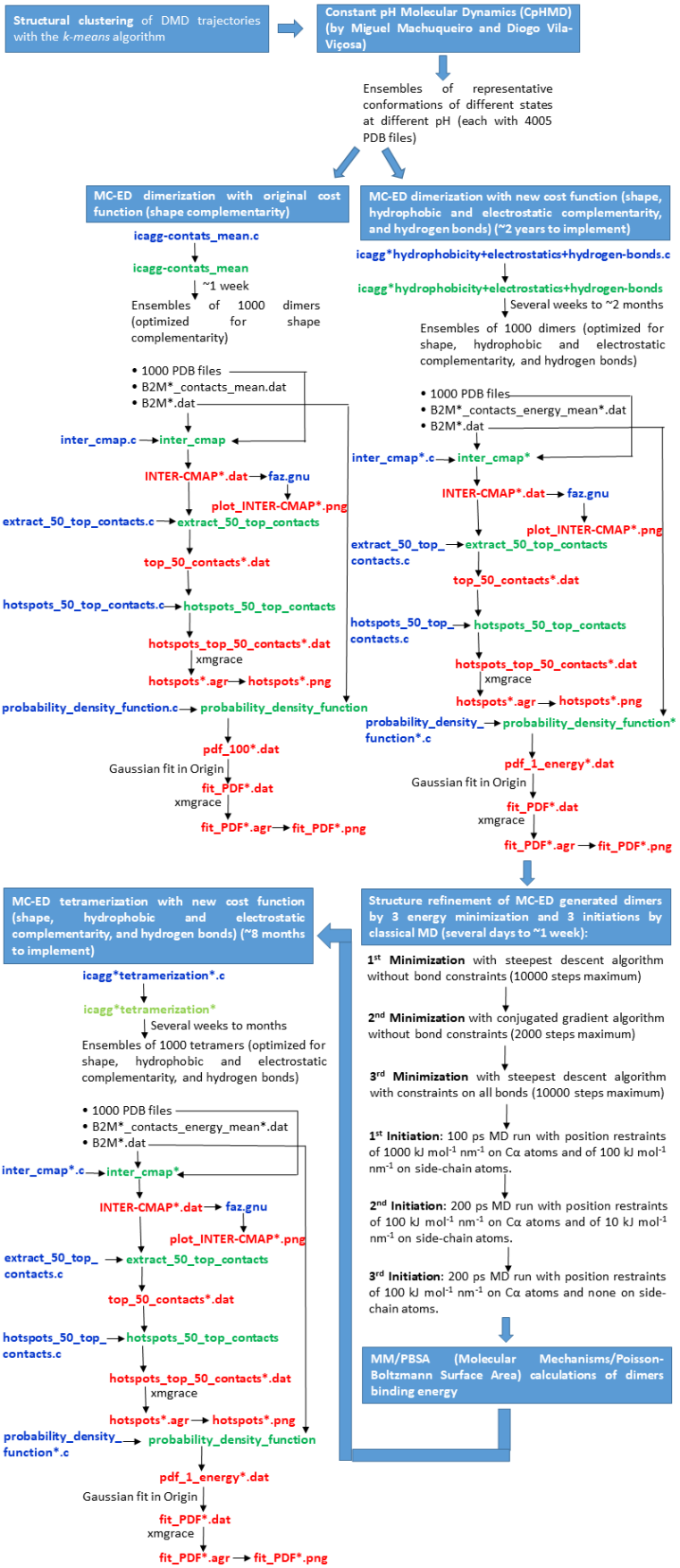
Supplementary Figure 2. Dimerization hot-spots (left) and three-dimensional structures (right) of representative conformations of I_1-I_1 and I_2-I_2 homodimers, and I_1-I_2 heterodimers of D76N at pH 6.2 obtained with the original version of the MC-ED.



Supplementary Figure 3. Probability maps for the intermolecular contacts established between monomers of different dimer structures of D76N at pH 6.2 produced with the novel version of the MC-ED.



Supplementary Figure 4. Dimerization hot-spots (left) and three-dimensional structures (right) of representative conformations of I_1-I_1 and I_2-I_2 homodimers, and I_1-I_2 heterodimers of D76N at pH 6.2 obtained with the novel version of the MC-ED.



Supplementary Figure 5. Representation of the operational steps involved in the main stages of the methodological approach followed in this Project.

MC-ED dimerization code with new cost function

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>
#include <unistd.h>
#include <sys/stat.h>
#include <time.h>
//=============Parameters you might want to change===============


#define MCSTEPS                 (2000)
#define PROTSTEP                (100)

#define STRUCTS                 (4005)
#define PAIRS                   (1000)
#define CONTACT_PAIR            (2)
#define SCALE                   (1.)            //d(CM1−CM2)=SCALE*(Rg1+Rg2)
#define SCALE_1                 (1.2)           //scale for adjusting CM−distanc
e in the initial pose/orientation sampling
#define SCALE_2                 (0.9)           //scale for adjusting CM−distanc
e in the initial pose/orientation sampling

#define SCALE_CLASHES           (0.5)

#define HARDCORE_R              (1.)
#define CONTACT_R               (2.33)

#define Tm                      (0.15)

//Tm = temperature at which structures were sampled
//ensemble sizes: NAT(WT)pH5.2=4005
//ensemble sizes: NAT(WT)pH6.2=4005
//ensemble sizes: NAT(WT)pH7.2=4005
//ensemble sizes: NAT(D76N)pH5.2=4005
//ensemble sizes: NAT(D76N)pH6.2=4005
//ensemble sizes: NAT(D76N)pH7.2=4005
//ensemble sizes: I1(D76N)pH5.2=4005
//ensemble sizes: I1(D76N)pH6.2=4005
//ensemble sizes: I1(D76N)pH7.2=4005
//ensemble sizes: I2(D76N)pH5.2=4005
//ensemble sizes: I2(D76N)pH6.2=4005
//ensemble sizes: I2(D76N)pH7.2=4005

//end to parameters you might want to change=====================

#define MAXLENLINE      (256)
#define LENCHARGESFILE  (287)
#define MAXRESIDUE      (99)
#define MAXATOMNUM      (1100)
#define MAXCONTACT      (50000)
#define HIS_TYPE_HIS    (0)
#define HIS_TYPE_HSD    (1)
#define HIS_TYPE_HSC    (2)
#define NUM_AA_TYPE     (26)
#define NUM_Atom_TYPE   (32)
#define PI              (3.14159265358979323846)


//=============Directories you might want to change============================
============================================================
char workingdir[]="/home/rjloureiro/DOCKING/B2M−D76N−I1−PH7p2−newcf/";
char writingdir[]="/home/rjloureiro/DOCKING/DOCKING_B2M_D76N−I1−PH7p2_newcf_n−clashes_0.5_
energy_hydropathy+electrostatics+hydrogen−bonds/";
char rootdir[]="/home/rjloureiro";
char ss[]="DOCKING_B2M_D76N−I1−PH7p2_newcf_n−clashes_0.5_energy_hydropathy+electrostatics+hydro
gen−bonds";
//============================================================================
============================================================
```



```
int ResNum[2];
int nAtom[2];

typedef struct  {

        int Res_Type;
        char AtmName[5];
        char Atm_Type[5];
        float Atm_Charge;

}MYCHARGES,*PMYCHARGES;

typedef struct  {

        double x,y,z;
        double r0;

        int NodeIndex;
        int Res_Type;
        int AtomIndex;
        int Atom_Type;
        int ASP_Type;
        char AtmName[5];
        char AtmHBlabel[5];

}MYATOM,*PMYATOM;

typedef struct  {

        double Rg;
        double RMSD;
        int contEn;

}MYCONF, *PMYCONF;

typedef struct  {

        double r01;
        double r02;
        double rsq;

}MYDISTANCES, *PMYDISTANCES;

double radii[]={1.61, 1.76, 1.88, 1.88, 1.88, 1.64, 1.64, 1.64, 1.64, 1.42, 1.46
, 1.77, 1.77, 1.05, 0.58};

char AA_Name[NUM_AA_TYPE][5]={"METN","ILE","ILEN","VAL","LEU","PHE","CYS","MET
","METC","ALA","GLY","THR","SER","TRP","TYR","PRO","HIS","HISH","GLN",
"ASN","GLU","GLUH","ASP","ASPH","LYS","ARG"};/*check*/

float atomic_solvation_parameters[6]={18.0, −7.0, 18.0, −20.0, −34.0, 0.0};

char atom_name[NUM_Atom_TYPE][4]={"CB","CG","CG1","CG2","CD","CD1","CD2","CE",
"CE1","CE2","CE3","CH2","CZ","CZ2","CZ3","ND1","ND2","NE","NE1","NE2",
"NH1","NH2","NZ","OD1","OD2","OE1","OE2","OG","OG1","OH","SG","SD"};

float atomic_SASA[NUM_AA_TYPE][NUM_Atom_TYPE]={
{24.0,  29.6, 0.0,  0.0,  0.0,  0.0,  0.0,  74.8, 0.0,  0.0,  0.0,  0.0,  0.0,
0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.
0,  0.0,  0.0,  0.0,  0.0,  36.4},
{11.0,  0.0,  37.0, 58.4, 0.0,  43.7, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,
0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.
0,  0.0,  0.0,  0.0,  0.0,  0.0},
{11.0,  0.0,  37.0, 58.4, 0.0,  43.7, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,
0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.
0,  0.0,  0.0,  0.0,  0.0,  0.0},
{11.1,  0.0,  57.4, 59.9, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,
0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.
```

```
0,  0.0,  0.0,  0.0,  0.0,  0.0},
{23.1,  9.6, 0.0,  0.0,  0.0,  63.4, 61.7, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0
.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0
,  0.0,  0.0,  0.0,  0.0,  0.0},
{29.3,  0.6,  0.0,  0.0,  0.0,  22.1, 21.9, 0.0,  36.7, 36.1, 0.0,  0.0,  37.7,
0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.
0,  0.0,  0.0,  0.0,  0.0,  0.0},
{38.5,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,
0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.
0,  0.0,  0.0,  0.0,  65.0, 0.0},
{24.0,  29.6, 0.0,  0.0,  0.0,  0.0,  0.0,  74.8, 0.0,  0.0,  0.0,  0.0,  0.0,
0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.
0,  0.0,  0.0,  0.0,  0.0,  36.4},
{24.0,  29.6, 0.0,  0.0,  0.0,  0.0,  0.0,  74.8, 0.0,  0.0,  0.0,  0.0,  0.0,
0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.
0,  0.0,  0.0,  0.0,  0.0,  36.4},
{71.9, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0
.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0
,  0.0,  0.0,  0.0,  0.0,  0.0},
{0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,
0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.
0,  0.0,  0.0,  0.0,  0.0,  0.0},
{18.0,  0.0,  0.0,  63.1, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,
0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.
0,  0.0,  33.5, 0.0,  0.0,  0.0},
{44.3, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0
.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0
,  41.5, 0.0,  0.0,  0.0,  0.0},
{30.1,  1.3,  0.0,  0.0,  0.0,  30.2, 2.2,  0.0,  0.0,  3.3,  21.5, 37.9, 0.0,
38.5, 34.1, 0.0,  0.0,  0.0,  29.8, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.
0,  0.0,  0.0,  0.0,  0.0,  0.0},
{29.2,  0.4,  0.0,  0.0,  0.0,  22.6, 21.5, 0.0,  34.8, 34.1, 0.0,  0.0,  2.6, 0
.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0
,  0.0,  0.0,  52.9, 0.0,  0.0},
{39.1,   44.3,  0.0,  0.0,  27.6,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0
,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,
 0.0,  0.0,  0.0,  0.0,  0.0,  0.0},
{34.0,  1.6,  0.0,  0.0,  0.0,  0.0,  33.0, 0.0,  51.6, 0.0,  0.0,  0.0,  0.0,
0.0,  0.0,  11.4, 0.0,  0.0,  0.0,  30.5, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.
0,  0.0,  0.0,  0.0,  0.0,  0.0},
{34.0,  1.6,  0.0,  0.0,  0.0,  0.0,  33.0, 0.0,  51.6, 0.0,  0.0,  0.0,  0.0,
0.0,  0.0,  11.4, 0.0,  0.0,  0.0,  30.5, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.
0,  0.0,  0.0,  0.0,  0.0,  0.0},
{26.6,  30.9, 0.0,  0.0,  3.7, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0
.0,  0.0,  0.0,  0.0,  0.0,  0.0,  60.0, 0.0,  0.0,  0.0,  0.0,  0.0,  34.2, 0.0
,  0.0,  0.0,  0.0,  0.0,  0.0},
{32.1,  3.0, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0
.0,  0.0,  0.0,  59.1, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  31.1, 0.0,  0.0,  0.0
,  0.0,  0.0,  0.0,  0.0,  0.0},
{26.8,  33.2, 0.0,  0.0,  5.5, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0
.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  41.4, 41.
5, 0.0,  0.0,  0.0,  0.0,  0.0},
{26.8,  33.2, 0.0,  0.0,  5.5, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0
.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  41.4, 41.
5, 0.0,  0.0,  0.0,  0.0,  0.0},
{33.5,  5.1, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0
.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  38.7, 40.9, 0.0,  0.0
,  0.0,  0.0,  0.0,  0.0,  0.0},
{33.5,  5.1, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0
.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  38.7, 40.9, 0.0,  0.0
,  0.0,  0.0,  0.0,  0.0,  0.0},
{25.9,  22.6, 0.0,  0.0,  28.0, 0.0,  0.0,  36.0, 0.0,  0.0,  0.0,  0.0,  0.0,
0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  74.6, 0.0,  0.0,  0.0,  0.
0,  0.0,  0.0,  0.0,  0.0,  0.0},
{25.9,  23.9, 0.0,  0.0,  29.6, 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  2.2, 0
.0,  0.0,  0.0,  0.0,  13.3, 0.0,  0.0,  58.6, 63.4, 0.0,  0.0,  0.0,  0.0,  0.0
,  0.0,  0.0,  0.0,  0.0,  0.0},
};/*atomic solvent accessible surface area*/
```



```

MYCHARGES Charges[LENCHARGESFILE];

MYATOM Atoms[MAXATOMNUM][2];

MYATOM Moved[MAXATOMNUM];

MYATOM RotMoved[MAXATOMNUM];

MYATOM stack1[MAXATOMNUM];

MYATOM stack2[MAXATOMNUM];

MYCONF Conf[2];
MYDISTANCES dist;

double MEAN_R[3][2];

float q0[MAXCONTACT];
float q1[MAXCONTACT];
int it[MAXCONTACT];
double rd[MAXCONTACT];
int ct0[MAXCONTACT];
int ct1[MAXCONTACT];
int at0[MAXCONTACT];
int at1[MAXCONTACT];
int ri0[MAXCONTACT];
int ri1[MAXCONTACT];
double ro0[MAXCONTACT];
double ro1[MAXCONTACT];
int asp0[MAXCONTACT];
int asp1[MAXCONTACT];
double sum_vdW[MAXCONTACT];
double hidropathy_0[MAXCONTACT];
double hidropathy_1[MAXCONTACT];
double hp[MAXCONTACT];
double ep[MAXCONTACT];
double hbp[MAXCONTACT];
double ip[MAXCONTACT];


//CODE PARTS
int main(void);
void READER(int a, int b);
void CENTRALIZER(double a);
void INITIALORIENTATOR(void);
int TYPECONTACTS(int ct0[MAXCONTACT], int ct1[MAXCONTACT], int at0[MAXCONTACT],
int at1[MAXCONTACT], int ri0[MAXCONTACT], int ri1[MAXCONTACT], double ro0[MAXCON
TACT], double ro1[MAXCONTACT], double rd[MAXCONTACT], int it[MAXCONTACT], float
q0[MAXCONTACT], float q1[MAXCONTACT], int asp0[MAXCONTACT], int asp1[MAXCONTACT]
);
int CONTACTS(void);
int CLASHES(void);
int TYPECONTACTSMOVED(int ct0[MAXCONTACT], int ct1[MAXCONTACT], int at0[MAXCONTA
CT], int at1[MAXCONTACT], int ri0[MAXCONTACT], int ri1[MAXCONTACT], double ro0[M
AXCONTACT], double ro1[MAXCONTACT], double rd[MAXCONTACT], int it[MAXCONTACT], f
loat q0[MAXCONTACT], float q1[MAXCONTACT], int asp0[MAXCONTACT], int asp1[MAXCON
TACT]);
int CONTACTSMOVED(void);
int CLASHESMOVED(void);
void MCMOVE(double a);
void READCHARGES(void);


//PDB HANDLING
double abbs(double a);
int IsANumber(char c);
int Get_AA_Type(char SzAAName[]);
int Get_Atom_Type(char SzAtomName[]);
```

```
void AssignAtomRadii(void);
int QueryAtomType(char *s, char *res);
void ExportSnapshot(int Index1, int Index2);


//RANDOM NUMBER GENERATOR HEADER
void RandomInitialise(int ij,int kl);
double RandomUniform(void);
double RandomGaussian(double mean,double stddev);
int RandomInt(int lower,int upper);
double RandomDouble(double lower,double upper);


double restdist=0;
double currentdist=0;
double olddist=0;
double mcinternal1=0.025;                       //perturbation amplitude for tra
nslations
double mcinternal2=0.025;                       //perturbation amplitude for rot
ations
double TOL=0.2;                                 //tolerance of CM−diffusion in p
ercent of d(CM1−CM2)
double TOL0=0.2;
double x = 1.3;


int DIR;

//      COMMENT on DIR
//      direction of docking:
//      +1 ...  positive x−axis
//      −1 ...  negative x−axis
//      +2 ...  positive y−axis
//      −2 ...  negative y−axis
//      +3 ...  positive z−axis
//      −3 ...  negative z−axis

int ORIENTATION;

//      COMMENT on ORIENTATION
//      initial orientation of docking:
//      0  ...  no rotation
//      +1 ...  +90º rotation arround x−axis
//      −1 ...  −90º rotation arround x−axis
//      +2 ...  +90º rotation arround y−axis
//      −2 ...  −90º rotation arround y−axis
//      +3 ...  +90º rotation arround z−axis
//      −3 ...  −90º rotation arround z−axis

int main(void)
{

        FILE *output1;
        FILE *output2;

        int i,j,k,l,m,o,p,e,mc,helper1,helper2,index,ncontacts,N,contact_flag,re
stdistchange1,restdistchange2;
        int newrestdist1,newrestdist2;
        int n = 2;
        int cc,clashinit[6][7][3],contactinit[6][7][3],hitwall;
        int hydropathic_flag;
        int residue_index_0;
        int residue_index_1;
        int accepted, rejected;
        int initial_counter;
        int final_counter;
        int counter_before_mcmove;
        int counter_after_mcmove;
        double SASA_AA[MAXRESIDUE];
```



```
        char oo[256], pp[256], qq[256], rr[256], s[256], tt[256], uu[256], vv[25
6];
        double shapeinit[6][7][3];
        double Etransl, Econtnew, Etranslold, Econtold,Eallold,Eallnew;
        double contactnew, contactold;
        double initclash,clashnew,clashold;
        double Pacc,Pdecl;
        double dummie;
        double lastTOL;
        int CONTACTS_SUM = 0;
        double CONTACTS_MEAN;
        double contacts_diff_sum = 0.0;
        double VARIANCE_CONTACTS;
        double VARIANCE_CONTACTS_MEAN;
        int CLASHES_SUM = 0;
        double CLASHES_MEAN;
        double ENERGY_SUM = 0.0;
        double ENERGY_MEAN;
        double energy_diff_sum = 0.0;
        double VARIANCE_ENERGY;
        double VARIANCE_ENERGY_MEAN;
        double initial_energy_sum = 0.0;
        double initial_hp_energy_sum = 0.0;
        double initial_ep_energy_sum = 0.0;
        double initial_hbp_energy_sum = 0.0;
        double energy_sum_before_mcmove = 0.0;
        double energy_sum_after_mcmove = 0.0;
        double final_energy_sum = 0.0;
        double final_hp_energy_sum = 0.0;
        double final_ep_energy_sum = 0.0;
        double final_hbp_energy_sum = 0.0;


        int CCONTACTS[PAIRS];
        int CCLASHES[PAIRS];
        double ENERGY[PAIRS];


        mkdir(ss,457);

        cc=time(NULL);
        RandomInitialise(cc%20000,cc%10000);
        READCHARGES();

        accepted=0;
        rejected=0;

        chdir(writingdir);


        sprintf(oo,"B2M_D76N−I1−PH7p2_2000_T−0.15_newcf_n−clashes_0.5_energy_hydropathy+electr
ostatics+hydrogen−bonds.dat");
        output1=fopen(oo,"a");

        sprintf(pp,"B2M_D76N−I1−PH7p2_contacts_energy_mean_2000_T−0.15_newcf_n−clashes_0.5_en
ergy_hydropathy+electrostatics+hydrogen−bonds.dat");
        output2=fopen(pp,"a");


        chdir(workingdir);


        for (k=1;k<=PAIRS;k++)
        {

                        chdir(workingdir);

                        //PICKING TWO STRUCTURE INDICES
                        i=RandomInt(1,STRUCTS);
```

```
                        j=RandomInt(1,STRUCTS);

                        READER(i,j);

                        chdir(writingdir);

                        index=0;

                        //restdist=SCALE*(Conf[0].Rg + Conf[1].Rg);

                        restdist=31.95; //SCALE*(2*Rg(D76N−I1))

                        hitwall=0;

                        //BEGINNING of INITIAL DIRECTION/ORIENTATION sampling

                        for (DIR=−3;DIR<=−1;DIR++)
                        {
                                for(ORIENTATION=−3;ORIENTATION<=−1;ORIENTATION++
)
                                {
                                        CENTRALIZER(restdist);
                                        clashinit[DIR+3][ORIENTATION+3][0] = CLA
SHES();
                                        contactinit[DIR+3][ORIENTATION+3][0] = −
CONTACTS();
                                        clashinit[DIR+3][ORIENTATION+3][1]=DIR;
                                        contactinit[DIR+3][ORIENTATION+3][1]=DIR
;
                                        clashinit[DIR+3][ORIENTATION+3][2]=ORIEN
TATION;
                                        contactinit[DIR+3][ORIENTATION+3][2]=ORI
ENTATION;
                                        chdir(workingdir);
                                        READER(i,j);
                                        chdir(writingdir);

                                }
                                for(ORIENTATION=1;ORIENTATION<=4;ORIENTATION++)
                                {
                                        CENTRALIZER(restdist);
                                        clashinit[DIR+3][ORIENTATION+2][0] = CLA
SHES();
                                        contactinit[DIR+3][ORIENTATION+2][0] = −
CONTACTS();
                                        clashinit[DIR+3][ORIENTATION+2][1]=DIR;
                                        contactinit[DIR+3][ORIENTATION+2][1]=DIR
;
                                        clashinit[DIR+3][ORIENTATION+2][2]=ORIEN
TATION;
                                        contactinit[DIR+3][ORIENTATION+2][2]=ORI
ENTATION;
                                        chdir(workingdir);
                                        READER(i,j);
                                        chdir(writingdir);



                                }
                        }
                        for (DIR=1;DIR<=3;DIR++)
                        {
                                for(ORIENTATION=−3;ORIENTATION<=−1;ORIENTATION++
)
                                {
                                        CENTRALIZER(restdist);
                                        clashinit[DIR+2][ORIENTATION+3][0] = CLA
SHES();
                                        contactinit[DIR+2][ORIENTATION+3][0] = −
CONTACTS();
```



```
                                        clashinit[DIR+2][ORIENTATION+3][1]=DIR;
                                        contactinit[DIR+2][ORIENTATION+3][1]=DIR
;
                                        clashinit[DIR+2][ORIENTATION+3][2]=ORIEN
TATION;
                                        contactinit[DIR+2][ORIENTATION+3][2]=ORI
ENTATION;
                                        chdir(workingdir);
                                        READER(i,j);
                                        chdir(writingdir);



                                }
                                for(ORIENTATION=1;ORIENTATION<=4;ORIENTATION++)
                                {
                                        CENTRALIZER(restdist);
                                        clashinit[DIR+2][ORIENTATION+2][0] = CLA
SHES();
                                        contactinit[DIR+2][ORIENTATION+2][0] = −
CONTACTS();
                                        clashinit[DIR+2][ORIENTATION+2][1]=DIR;
                                        contactinit[DIR+2][ORIENTATION+2][1]=DIR
;
                                        clashinit[DIR+2][ORIENTATION+2][2]=ORIEN
TATION;
                                        contactinit[DIR+2][ORIENTATION+2][2]=ORI
ENTATION;
                                        chdir(workingdir);
                                        READER(i,j);
                                        chdir(writingdir);



                                }
                        }

                        DIR=0;
                        ORIENTATION=0;
                        dummie=0;
                        for (o=0;o<6;o++)
                        {
                                for (p=0;p<7;p++)
                                {
                                        restdistchange1=0;
                                        restdistchange2=0;

                                        //INITIAL OPTIMIZATION OF CLASHES AND CO
NTACTS

                                        if (clashinit[o][p][0]>=200)
                                        {
                                                DIR=contactinit[o][p][1];
                                                ORIENTATION=contactinit[o][p][2]
;

                                                restdist=SCALE_1*restdist;
                                                CENTRALIZER(restdist);
                                                restdistchange1=1;
                                                contactinit[o][p][0] = −CONTACTS
();
                                                clashinit[o][p][0] = CLASHES();
                                                chdir(workingdir);
                                                READER(i,j);
                                                chdir(writingdir);
                                                restdist=31.95;

                                        }
                                        else if (contactinit[o][p][0]>=−3000)
                                        {
                                                DIR=contactinit[o][p][1];
                                                ORIENTATION=contactinit[o][p][2]
```

```
;
                                        restdist=SCALE_2*restdist;
                                        CENTRALIZER(restdist);
                                        restdistchange2=1;
                                        contactinit[o][p][0] = −CONTACTS
();
                                        clashinit[o][p][0] = CLASHES();
                                        chdir(workingdir);
                                        READER(i,j);
                                        chdir(writingdir);
                                        restdist=31.95;

                                }

                                printf("%d      %d      %d\n",contactini
t[o][p][1],contactinit[o][p][2],contactinit[o][p][0]);


                                if (contactinit[o][p][0]<dummie)
                                {
                                        dummie=contactinit[o][p][0];
                                        if(restdistchange1)
                                        {
                                                newrestdist1=1;
                                        }
                                        else
                                        {
                                                newrestdist1=0;
                                        }
                                        if(restdistchange2)
                                        {
                                                newrestdist2=1;
                                        }
                                        else
                                        {
                                                newrestdist2=0;
                                        }
                                }
                        }

                        printf("\n\n");
                        for (o=0;o<6;o++)
                        {
                                for (p=0;p<7;p++)
                                {
                                        if (dummie==contactinit[o][p][0])
                                        {
                                                DIR=contactinit[o][p][1];
                                                ORIENTATION=contactinit[o][p][2]
;
                                                break;
                                        }
                                }
                        }

                        if(newrestdist1==1)
                        {
                                restdist=SCALE_1*restdist;
                        }
                        if(newrestdist2==1)
                        {
                                restdist=SCALE_2*restdist;
                        }

                        CENTRALIZER(restdist);
                        currentdist=restdist;
                        //ExportSnapshot(i,j);
                        //sprintf(oo,"MCRUN_I_H_BETA2M_%d_%d_DIR=%d.dat",i,j,(in
```



```
t)DIR);
                        //output1=fopen(oo,"w");

                        //END of INITIAL DIRECTION/ORIENTATION sampling

                        //BEGINNING of the computation of the INITIAL nº of CONT
ACTS, CLASHES and INTERMOLECULAR ENERGY

                        ncontacts = CONTACTS();

                        if (ncontacts > MAXCONTACT)
                        {
                                contact_flag=1;
                                continue;
                        }

                        for (N=0; N<ncontacts; N++)
                        {
                                it[N]=0;
                        }

                        for (N=0; N<MAXATOMNUM; N++)
                        {
                                strcpy(Atoms[N][0].AtmHBlabel,"NoHB");
                                strcpy(Atoms[N][1].AtmHBlabel,"NoHB");
                        }

                        initial_counter = TYPECONTACTS(ct0, ct1, at0, at1, ri0,
ri1, ro0, ro1, rd, it, q0, q1, asp0, asp1);

                        initial_energy_sum = 0.0;
                        initial_hp_energy_sum = 0.0;
                        initial_ep_energy_sum = 0.0;
                        initial_hbp_energy_sum = 0.0;

                        hydropathic_flag = 0;
                        for (m=0; m<initial_counter; m++)
                        {
                                sum_vdW[m] = HARDCORE_R*(ro0[m] + ro1[m]);

                                if (it[m] == 1 || it[m] == 3 || it[m] == 4) //TY
PE OF INTERACTION: 1=electro;3=electro+hidro;4=electro+HB.
                                {
                                        if (rd[m] > sum_vdW[m] & rd[m] < 1.4*sum
_vdW[m]) //1st well of the electrostatic potential
                                        {
                                                if (q0[m] < 0.0 & q1[m] < 0.0)
                                                {
                                                        ep[m] = 0.4;
                                                }
                                                else if (q0[m] > 0.0 & q1[m] > 0
.0)
                                                {
                                                        ep[m] = 0.4;
                                                }
                                                else if (q0[m] < 0.0 & q1[m] > 0
.0)
                                                {
                                                        ep[m] = −0.4;
                                                }
                                                else if (q0[m] > 0.0 & q1[m] < 0
.0)
                                                {
                                                        ep[m] = −0.4;
                                                }
                                                else
                                                {
                                                        ep[m] = 0.0;
                                                }
```

```
                                        }
                                        else if (rd[m] > 1.4*sum_vdW[m] & rd[m]
< 2.33*sum_vdW[m]) //2nd well of the electrostatic potential
                                        {
                                                if (q0[m] < 0.0 & q1[m] < 0.0)
                                                {
                                                        ep[m] = 0.3*0.4;
                                                }
                                                else if (q0[m] > 0.0 & q1[m] > 0
.0)
                                                {
                                                        ep[m] = 0.3*0.4;
                                                }
                                                else if (q0[m] < 0.0 & q1[m] > 0
.0)
                                                {
                                                        ep[m] = −0.3*0.4;
                                                }
                                                else if (q0[m] > 0.0 & q1[m] < 0
.0)
                                                {
                                                        ep[m] = −0.3*0.4;
                                                }
                                                else
                                                {
                                                        ep[m] = 0.0;
                                                }
                                        }
                                        else
                                        {
                                                ep[m] = 0.0;
                                        }
                                        hp[m] = 0.0;
                                        hbp[m] = 0.0;
                                        if ((it[m] == 3) && (rd[m] < 1.6*sum_vdW
[m]))   //TYPE OF INTERACTION: 3=electro+hidro;
                                        {
                                                hidropathy_0[m] = −((atomic_solv
ation_parameters[asp0[m]]*atomic_SASA[ct0[m]][at0[m]])/n)/3000;
                                                hidropathy_1[m] = −((atomic_solv
ation_parameters[asp1[m]]*atomic_SASA[ct1[m]][at1[m]])/n)/3000;

                                                if (hidropathy_0[m] <= −0.053999
5)
                                                {
                                                        hidropathy_0[m] = −0.1;
                                                }
                                                else if ((hidropathy_0[m] >= −0.
0332997) && (hidropathy_0[m] <= 0.0484162))
                                                {
                                                        hidropathy_0[m] = 0.0;
                                                }
                                                else if ((hidropathy_0[m] >= 0.0
617161) && (hidropathy_0[m] <= 0.138332))
                                                {
                                                        hidropathy_0[m] = 0.1;
                                                }
                                                else if (hidropathy_0[m] >= 0.33
2063)
                                                {
                                                        hidropathy_0[m] = 0.4;
                                                }

                                                if (hidropathy_1[m] <= −0.053999
5)
                                                {
                                                        hidropathy_1[m] = −0.1;
                                                }
                                                else if ((hidropathy_1[m] >= −0.
```



```
0332997) && (hidropathy_1[m] <= 0.0484162))
                                                {
                                                        hidropathy_1[m] = 0.0;
                                                }
                                                else if ((hidropathy_1[m] >= 0.0
617161) && (hidropathy_1[m] <= 0.138332))
                                                {
                                                        hidropathy_1[m] = 0.1;
                                                }
                                                else if (hidropathy_1[m] >= 0.33
2063)
                                                {
                                                        hidropathy_1[m] = 0.4;
                                                }

                                                hp[m] = hidropathy_0[m]+hidropat
hy_1[m];
                                        }

                                        if (it[m] == 4)   //TYPE OF INTERACTION:
 4=electro+HB;
                                        {
                                                hbp[m] = −1.0;
                                        }
                                        else
                                        {
                                                hbp[m] = 0.0;
                                        }
                                }
                                else if ((it[m] == 2) && (rd[m] < 1.6*sum_vdW[m]
)) //TYPE OF INTERACTION: 2=hidro.
                                {
                                        hidropathy_0[m] = −((atomic_solvation_pa
rameters[asp0[m]]*atomic_SASA[ct0[m]][at0[m]])/n)/3000;
                                        hidropathy_1[m] = −((atomic_solvation_pa
rameters[asp1[m]]*atomic_SASA[ct1[m]][at1[m]])/n)/3000;

                                        if (hidropathy_0[m] <= −0.0539995)
                                        {
                                                hidropathy_0[m] = −0.1;
                                        }
                                        else if ((hidropathy_0[m] >= −0.0332997)
 && (hidropathy_0[m] <= 0.0484162))
                                        {
                                                hidropathy_0[m] = 0.0;
                                        }
                                        else if ((hidropathy_0[m] >= 0.0617161)
&& (hidropathy_0[m] <= 0.138332))
                                        {
                                                hidropathy_0[m] = 0.1;
                                        }
                                        else if (hidropathy_0[m] >= 0.332063)
                                        {
                                                hidropathy_0[m] = 0.4;
                                        }

                                        if (hidropathy_1[m] <= −0.0539995)
                                        {
                                                hidropathy_1[m] = −0.1;
                                        }
                                        else if ((hidropathy_1[m] >= −0.0332997)
 && (hidropathy_1[m] <= 0.0484162))
                                        {
                                                hidropathy_1[m] = 0.0;
                                        }
                                        else if ((hidropathy_1[m] >= 0.0617161)
&& (hidropathy_1[m] <= 0.138332))
                                        {
                                                hidropathy_1[m] = 0.1;
```

```
                                        }
                                        else if (hidropathy_1[m] >= 0.332063)
                                        {
                                                hidropathy_1[m] = 0.4;
                                        }

                                        hp[m] = hidropathy_0[m]+hidropathy_1[m];

                                        ep[m] = 0.0;
                                        hbp[m] = 0.0;
                                }
                                else
                                {
                                        ep[m] = 0.0;
                                        hp[m] = 0.0;
                                        hbp[m] = 0.0;
                                }
                                ip[m] = ep[m] + hp[m] + hbp[m];
                                initial_energy_sum = initial_energy_sum + ip[m];
                                initial_hp_energy_sum = initial_hp_energy_sum +
hp[m];
                                initial_ep_energy_sum = initial_ep_energy_sum +
ep[m];
                                initial_hbp_energy_sum = initial_hbp_energy_sum
+ hbp[m];
                        }
                        printf("%d       %d       %d       %d       %4.3lf     %4.3lf
%4.3lf     %4.3lf     %4.3lf     %4.3lf     %4.3lf     %d       %d     %d           %d\n",k,i,j,0,(
1−TOL)*restdist,currentdist,(1+TOL0)*restdist,initial_energy_sum,initial_hp_ener
gy_sum,initial_ep_energy_sum,initial_hbp_energy_sum,CONTACTS(),initial_counter,C
LASHES(),DIR);
                        /*fprintf(output7,"%d   %d      %d      %d      %4.3lf
%4.3lf  %4.3lf  %4.3lf  %d       %d       %d\n",k,i,j,0,(1−TOL)*restdist,current
dist,(1+TOL0)*restdist,initial_energy_sum,CONTACTS(),CLASHES(),DIR);*/
                        //fprintf(output1,"%d   %d      %d      %4.3lf  %4.3lf
%4.3lf      %d      %d      %4.3lf\n",i,j,index,currentdist,restdist,initial_ene
rgy_sum,CONTACTS(),CLASHES(),Pacc);
                        //fflush(output1);

                        //END of the computation of the INITIAL nº of CONTACTS,
CLASHES and INTERMOLECULAR ENERGY

                        //BEGINNING OF MONTE CARLO

                        initclash=(double)CLASHES();

                        accepted=0;
                        rejected=0;
                        Pacc=0;
                        Pdecl=0;
                        for (index=1;index<=MCSTEPS;index++)
                        {


                                if (currentdist<=(1+0.01)*(1−TOL)*restdist)
                                {
                                        lastTOL=TOL;
                                        TOL=TOL+0.01;
                                        hitwall++;
                                        if ((1−TOL)*restdist <= 0)
                                                TOL=lastTOL;
                                }


                                //copying information of structure 2 to a stack
variable which will be
                                //subjected to a MC move

                                for (m=0;m<nAtom[1];m++)
```



```
                                {
                                        Moved[m]=Atoms[m][1];
                                }

                                olddist=currentdist;

                                //total energy for l−th Markov state



                                //=== translating + rotating ==================
=================

                                dummie=RandomUniform();
                                MCMOVE(dummie);

                                //=============================================
=================

                                //total energy for possible (l+1)−th Markov stat
e

                                //BEGINNING of the computation of the nº of CONT
ACTS, CLASHES and INTERMOLECULAR ENERGY BEFORE THE MOVE

                                ncontacts = CONTACTS();

                                if (ncontacts > MAXCONTACT)
                                {
                                        contact_flag=1;
                                        break;
                                }

                                for (N=0; N<ncontacts; N++)
                                {
                                        it[N]=0;
                                }

                                for (N=0; N<MAXATOMNUM; N++)
                                {
                                        strcpy(Atoms[N][0].AtmHBlabel,"NoHB");
                                        strcpy(Atoms[N][1].AtmHBlabel,"NoHB");
                                }

                                counter_before_mcmove = TYPECONTACTS(ct0, ct1, a
t0, at1, ri0, ri1, ro0, ro1, rd, it, q0, q1, asp0, asp1);

                                energy_sum_before_mcmove = 0.0;
                                hydropathic_flag = 0;
                                for (m=0; m<counter_before_mcmove; m++)
                                {
                                        sum_vdW[m] = HARDCORE_R*(ro0[m] + ro1[m]
);

                                        if (it[m] == 1 || it[m] == 3 || it[m] ==
 4) //TYPE OF INTERACTION: 1=electro;3=electro+hidro;4=electro+HB.
                                        {
                                                if (rd[m] > sum_vdW[m] & rd[m] <
 1.4*sum_vdW[m])   //1st well of the electrostatic potential
                                                {
                                                        if (q0[m] < 0.0 & q1[m]
< 0.0)
                                                        {
                                                                ep[m] = 0.4;
                                                        }
                                                        else if (q0[m] > 0.0 & q
1[m] > 0.0)
                                                        {
                                                                ep[m] = 0.4;
```

```
                                                }
                                                else if (q0[m] < 0.0 & q
1[m] > 0.0)
                                                {
                                                        ep[m] = −0.4;
                                                }
                                                else if (q0[m] > 0.0 & q
1[m] < 0.0)
                                                {
                                                        ep[m] = −0.4;
                                                }
                                                else
                                                {
                                                        ep[m] = 0.0;
                                                }
                                        }
                                        else if (rd[m] > 1.4*sum_vdW[m]
& rd[m] < 2.33*sum_vdW[m])  //2nd well of the electrostatic potential
                                        {
                                                if (q0[m] < 0.0 & q1[m]
< 0.0)
                                                {
                                                        ep[m] = 0.3*0.4;
                                                }
                                                else if (q0[m] > 0.0 & q
1[m] > 0.0)
                                                {
                                                        ep[m] = 0.3*0.4;
                                                }
                                                else if (q0[m] < 0.0 & q
1[m] > 0.0)
                                                {
                                                        ep[m] = −0.3*0.4
;
                                                }
                                                else if (q0[m] > 0.0 & q
1[m] < 0.0)
                                                {
                                                        ep[m] = −0.3*0.4
;
                                                }
                                                else
                                                {
                                                        ep[m] = 0.0;
                                                }
                                        }
                                        else
                                        {
                                                ep[m] = 0.0;
                                        }
                                        hp[m] = 0.0;
                                        hbp[m] = 0.0;
                                        if ((it[m] == 3) && (rd[m] < 1.6
*sum_vdW[m]))  //TYPE OF INTERACTION: 3=electro+hidro;
                                        {
                                                hidropathy_0[m] = −((ato
mic_solvation_parameters[asp0[m]]*atomic_SASA[ct0[m]][at0[m]])/n)/3000;
                                                hidropathy_1[m] = −((ato
mic_solvation_parameters[asp1[m]]*atomic_SASA[ct1[m]][at1[m]])/n)/3000;

                                                if (hidropathy_0[m] <= −
0.0539995)
                                                {
                                                        hidropathy_0[m]
= −0.1;
                                                }
                                                else if ((hidropathy_0[m
] >= −0.0332997) && (hidropathy_0[m] <= 0.0484162))
                                                {
```



```
                                                        hidropathy_0[m]
= 0.0;
                                                }
                                                else if ((hidropathy_0[m
] >= 0.0617161) && (hidropathy_0[m] <= 0.138332))
                                                {
                                                        hidropathy_0[m]
= 0.1;
                                                }
                                                else if (hidropathy_0[m]
 >= 0.332063)
                                                {
                                                        hidropathy_0[m]
= 0.4;
                                                }

                                                if (hidropathy_1[m] <= −
0.0539995)
                                                {
                                                        hidropathy_1[m]
= −0.1;
                                                }
                                                else if ((hidropathy_1[m
] >= −0.0332997) && (hidropathy_1[m] <= 0.0484162))
                                                {
                                                        hidropathy_1[m]
= 0.0;
                                                }
                                                else if ((hidropathy_1[m
] >= 0.0617161) && (hidropathy_1[m] <= 0.138332))
                                                {
                                                        hidropathy_1[m]
= 0.1;
                                                }
                                                else if (hidropathy_1[m]
 >= 0.332063)
                                                {
                                                        hidropathy_1[m]
= 0.4;
                                                }

                                                hp[m] = hidropathy_0[m]+
hidropathy_1[m];
                                        }

                                        if (it[m] == 4)   //TYPE OF INTE
RACTION: 4=electro+HB;
                                        {
                                                hbp[m] = −1.0;
                                        }
                                        else
                                        {
                                                hbp[m] = 0.0;
                                        }
                                }
                                else if ((it[m] == 2) && (rd[m] < 1.6*su
m_vdW[m])) //TYPE OF INTERACTION: 2=hidro.
                                {
                                        hidropathy_0[m] = −((atomic_solv
ation_parameters[asp0[m]]*atomic_SASA[ct0[m]][at0[m]])/n)/3000;
                                        hidropathy_1[m] = −((atomic_solv
ation_parameters[asp1[m]]*atomic_SASA[ct1[m]][at1[m]])/n)/3000;

                                        if (hidropathy_0[m] <= −0.053999
5)
                                        {
                                                hidropathy_0[m] = −0.1;
                                        }
                                        else if ((hidropathy_0[m] >= −0.
```

```
0332997) && (hidropathy_0[m] <= 0.0484162))
                                        {
                                                hidropathy_0[m] = 0.0;
                                        }
                                        else if ((hidropathy_0[m] >= 0.0
617161) && (hidropathy_0[m] <= 0.138332))
                                        {
                                                hidropathy_0[m] = 0.1;
                                        }
                                        else if (hidropathy_0[m] >= 0.33
2063)
                                        {
                                                hidropathy_0[m] = 0.4;
                                        }

                                        if (hidropathy_1[m] <= −0.053999
5)
                                        {
                                                hidropathy_1[m] = −0.1;
                                        }
                                        else if ((hidropathy_1[m] >= −0.
0332997) && (hidropathy_1[m] <= 0.0484162))
                                        {
                                                hidropathy_1[m] = 0.0;
                                        }
                                        else if ((hidropathy_1[m] >= 0.0
617161) && (hidropathy_1[m] <= 0.138332))
                                        {
                                                hidropathy_1[m] = 0.1;
                                        }
                                        else if (hidropathy_1[m] >= 0.33
2063)
                                        {
                                                hidropathy_1[m] = 0.4;
                                        }

                                        hp[m] = hidropathy_0[m]+hidropat
hy_1[m];
                                        ep[m] = 0.0;
                                        hbp[m] = 0.0;
                                }
                                else
                                {
                                        ep[m] = 0.0;
                                        hp[m] = 0.0;
                                        hbp[m] = 0.0;
                                }
                                ip[m] = ep[m] + hp[m] + hbp[m];
                                energy_sum_before_mcmove = energy_sum_be
fore_mcmove + /*f[m][0]*f[m][1]**/ip[m];
                        }
                        //END of the computation of the nº of CONTACTS,
CLASHES and INTERMOLECULAR ENERGY BEFORE THE MOVE

                        //BEGINNING of the computation of the nº of CONT
ACTS, CLASHES and INTERMOLECULAR ENERGY AFTER THE MOVE

                        ncontacts = CONTACTSMOVED();

                        if (ncontacts > MAXCONTACT)
                        {
                                contact_flag=1;
                                break;
                        }

                        for (N=0; N<ncontacts; N++)
                        {
                                it[N]=0;
                        }
```



```
                        for (N=0; N<MAXATOMNUM; N++)
                        {
                                strcpy(Atoms[N][0].AtmHBlabel,"NoHB");
                                strcpy(Moved[N].AtmHBlabel,"NoHB");
                        }

                        counter_after_mcmove = TYPECONTACTSMOVED(ct0, ct
1, at0, at1, ri0, ri1, ro0, ro1, rd, it, q0, q1, asp0, asp1);

                        energy_sum_after_mcmove = 0.0;
                        hydropathic_flag = 0;
                        for (m=0; m<counter_after_mcmove; m++)
                        {
                                sum_vdW[m] = HARDCORE_R*(ro0[m] + ro1[m]
);

                                if (it[m] == 1 || it[m] == 3 || it[m] ==
 4)
                                {
                                        if (rd[m] > sum_vdW[m] & rd[m] <
 1.4*sum_vdW[m])
                                        {
                                                if (q0[m] < 0.0 & q1[m]
< 0.0)
                                                {
                                                        ep[m] = 0.4;
                                                }
                                                else if (q0[m] > 0.0 & q
1[m] > 0.0)
                                                {
                                                        ep[m] = 0.4;
                                                }
                                                else if (q0[m] < 0.0 & q
1[m] > 0.0)
                                                {
                                                        ep[m] = −0.4;
                                                }
                                                else if (q0[m] > 0.0 & q
1[m] < 0.0)
                                                {
                                                        ep[m] = −0.4;
                                                }
                                                else
                                                {
                                                        ep[m] = 0.0;
                                                }
                                        }
                                        else if (rd[m] > 1.4*sum_vdW[m]
& rd[m] < 2.33*sum_vdW[m])
                                        {
                                                if (q0[m] < 0.0 & q1[m]
< 0.0)
                                                {
                                                        ep[m] = 0.3*0.4;
                                                }
                                                else if (q0[m] > 0.0 & q
1[m] > 0.0)
                                                {
                                                        ep[m] = 0.3*0.4;
                                                }
                                                else if (q0[m] < 0.0 & q
1[m] > 0.0)
                                                {
                                                        ep[m] = −0.3*0.4
;
                                                }
                                                else if (q0[m] > 0.0 & q
1[m] < 0.0)
```

```
                                                {
                                                        ep[m] = −0.3*0.4
;
                                                }
                                                else
                                                {
                                                        ep[m] = 0.0;
                                                }
                                        }
                                        else
                                        {
                                                ep[m] = 0.0;
                                        }
                                        hp[m] = 0.0;
                                        hbp[m] = 0.0;
                                        if ((it[m] == 3) && (rd[m] < 1.6
*sum_vdW[m]))
                                        {
                                                hidropathy_0[m] = −((ato
mic_solvation_parameters[asp0[m]]*atomic_SASA[ct0[m]][at0[m]])/n)/3000;
                                                hidropathy_1[m] = −((ato
mic_solvation_parameters[asp1[m]]*atomic_SASA[ct1[m]][at1[m]])/n)/3000;

                                                if (hidropathy_0[m] <= −
0.0539995)
                                                {
                                                        hidropathy_0[m]
= −0.1;
                                                }
                                                else if ((hidropathy_0[m
] >= −0.0332997) && (hidropathy_0[m] <= 0.0484162))
                                                {
                                                        hidropathy_0[m]
= 0.0;
                                                }
                                                else if ((hidropathy_0[m
] >= 0.0617161) && (hidropathy_0[m] <= 0.138332))
                                                {
                                                        hidropathy_0[m]
= 0.1;
                                                }
                                                else if (hidropathy_0[m]
 >= 0.332063)
                                                {
                                                        hidropathy_0[m]
= 0.4;
                                                }

                                                if (hidropathy_1[m] <= −
0.0539995)
                                                {
                                                        hidropathy_1[m]
= −0.1;
                                                }
                                                else if ((hidropathy_1[m
] >= −0.0332997) && (hidropathy_1[m] <= 0.0484162))
                                                {
                                                        hidropathy_1[m]
= 0.0;
                                                }
                                                else if ((hidropathy_1[m
] >= 0.0617161) && (hidropathy_1[m] <= 0.138332))
                                                {
                                                        hidropathy_1[m]
= 0.1;
                                                }
                                                else if (hidropathy_1[m]
 >= 0.332063)
                                                {
```



```
                                                        hidropathy_1[m]
= 0.4;
                                                }

                                                hp[m] = hidropathy_0[m]+
hidropathy_1[m];
                                        }

                                        if (it[m] == 4)
                                        {
                                                hbp[m] = −1.0;
                                        }
                                        else
                                        {
                                                hbp[m] = 0.0;
                                        }
                                }
                                else if ((it[m] == 2) && (rd[m] < 1.6*su
m_vdW[m]))
                                {
                                        hidropathy_0[m] = −((atomic_solv
ation_parameters[asp0[m]]*atomic_SASA[ct0[m]][at0[m]])/n)/3000;
                                        hidropathy_1[m] = −((atomic_solv
ation_parameters[asp1[m]]*atomic_SASA[ct1[m]][at1[m]])/n)/3000;

                                        if (hidropathy_0[m] <= −0.053999
5)
                                        {
                                                hidropathy_0[m] = −0.1;
                                        }
                                        else if ((hidropathy_0[m] >= −0.
0332997) && (hidropathy_0[m] <= 0.0484162))
                                        {
                                                hidropathy_0[m] = 0.0;
                                        }
                                        else if ((hidropathy_0[m] >= 0.0
617161) && (hidropathy_0[m] <= 0.138332))
                                        {
                                                hidropathy_0[m] = 0.1;
                                        }
                                        else if (hidropathy_0[m] >= 0.33
2063)
                                        {
                                                hidropathy_0[m] = 0.4;
                                        }

                                        if (hidropathy_1[m] <= −0.053999
5)
                                        {
                                                hidropathy_1[m] = −0.1;
                                        }
                                        else if ((hidropathy_1[m] >= −0.
0332997) && (hidropathy_1[m] <= 0.0484162))
                                        {
                                                hidropathy_1[m] = 0.0;
                                        }
                                        else if ((hidropathy_1[m] >= 0.0
617161) && (hidropathy_1[m] <= 0.138332))
                                        {
                                                hidropathy_1[m] = 0.1;
                                        }
                                        else if (hidropathy_1[m] >= 0.33
2063)
                                        {
                                                hidropathy_1[m] = 0.4;
                                        }

                                        hp[m] = hidropathy_0[m]+hidropat
hy_1[m];
```

```
                                        ep[m] = 0.0;
                                        hbp[m] = 0.0;
                                }
                                else
                                {
                                        ep[m] = 0.0;
                                        hp[m] = 0.0;
                                        hbp[m] = 0.0;
                                }
                                ip[m] = ep[m] + hp[m] + hbp[m];
                                energy_sum_after_mcmove = energy_sum_aft
er_mcmove + ip[m];
                        }
                        //END of the computation of the nº of CONTACTS,
CLASHES and INTERMOLECULAR ENERGY AFTER THE MOVE

                        Econtold = energy_sum_before_mcmove;
                        Econtnew = energy_sum_after_mcmove;
                        contactnew = (double)CONTACTSMOVED();
                        contactold = (double)CONTACTS();
                        clashnew = (double)CLASHESMOVED();
                        clashold = (double)CLASHES();


                        //METROPOLIS CRITERION

                        if (SCALE_CLASHES*(clashnew − clashold) < 0.)
                        {
                                if ((1−SCALE_CLASHES)*(Econtnew − Econto
ld) < 0.)
                                {
                                        accepted++;
                                        for (m = 0; m<nAtom[1]; m++)
                                        {
                                                Atoms[m][1] = Moved[m];
                                        }
                                        e = k−1;
                                        mc = index−1;
                                }
                                else
                                {
                                        if (exp(−((1−SCALE_CLASHES)*(Eco
ntnew − Econtold)) / Tm) >= RandomUniform())
                                        {
                                                accepted++;
                                                for (m = 0; m<nAtom[1];
m++)
                                                {
                                                        Atoms[m][1] = Mo
ved[m];
                                                }
                                                e = k−1;
                                                mc = index−1;
                                        }
                                        else
                                        {
                                                rejected++;
                                                e = k−1;
                                                mc = index−1;
                                        }
                                }
                        }
                        else
                        {
                                if ((exp(−(SCALE_CLASHES*(clashnew − cla
shold)) / Tm)) >= RandomUniform())
                                {
                                        if ((1−SCALE_CLASHES)*(Econtnew
− Econtold) < 0)
```



```
                                        {
                                                accepted++;
                                                for (m = 0; m<nAtom[1];
m++)
                                                {
                                                        Atoms[m][1] = Mo
ved[m];
                                                }
                                                e = k−1;
                                                mc = index−1;
                                        }
                                        else
                                        {
                                                if (exp(−((1−SCALE_CLASH
ES)*(Econtnew − Econtold)) / Tm) >= RandomUniform())
                                                {
                                                        accepted++;
                                                        for (m = 0; m<nA
tom[1]; m++)
                                                        {
                                                                Atoms[m]
[1] = Moved[m];
                                                        }
                                                        e = k−1;
                                                        mc = index−1;
                                                }
                                                else
                                                {
                                                        rejected++;
                                                        e = k−1;
                                                        mc = index−1;
                                                }
                                        }
                                }
                                else
                                {
                                        rejected++;
                                        e = k−1;
                                        mc = index−1;
                                }
                        }
                        Pacc=(double)accepted/((double)index);
                        Pdecl=(double)rejected/((double)index);

                        //METROPOLIS CONVERGENCE OPTIMIZATION

                        if (Pacc>0.55)
                        {
                                mcinternal1=mcinternal1+0.00001*mcintern
al1;
                                mcinternal2=mcinternal2+0.00001*mcintern
al2;
                        }
                        else
                        {
                                if (Pacc<0.45)
                                {
                                        mcinternal1=mcinternal1−0.00001*
mcinternal1;
                                        mcinternal2=mcinternal2−0.00001*
mcinternal2;
                                }
                        }

                        if (mcinternal1>=1.0 & mcinternal2>=1.0)
                        {
                                mcinternal1=0.025;
                                mcinternal2=0.025;
                        }
```

```
                                if (mcinternal1<=0.01 & mcinternal2<=0.01)
                                {
                                        mcinternal1=0.025;
                                        mcinternal2=0.025;
                                }


                                //PROTOCOL PART

                                //COMPUTATION OF CONTACTS, CLASHES AND ENERGIES
AFTER METROPOLIS CRITERION

                                ncontacts = CONTACTS();

                                if (ncontacts > MAXCONTACT)
                                {
                                        contact_flag=1;
                                        break;
                                }

                                for (N=0; N<ncontacts; N++)
                                {
                                        it[N]=0;
                                }

                                for (N=0; N<MAXATOMNUM; N++)
                                {
                                        strcpy(Atoms[N][0].AtmHBlabel,"NoHB");
                                        strcpy(Atoms[N][1].AtmHBlabel,"NoHB");
                                }


                                final_counter = TYPECONTACTS(ct0, ct1, at0, at1,
 ri0, ri1, ro0, ro1, rd, it, q0, q1, asp0, asp1);

                                final_energy_sum = 0.0;
                                final_hp_energy_sum = 0.0;
                                final_ep_energy_sum = 0.0;
                                final_hbp_energy_sum = 0.0;

                                hydropathic_flag = 0;
                                for (m=0; m<final_counter; m++)
                                {
                                        sum_vdW[m] = HARDCORE_R*(ro0[m] + ro1[m]
);

                                        if (it[m] == 1 || it[m] == 3 || it[m] ==
 4)
                                        {
                                                if (rd[m] > sum_vdW[m] & rd[m] <
 1.4*sum_vdW[m])
                                                {
                                                        if (q0[m] < 0.0 & q1[m]
< 0.0)
                                                        {
                                                                ep[m] = 0.4;
                                                        }
                                                        else if (q0[m] > 0.0 & q
1[m] > 0.0)
                                                        {
                                                                ep[m] = 0.4;
                                                        }
                                                        else if (q0[m] < 0.0 & q
1[m] > 0.0)
                                                        {
                                                                ep[m] = −0.4;
                                                        }
                                                        else if (q0[m] > 0.0 & q
```



```
1[m] < 0.0)
                                                        {
                                                                ep[m] = −0.4;
                                                        }
                                                        else
                                                        {
                                                                ep[m] = 0.0;
                                                        }
                                                }
                                                else if (rd[m] > 1.4*sum_vdW[m]
& rd[m] < 2.33*sum_vdW[m])
                                                {
                                                        if (q0[m] < 0.0 & q1[m]
< 0.0)
                                                        {
                                                                ep[m] = 0.3*0.4;
                                                        }
                                                        else if (q0[m] > 0.0 & q
1[m] > 0.0)
                                                        {
                                                                ep[m] = 0.3*0.4;
                                                        }
                                                        else if (q0[m] < 0.0 & q
1[m] > 0.0)
                                                        {
                                                                ep[m] = −0.3*0.4
;
                                                        }
                                                        else if (q0[m] > 0.0 & q
1[m] < 0.0)
                                                        {
                                                                ep[m] = −0.3*0.4
;
                                                        }
                                                        else
                                                        {
                                                                ep[m] = 0.0;
                                                        }
                                                }
                                                else
                                                {
                                                        ep[m] = 0.0;
                                                }
                                                hp[m] = 0.0;
                                                hbp[m] = 0.0;
                                                if ((it[m] == 3) && (rd[m] < 1.6
*sum_vdW[m]))
                                                {
                                                        hidropathy_0[m] = −((ato
mic_solvation_parameters[asp0[m]]*atomic_SASA[ct0[m]][at0[m]])/n)/3000;
                                                        hidropathy_1[m] = −((ato
mic_solvation_parameters[asp1[m]]*atomic_SASA[ct1[m]][at1[m]])/n)/3000;

                                                        if (hidropathy_0[m] <= −
0.0539995)
                                                        {
                                                                hidropathy_0[m]
= −0.1;
                                                        }
                                                        else if ((hidropathy_0[m
] >= −0.0332997) && (hidropathy_0[m] <= 0.0484162))
                                                        {
                                                                hidropathy_0[m]
= 0.0;
                                                        }
                                                        else if ((hidropathy_0[m
] >= 0.0617161) && (hidropathy_0[m] <= 0.138332))
                                                        {
                                                                hidropathy_0[m]
```

```
= 0.1;
                                                }
                                                else if (hidropathy_0[m]
 >= 0.332063)
                                                {
                                                        hidropathy_0[m]
= 0.4;
                                                }

                                                if (hidropathy_1[m] <= −
0.0539995)
                                                {
                                                        hidropathy_1[m]
= −0.1;
                                                }
                                                else if ((hidropathy_1[m
] >= −0.0332997) && (hidropathy_1[m] <= 0.0484162))
                                                {
                                                        hidropathy_1[m]
= 0.0;
                                                }
                                                else if ((hidropathy_1[m
] >= 0.0617161) && (hidropathy_1[m] <= 0.138332))
                                                {
                                                        hidropathy_1[m]
= 0.1;
                                                }
                                                else if (hidropathy_1[m]
 >= 0.332063)
                                                {
                                                        hidropathy_1[m]
= 0.4;
                                                }

                                                hp[m] = hidropathy_0[m]+
hidropathy_1[m];
                                        }

                                        if (it[m] == 4)
                                        {
                                                hbp[m] = −1.0;
                                        }
                                        else
                                        {
                                                hbp[m] = 0.0;
                                        }
                                }
                                else if ((it[m] == 2) && (rd[m] < 1.6*su
m_vdW[m]))
                                {
                                        hidropathy_0[m] = −((atomic_solv
ation_parameters[asp0[m]]*atomic_SASA[ct0[m]][at0[m]])/n)/3000;
                                        hidropathy_1[m] = −((atomic_solv
ation_parameters[asp1[m]]*atomic_SASA[ct1[m]][at1[m]])/n)/3000;

                                        if (hidropathy_0[m] <= −0.053999
5)
                                        {
                                                hidropathy_0[m] = −0.1;
                                        }
                                        else if ((hidropathy_0[m] >= −0.
0332997) && (hidropathy_0[m] <= 0.0484162))
                                        {
                                                hidropathy_0[m] = 0.0;
                                        }
                                        else if ((hidropathy_0[m] >= 0.0
617161) && (hidropathy_0[m] <= 0.138332))
                                        {
                                                hidropathy_0[m] = 0.1;
```



```
                                                }
                                                else if (hidropathy_0[m] >= 0.33
2063)
                                                {
                                                        hidropathy_0[m] = 0.4;
                                                }

                                                if (hidropathy_1[m] <= −0.053999
5)
                                                {
                                                        hidropathy_1[m] = −0.1;
                                                }
                                                else if ((hidropathy_1[m] >= −0.
0332997) && (hidropathy_1[m] <= 0.0484162))
                                                {
                                                        hidropathy_1[m] = 0.0;
                                                }
                                                else if ((hidropathy_1[m] >= 0.0
617161) && (hidropathy_1[m] <= 0.138332))
                                                {
                                                        hidropathy_1[m] = 0.1;
                                                }
                                                else if (hidropathy_1[m] >= 0.33
2063)
                                                {
                                                        hidropathy_1[m] = 0.4;
                                                }

                                                hp[m] = hidropathy_0[m]+hidropat
hy_1[m];
                                                ep[m] = 0.0;
                                                hbp[m] = 0.0;
                                        }
                                        else
                                        {
                                                ep[m] = 0.0;
                                                hp[m] = 0.0;
                                                hbp[m] = 0.0;
                                        }
                                        ip[m] = ep[m] + hp[m] + hbp[m];
                                        final_energy_sum = final_energy_sum + /*
f[m][0]*f[m][1]**/ip[m];
                                        final_hp_energy_sum = final_hp_energy_su
m + hp[m];
                                        final_ep_energy_sum = final_ep_energy_su
m + ep[m];
                                        final_hbp_energy_sum = final_hbp_energy_
sum + hbp[m];
                                }

                                //COMPUTATION OF CONTACTS, CLASHES AND ENERGIES
AT PROTSTEP MC STEP
                                if ((index % PROTSTEP == 0)&&(index>0))
                                {
                                        ncontacts = CONTACTS();

                                        if (ncontacts > MAXCONTACT)
                                        {
                                                contact_flag=1;
                                                break;
                                        }

                                        for (N=0; N<ncontacts; N++)
                                        {
                                                it[N]=0;
                                        }

                                        for (N=0; N<MAXATOMNUM; N++)
                                        {
```

```
                                        strcpy(Atoms[N][0].AtmHBlabel,"
NoHB");
                                        strcpy(Atoms[N][1].AtmHBlabel,"
NoHB");
                                }

                                final_counter = TYPECONTACTS(ct0, ct1, a
t0, at1, ri0, ri1, ro0, ro1, rd, it, q0, q1, asp0, asp1);

                                final_energy_sum = 0.0;
                                final_hp_energy_sum = 0.0;
                                final_ep_energy_sum = 0.0;
                                final_hbp_energy_sum = 0.0;

                                hydropathic_flag = 0;
                                for (m=0; m<final_counter; m++)
                                {
                                        sum_vdW[m] = HARDCORE_R*(ro0[m]
+ ro1[m]);

                                        if (it[m] == 1 || it[m] == 3 ||
it[m] == 4)
                                        {
                                                if (rd[m] > sum_vdW[m] &
 rd[m] < 1.4*sum_vdW[m])
                                                {
                                                        if (q0[m] < 0.0
& q1[m] < 0.0)
                                                        {
                                                                ep[m] =
0.4;
                                                        }
                                                        else if (q0[m] >
 0.0 & q1[m] > 0.0)
                                                        {
                                                                ep[m] =
0.4;
                                                        }
                                                        else if (q0[m] <
 0.0 & q1[m] > 0.0)
                                                        {
                                                                ep[m] =
−0.4;
                                                        }
                                                        else if (q0[m] >
 0.0 & q1[m] < 0.0)
                                                        {
                                                                ep[m] =
−0.4;
                                                        }
                                                        else
                                                        {
                                                                ep[m] =
0.0;
                                                        }
                                                }
                                                else if (rd[m] > 1.4*sum
_vdW[m] & rd[m] < 2.33*sum_vdW[m])
                                                {
                                                        if (q0[m] < 0.0
& q1[m] < 0.0)
                                                        {
                                                                ep[m] =
0.3*0.4;
                                                        }
                                                        else if (q0[m] >
 0.0 & q1[m] > 0.0)
                                                        {
                                                                ep[m] =
```



```
0.3*0.4;
                                                        }
                                                        else if (q0[m] <
 0.0 & q1[m] > 0.0)
                                                        {
                                                                ep[m] =
−0.3*0.4;
                                                        }
                                                        else if (q0[m] >
 0.0 & q1[m] < 0.0)
                                                        {
                                                                ep[m] =
−0.3*0.4;
                                                        }
                                                        else
                                                        {
                                                                ep[m] =
0.0;
                                                        }
                                                }
                                                else
                                                {
                                                        ep[m] = 0.0;
                                                }
                                                hp[m] = 0.0;
                                                hbp[m] = 0.0;
                                                if ((it[m] == 3) && (rd[
m] < 1.6*sum_vdW[m]))
                                                {
                                                        hidropathy_0[m]
= −((atomic_solvation_parameters[asp0[m]]*atomic_SASA[ct0[m]][at0[m]])/n)/3000;
                                                        hidropathy_1[m]
= −((atomic_solvation_parameters[asp1[m]]*atomic_SASA[ct1[m]][at1[m]])/n)/3000;

                                                        if (hidropathy_0
[m] <= −0.0539995)
                                                        {
                                                                hidropat
hy_0[m] = −0.1;
                                                        }
                                                        else if ((hidrop
athy_0[m] >= −0.0332997) && (hidropathy_0[m] <= 0.0484162))
                                                        {
                                                                hidropat
hy_0[m] = 0.0;
                                                        }
                                                        else if ((hidrop
athy_0[m] >= 0.0617161) && (hidropathy_0[m] <= 0.138332))
                                                        {
                                                                hidropat
hy_0[m] = 0.1;
                                                        }
                                                        else if (hidropa
thy_0[m] >= 0.332063)
                                                        {
                                                                hidropat
hy_0[m] = 0.4;
                                                        }

                                                        if (hidropathy_1
[m] <= −0.0539995)
                                                        {
                                                                hidropat
hy_1[m] = −0.1;
                                                        }
                                                        else if ((hidrop
athy_1[m] >= −0.0332997) && (hidropathy_1[m] <= 0.0484162))
                                                        {
                                                                hidropat
```

```
hy_1[m] = 0.0;
                                                        }
                                                        else if ((hidrop
athy_1[m] >= 0.0617161) && (hidropathy_1[m] <= 0.138332))
                                                        {
                                                                hidropat
hy_1[m] = 0.1;
                                                        }
                                                        else if (hidropa
thy_1[m] >= 0.332063)
                                                        {
                                                                hidropat
hy_1[m] = 0.4;
                                                        }

                                                        hp[m] = hidropat
hy_0[m]+hidropathy_1[m];
                                                }

                                                if (it[m] == 4)
                                                {
                                                        hbp[m] = −1.0;
                                                }
                                                else
                                                {
                                                        hbp[m] = 0.0;
                                                }
                                        }
                                        else if ((it[m] == 2) && (rd[m]
< 1.6*sum_vdW[m]))
                                        {
                                                hidropathy_0[m] = −((ato
mic_solvation_parameters[asp0[m]]*atomic_SASA[ct0[m]][at0[m]])/n)/3000;
                                                hidropathy_1[m] = −((ato
mic_solvation_parameters[asp1[m]]*atomic_SASA[ct1[m]][at1[m]])/n)/3000;

                                                if (hidropathy_0[m] <= −
0.0539995)
                                                {
                                                        hidropathy_0[m]
= −0.1;
                                                }
                                                else if ((hidropathy_0[m
] >= −0.0332997) && (hidropathy_0[m] <= 0.0484162))
                                                {
                                                        hidropathy_0[m]
= 0.0;
                                                }
                                                else if ((hidropathy_0[m
] >= 0.0617161) && (hidropathy_0[m] <= 0.138332))
                                                {
                                                        hidropathy_0[m]
= 0.1;
                                                }
                                                else if (hidropathy_0[m]
 >= 0.332063)
                                                {
                                                        hidropathy_0[m]
= 0.4;
                                                }

                                                if (hidropathy_1[m] <= −
0.0539995)
                                                {
                                                        hidropathy_1[m]
= −0.1;
                                                }
                                                else if ((hidropathy_1[m
] >= −0.0332997) && (hidropathy_1[m] <= 0.0484162))
```



```
                                                {
                                                        hidropathy_1[m]
= 0.0;
                                                }
                                                else if ((hidropathy_1[m
] >= 0.0617161) && (hidropathy_1[m] <= 0.138332))
                                                {
                                                        hidropathy_1[m]
= 0.1;
                                                }
                                                else if (hidropathy_1[m]
 >= 0.332063)
                                                {
                                                        hidropathy_1[m]
= 0.4;
                                                }

                                                hp[m] = hidropathy_0[m]+
hidropathy_1[m];
                                                ep[m] = 0.0;
                                                hbp[m] = 0.0;
                                        }
                                        else
                                        {
                                                ep[m] = 0.0;
                                                hp[m] = 0.0;
                                                hbp[m] = 0.0;
                                        }

                                        ip[m] = ep[m] + hp[m] + hbp[m];
                                        final_energy_sum = final_energy_
sum + /*f[m][0]*f[m][1]**/ip[m];
                                        final_hp_energy_sum = final_hp_e
nergy_sum + hp[m];
                                        final_ep_energy_sum = final_ep_e
nergy_sum + ep[m];
                                        final_hbp_energy_sum = final_hbp
_energy_sum + hbp[m];
                                }
                                /*fprintf(output7,"%d    %d    %d     %d
   %4.3lf  %4.3lf  %4.3lf  %4.3lf  %d    %d   %4.3lf   %7.6lf    %7.6lf  %d\n",k
,i,j,index,(1−TOL)*restdist,currentdist,(1+TOL0)*restdist,final_energy_sum,CONTA
CTS(),CLASHES(),Pacc, mcinternal1,mcinternal2,hitwall);*/
                                //fprintf(output1,"%d   %d      %d
%4.3lf  %4.3lf  %4.3lf  %d  %d %4.3lf\n",i,j,index,currentdist,restdist,final_en
ergy_sum,CONTACTS(),CLASHES(),Pacc);
                                //fflush(output1);
                                printf("%d  %d  %d   %d  %4.3lf %4.3lf %4.3lf
%4.3lf %4.3lf       %4.3lf     %4.3lf     %d       %d   %d  %4.3lf  %7.6lf   %7.6lf %d\n",k,i,j,ind
ex,(1−TOL)*restdist,currentdist,(1+TOL0)*restdist,final_energy_sum,final_hp_ener
gy_sum,final_ep_energy_sum,final_hbp_energy_sum,CONTACTS(),final_counter,CLASHES
(),Pacc, mcinternal1,mcinternal2,hitwall);
                        }
                        //COMPUTATION OF CONTACTS, CLASHES AND ENERGIES
AT LAST MC STEP
                        if (index == MCSTEPS)
                        {
                                ncontacts = CONTACTS();

                                if (ncontacts > MAXCONTACT)
                                {
                                        contact_flag=1;
                                        break;
                                }

                                for (N=0; N<ncontacts; N++)
                                {
                                        it[N]=0;
                                }
```

```
                                        for (N=0; N<MAXATOMNUM; N++)
                                        {
                                                strcpy(Atoms[N][0].AtmHBlabel,"
NoHB");
                                                strcpy(Atoms[N][1].AtmHBlabel,"
NoHB");
                                        }

                                        final_counter = TYPECONTACTS(ct0, ct1, a
t0, at1, ri0, ri1, ro0, ro1, rd, it, q0, q1, asp0, asp1);

                                        final_energy_sum = 0.0;
                                        final_hp_energy_sum = 0.0;
                                        final_ep_energy_sum = 0.0;
                                        final_hbp_energy_sum = 0.0;

                                        hydropathic_flag = 0;
                                        for (m=0; m<final_counter; m++)
                                        {
                                                sum_vdW[m] = HARDCORE_R*(ro0[m]
+ ro1[m]);

                                                if (it[m] == 1 || it[m] == 3 ||
it[m] == 4)
                                                {
                                                        if (rd[m] > sum_vdW[m] &
 rd[m] < 1.4*sum_vdW[m])
                                                        {
                                                                if (q0[m] < 0.0
& q1[m] < 0.0)
                                                                {
                                                                        ep[m] =
0.4;
                                                                }
                                                                else if (q0[m] >
 0.0 & q1[m] > 0.0)
                                                                {
                                                                        ep[m] =
0.4;
                                                                }
                                                                else if (q0[m] <
 0.0 & q1[m] > 0.0)
                                                                {
                                                                        ep[m] =
−0.4;
                                                                }
                                                                else if (q0[m] >
 0.0 & q1[m] < 0.0)
                                                                {
                                                                        ep[m] =
−0.4;
                                                                }
                                                                else
                                                                {
                                                                        ep[m] =
0.0;
                                                                }
                                                        }
                                                        else if (rd[m] > 1.4*sum
_vdW[m] & rd[m] < 2.33*sum_vdW[m])
                                                        {
                                                                if (q0[m] < 0.0
& q1[m] < 0.0)
                                                                {
                                                                        ep[m] =
0.3*0.4;
                                                                }
                                                                else if (q0[m] >
```



```
 0.0 & q1[m] > 0.0)
                                                                {
                                                                        ep[m] =
0.3*0.4;
                                                                }
                                                                else if (q0[m] <
 0.0 & q1[m] > 0.0)
                                                                {
                                                                        ep[m] =
−0.3*0.4;
                                                                }
                                                                else if (q0[m] >
 0.0 & q1[m] < 0.0)
                                                                {
                                                                        ep[m] =
−0.3*0.4;
                                                                }
                                                                else
                                                                {
                                                                        ep[m] =
0.0;
                                                                }
                                                        }
                                                        else
                                                        {
                                                                ep[m] = 0.0;
                                                        }
                                                        hp[m] = 0.0;
                                                        hbp[m] = 0.0;
                                                        if ((it[m] == 3) && (rd[
m] < 1.6*sum_vdW[m]))
                                                        {
                                                                hidropathy_0[m]
= −((atomic_solvation_parameters[asp0[m]]*atomic_SASA[ct0[m]][at0[m]])/n)/3000;
                                                                hidropathy_1[m]
= −((atomic_solvation_parameters[asp1[m]]*atomic_SASA[ct1[m]][at1[m]])/n)/3000;

                                                                if (hidropathy_0
[m] <= −0.0539995)
                                                                {
                                                                        hidropat
hy_0[m] = −0.1;
                                                                }
                                                                else if ((hidrop
athy_0[m] >= −0.0332997) && (hidropathy_0[m] <= 0.0484162))
                                                                {
                                                                        hidropat
hy_0[m] = 0.0;
                                                                }
                                                                else if ((hidrop
athy_0[m] >= 0.0617161) && (hidropathy_0[m] <= 0.138332))
                                                                {
                                                                        hidropat
hy_0[m] = 0.1;
                                                                }
                                                                else if (hidropa
thy_0[m] >= 0.332063)
                                                                {
                                                                        hidropat
hy_0[m] = 0.4;
                                                                }

                                                                if (hidropathy_1
[m] <= −0.0539995)
                                                                {
                                                                        hidropat
hy_1[m] = −0.1;
                                                                }
```

```
                                                        else if ((hidrop
athy_1[m] >= −0.0332997) && (hidropathy_1[m] <= 0.0484162))
                                                        {
                                                                hidropat
hy_1[m] = 0.0;
                                                        }
                                                        else if ((hidrop
athy_1[m] >= 0.0617161) && (hidropathy_1[m] <= 0.138332))
                                                        {
                                                                hidropat
hy_1[m] = 0.1;
                                                        }
                                                        else if (hidropa
thy_1[m] >= 0.332063)
                                                        {
                                                                hidropat
hy_1[m] = 0.4;
                                                        }

                                                        hp[m] = hidropat
hy_0[m]+hidropathy_1[m];
                                                }

                                                if (it[m] == 4)
                                                {
                                                        hbp[m] = −1.0;
                                                }
                                                else
                                                {
                                                        hbp[m] = 0.0;
                                                }
                                        }
                                        else if ((it[m] == 2) && (rd[m]
< 1.6*sum_vdW[m]))
                                        {
                                                hidropathy_0[m] = −((ato
mic_solvation_parameters[asp0[m]]*atomic_SASA[ct0[m]][at0[m]])/n)/3000;
                                                hidropathy_1[m] = −((ato
mic_solvation_parameters[asp1[m]]*atomic_SASA[ct1[m]][at1[m]])/n)/3000;

                                                if (hidropathy_0[m] <= −
0.0539995)
                                                {
                                                        hidropathy_0[m]
= −0.1;
                                                }
                                                else if ((hidropathy_0[m
] >= −0.0332997) && (hidropathy_0[m] <= 0.0484162))
                                                {
                                                        hidropathy_0[m]
= 0.0;
                                                }
                                                else if ((hidropathy_0[m
] >= 0.0617161) && (hidropathy_0[m] <= 0.138332))
                                                {
                                                        hidropathy_0[m]
= 0.1;
                                                }
                                                else if (hidropathy_0[m]
 >= 0.332063)
                                                {
                                                        hidropathy_0[m]
= 0.4;
                                                }

                                                if (hidropathy_1[m] <= −
0.0539995)
                                                {
                                                        hidropathy_1[m]
```



```
= −0.1;
                                                }
                                                else if ((hidropathy_1[m
] >= −0.0332997) && (hidropathy_1[m] <= 0.0484162))
                                                {
                                                        hidropathy_1[m]
= 0.0;
                                                }
                                                else if ((hidropathy_1[m
] >= 0.0617161) && (hidropathy_1[m] <= 0.138332))
                                                {
                                                        hidropathy_1[m]
= 0.1;
                                                }
                                                else if (hidropathy_1[m]
 >= 0.332063)
                                                {
                                                        hidropathy_1[m]
= 0.4;
                                                }

                                                hp[m] = hidropathy_0[m]+
hidropathy_1[m];
                                                ep[m] = 0.0;
                                                hbp[m] = 0.0;
                                        }
                                        else
                                        {
                                                ep[m] = 0.0;
                                                hp[m] = 0.0;
                                                hbp[m] = 0.0;
                                        }

                                        ip[m] = ep[m] + hp[m] + hbp[m];
                                        final_energy_sum = final_energy_
sum + ip[m];
                                        final_hp_energy_sum = final_hp_e
nergy_sum + hp[m];
                                        final_ep_energy_sum = final_ep_e
nergy_sum + ep[m];
                                        final_hbp_energy_sum = final_hbp
_energy_sum + hbp[m];
                                }
                                fprintf(output1,"%d      %d      %d
%4.3lf     %4.3lf     %4.3lf     %4.3lf     %4.3lf     %4.3lf     %d      %d   %d    %4.3lf\n",i,j,ind
ex,currentdist,restdist,final_energy_sum,final_hp_energy_sum,final_ep_energy_sum
,final_hbp_energy_sum,CONTACTS(),final_counter,CLASHES(),Pacc);
                                fflush(output1);

                                ExportSnapshot(i,j);
                        }

                }
                //END OF MONTE CARLO

                e = k−1;

                CCONTACTS[e] = CONTACTS();
                CCLASHES[e] = CLASHES();
                ENERGY[e] = final_energy_sum;
                CONTACTS_SUM = CONTACTS_SUM + CCONTACTS[e];
                CLASHES_SUM = CLASHES_SUM + CCLASHES[e];
                ENERGY_SUM = ENERGY_SUM + ENERGY[e];

                if (k % 100 == 0)
                {
                        CONTACTS_MEAN = (CONTACTS_SUM/(double)k);

                        CLASHES_MEAN = (CLASHES_SUM/(double)k);
```

```
                                ENERGY_MEAN = (ENERGY_SUM/(double)k);

                                for (e=0; e < k; e++)
                                {
                                        contacts_diff_sum = contacts_diff_sum +
(CCONTACTS[e] − CONTACTS_MEAN)*(CCONTACTS[e] − CONTACTS_MEAN);
                                }

                                VARIANCE_CONTACTS = contacts_diff_sum/(double)k;

                                VARIANCE_CONTACTS_MEAN = VARIANCE_CONTACTS/(doub
le)k;

                                for (e=0; e < k; e++)
                                {
                                        energy_diff_sum = energy_diff_sum + (ENE
RGY[e] − ENERGY_MEAN)*(ENERGY[e] − ENERGY_MEAN);
                                }

                                VARIANCE_ENERGY = energy_diff_sum/(double)k;

                                VARIANCE_ENERGY_MEAN = VARIANCE_ENERGY/(double)k
;

                                fprintf(output2,"%lf  %lf  %lf  %lf  %lf\n", CONTAC
TS_MEAN, CLASHES_MEAN, ENERGY_MEAN, VARIANCE_CONTACTS_MEAN, VARIANCE_ENERGY_MEAN
);

                        }

        }


        fclose(output1);

        fclose(output2);


        return 0;

}


void MCMOVE(double a)
{
        int i,j,k;
        double translrand;
        double rotrand1;
        double rotrand2;
        long double helper1,helper2,helper3,helper4;
        double rotvector[3];
        double rotmat[3][3];
        double oldmoved_x[MAXATOMNUM];
        double oldmoved_y[MAXATOMNUM];
        double oldmoved_z[MAXATOMNUM];

        long double tester1,tester2;

        if (a<0.5)
        {

                //translational MC−submove along x−axis

                while (1)
                {
                        translrand=RandomUniform()−0.5;
                        /*printf("%lf\n", mcinternal1); */
                        currentdist=currentdist+mcinternal1*translrand;
```



```
                        if ((currentdist<=(1+TOL0)*restdist)&&(currentdist>=(1−T
OL)*restdist))
                                break;
                }

                if (abbs(DIR)==1)
                {
                        for (j=0;j<nAtom[1];j++)
                        {
                                Moved[j].x = Moved[j].x + mcinternal1*translrand
;
                        }

                }
                else
                {
                        if (abbs(DIR)==2)
                        {
                                for (j=0;j<nAtom[1];j++)
                                {
                                        Moved[j].y = Moved[j].y + mcinternal1*tr
anslrand;
                                }

                        }
                        else
                        {
                                if (abbs(DIR)==3)
                                {
                                        for (j=0;j<nAtom[1];j++)
                                        {
                                                Moved[j].z = Moved[j].z + mcinte
rnal1*translrand;
                                        }

                                }
                        }
                }
        }
        else
        {

                /*for (i=0;i<nAtom[0];i++)
                {
                        controldist[i][0]=sqrt((Moved[i].x − CM2[0])*(Moved[i].x
 − CM2[0])+(Moved[i].y − CM2[1])*(Moved[i].y − CM2[1])+(Moved[i].z − CM2[2])*(Mo
ved[i].z − CM2[2]));
                }*/


                //rotational MC−submove

                while (1)
                {
                        rotrand1=PI*RandomUniform();                    //theta
on unit sphere in [0,pi]
                        rotrand2=2*PI*RandomUniform();                  //phi on
 unit sphere in [0,2pi]

                        helper1=sin(rotrand1);
                        helper2=cos(rotrand1);
                        helper3=sin(rotrand2);
                        helper4=cos(rotrand2);

                        rotvector[0]=helper1*helper4;
//random unit vector in spherical coord. defining rotation axis
                        rotvector[1]=helper1*helper3;
                        rotvector[2]=helper4;
```

```
                        tester1=rotvector[0]*rotvector[0]+rotvector[1]*rotvector
[1]+rotvector[2]*rotvector[2];

                        rotvector[0]=rotvector[0]/tester1;
                        rotvector[1]=rotvector[1]/tester1;
                        rotvector[2]=rotvector[2]/tester1;

                        tester2=rotvector[0]*rotvector[0]+rotvector[1]*rotvector
[1]+rotvector[2]*rotvector[2];

                        if (tester2==1.)
                        {
                                break;
                        }
                }
                /*printf("%lf\n", mcinternal2);*/

                helper1=2*PI*mcinternal2*RandomUniform();               //alpha

                helper2=sin(helper1);
                helper3=cos(helper1);
                helper4=1−helper3;

                //first matrix index: line
                //second matrix index: column

                //setting up rotation matrix
                rotmat[0][0] = rotvector[0]*rotvector[0]*helper4+helper3;
                rotmat[0][1] = rotvector[0]*rotvector[1]*helper4−rotvector[2]*he
lper2;
                rotmat[0][2] = rotvector[0]*rotvector[2]*helper4+rotvector[1]*he
lper2;
                rotmat[1][0] = rotvector[0]*rotvector[1]*helper4+rotvector[2]*he
lper2;
                rotmat[1][1] = rotvector[1]*rotvector[1]*helper4+helper3;
                rotmat[1][2] = rotvector[1]*rotvector[2]*helper4−rotvector[0]*he
lper2;
                rotmat[2][0] = rotvector[0]*rotvector[2]*helper4−rotvector[1]*he
lper2;
                rotmat[2][1] = rotvector[1]*rotvector[2]*helper4+rotvector[0]*he
lper2;
                rotmat[2][2] = rotvector[2]*rotvector[2]*helper4+helper3;

                //recalculating the coordinates of the structure stored in Moved
 in relation to its centre of mass

                for (j=0;j<3;j++)
                {
                        MEAN_R[j][1]=0;
                }

                for (j=0;j<nAtom[1];j++)
                {
                        MEAN_R[0][1]=MEAN_R[0][1]+Moved[j].x;
                        MEAN_R[1][1]=MEAN_R[1][1]+Moved[j].y;
                        MEAN_R[2][1]=MEAN_R[2][1]+Moved[j].z;
                }

                for (j=0;j<3;j++)
                {
                        MEAN_R[j][1]=MEAN_R[j][1]/nAtom[1];
                }

                for (i=0;i<nAtom[1];i++)
                {
                        RotMoved[i].x = Moved[i].x − MEAN_R[0][1];
                        RotMoved[i].y = Moved[i].y − MEAN_R[1][1];
                        RotMoved[i].z = Moved[i].z − MEAN_R[2][1];
                }
```



```
                //performing rotation on structure stored in RotMoved via matrix
 multiplication

                for (i=0;i<nAtom[1];i++)
                {
                        oldmoved_x[i] = RotMoved[i].x;
                        oldmoved_y[i] = RotMoved[i].y;
                        oldmoved_z[i] = RotMoved[i].z;
                        RotMoved[i].x = rotmat[0][0]*oldmoved_x[i] + rotmat[0][1
]*oldmoved_y[i] + rotmat[0][2]*oldmoved_z[i];
                        RotMoved[i].y = rotmat[1][0]*oldmoved_x[i] + rotmat[1][1
]*oldmoved_y[i] + rotmat[1][2]*oldmoved_z[i];
                        RotMoved[i].z = rotmat[2][0]*oldmoved_x[i] + rotmat[2][1
]*oldmoved_y[i] + rotmat[2][2]*oldmoved_z[i];
                }

                //recalculating the coordinates of the structure stored in RotMo
ved in relation to the origin of the cartesian axes

                for (i=0;i<nAtom[1];i++)
                {
                        Moved[i].x = RotMoved[i].x + MEAN_R[0][1];
                        Moved[i].y = RotMoved[i].y + MEAN_R[1][1];
                        Moved[i].z = RotMoved[i].z + MEAN_R[2][1];
                }
        }
}


int TYPECONTACTSMOVED(int ct0[MAXCONTACT], int ct1[MAXCONTACT], int at0[MAXCONTA
CT], int at1[MAXCONTACT], int ri0[MAXCONTACT], int ri1[MAXCONTACT], double ro0[M
AXCONTACT], double ro1[MAXCONTACT], double rd[MAXCONTACT], int it[MAXCONTACT], f
loat q0[MAXCONTACT], float q1[MAXCONTACT], int asp0[MAXCONTACT], int asp1[MAXCON
TACT])
{
        FILE *test;

        int i, k, l, n, o, p, r, maxcontactflag, counter, len1, len2, residue_in
dex_0, residue_index_1, s, t, ss, tt;
        int AtomIndex_1, AtomIndex_2;
        int electrostatic;
        float dDA, dc1, dc2, dc3;
        char s[256];


        for (k = 0; k<nAtom[0]; k++)
        {
                strcpy(Atoms[k][0].AtmHBlabel, "NoHB");
        }

        for (l = 0; l<nAtom[1]; l++)
        {
                strcpy(Moved[l].AtmHBlabel, "NoHB");
        }

        counter = 0;
        for (i = Atoms[0][0].NodeIndex; i <= Atoms[nAtom[0] − 1][0].NodeIndex; i
++)
        {
                //picking coordinates of residue with index i.

                len1 = 0;
                s = 0;
                for (k = 0; k<nAtom[0]; k++)
                {
                        if (Atoms[k][0].NodeIndex == i)
                        {
                                stack1[len1].x = Atoms[k][0].x;
```

```
                                stack1[len1].y = Atoms[k][0].y;
                                stack1[len1].z = Atoms[k][0].z;
                                stack1[len1].r0 = Atoms[k][0].r0;
                                stack1[len1].Res_Type = Atoms[k][0].Res_Type;
                                stack1[len1].Atom_Type = Atoms[k][0].Atom_Type;
                                stack1[len1].NodeIndex = Atoms[k][0].NodeIndex;
                                stack1[len1].AtomIndex = Atoms[k][0].AtomIndex;
                                strcpy(stack1[len1].AtmName, Atoms[k][0].AtmName
);
                                len1++;
                        }
                }
                for (k = Moved[0].NodeIndex; k <= Moved[nAtom[1] − 1].NodeIndex;
 k++)
                {
                        //picking coordinates of residue with index k

                        len2 = 0;
                        t = 0;
                        for (l = 0; l<nAtom[1]; l++)
                        {
                                if (Moved[l].NodeIndex == k)
                                {
                                        stack2[len2].x = Moved[l].x;
                                        stack2[len2].y = Moved[l].y;
                                        stack2[len2].z = Moved[l].z;
                                        stack2[len2].r0 = Moved[l].r0;
                                        stack2[len2].Res_Type = Moved[l].Res_Typ
e;
                                        stack2[len2].Atom_Type = Moved[l].Atom_T
ype;
                                        stack2[len2].NodeIndex = Moved[l].NodeIn
dex;
                                        stack2[len2].AtomIndex = Moved[l].AtomIn
dex;
                                        strcpy(stack2[len2].AtmName, Moved[l].At
mName);
                                        len2++;
                                }
                        }

                        for (n=0;n<len1;n++)
                        {
                                for (o=0;o<len2;o++)
                                {
                                        dist.r01 = HARDCORE_R*(stack1[n].r0 + st
ack2[o].r0);
                                        dist.r02 = CONTACT_R*(stack1[n].r0 + sta
ck2[o].r0);
                                        dist.rsq = (stack2[o].x − stack1[n].x)*(
stack2[o].x − stack1[n].x) + (stack2[o].y − stack1[n].y)*(stack2[o].y − stack1[n
].y) + (stack2[o].z − stack1[n].z)*(stack2[o].z − stack1[n].z);


                                        if ((dist.rsq>=dist.r01*dist.r01)&&(dist
.rsq<=dist.r02*dist.r02))
                                        {
                                                for (p = 0; p < LENCHARGESFILE;
p++)
                                                {
                                                        if ((strcmp(stack1[n].At
mName,Charges[p].AtmName) == 0) && (stack1[n].Res_Type == Charges[p].Res_Type))
                                                        {
                                                                for (r = 0; r <
LENCHARGESFILE; r++)
                                                                {
                                                                        if ((str
cmp(stack2[o].AtmName,Charges[r].AtmName) == 0) && (stack2[o].Res_Type == Charge
s[r].Res_Type))
```



```
                                                                        {

q0[counter] = Charges[p].Atm_Charge; //GROMOS54A7 point charges of each interact
ing atom

q1[counter] = Charges[r].Atm_Charge; //GROMOS54A7 point charges of each interact
ing atom
                                                                        }
                                                                }
                                                        }

                                                }
                                                if (q0[counter] < 0.0 & q1[count
er] < 0.0)
                                                {
                                                        electrostatic++;
                                                        it[counter] = 1; /*possi
ble electrostatic interaction*/
                                                        ct0[counter] = stack1[n]
.Res_Type;
                                                        ct1[counter] = stack2[o]
.Res_Type;
                                                        ro0[counter] = stack1[n]
.r0;
                                                        ro1[counter] = stack2[o]
.r0;
                                                        ri0[counter] = stack1[n]
.NodeIndex;
                                                        ri1[counter] = stack2[o]
.NodeIndex;
                                                        rd[counter] = sqrt(dist.
rsq);

                                                        //HYDROPHOBIC interactio
ns between non−hydrogen side−chain atoms

                                                        if (((strcmp(stack1[n].A
tmName,"N") != 0) && (strcmp(stack1[n].AtmName,"H") != 0) && (strcmp(stack1[n].
AtmName,"H1") != 0) &&
                                                        (strcmp(stack1[n].AtmNam
e,"H2") != 0) && (strcmp(stack1[n].AtmName,"H3") != 0) && (strcmp(stack1[n].Atm
Name,"CA") != 0) &&
                                                        (strcmp(stack1[n].AtmNam
e,"C") != 0) && (strcmp(stack1[n].AtmName,"O") != 0) && (strcmp(stack1[n].AtmNa
me,"O1") != 0) &&
                                                        (strcmp(stack1[n].AtmNam
e,"O2") != 0)) && ((strcmp(stack2[o].AtmName,"N") != 0) && (strcmp(stack2[o].At
mName,"H") != 0) && (strcmp(stack2[o].AtmName,"H1") != 0) &&
                                                        (strcmp(stack2[o].AtmNam
e,"H2") != 0) && (strcmp(stack2[o].AtmName,"H3") != 0) && (strcmp(stack2[o].Atm
Name,"CA") != 0) &&
                                                        (strcmp(stack2[o].AtmNam
e,"C") != 0) && (strcmp(stack2[o].AtmName,"O") != 0) && (strcmp(stack2[o].AtmNa
me,"O1") != 0) && (strcmp(stack2[o].AtmName,"O2") != 0)))
                                                        {
                                                                if (strncmp(stac
k1[n].AtmName,"H", 1) != 0 && strncmp(stack2[o].AtmName,"H", 1) != 0)
                                                                {
                                                                        it[count
er] = 3; /*possible electrostatic and hydropathic interaction*/
                                                                        ct0[coun
ter] = stack1[n].Res_Type;
                                                                        ct1[coun
ter] = stack2[o].Res_Type;
                                                                        at0[coun
ter] = stack1[n].Atom_Type;
                                                                        at1[coun
ter] = stack2[o].Atom_Type;
```

```
                                                                        //Assign
ing the ATOMIC SOLVATION PARAMETERS(Cummings, 1995) to the different atom types

                                                                        if ((str
ncmp(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2") == 0)
 || (strcmp(stack1[n].AtmName,"NZ") == 0))) //fully charged group
                                                                        {

stack1[n].ASP_Type = 4;
                                                                        }
                                                                        else if
((strncmp(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2")
!= 0) && (strcmp(stack1[n].AtmName,"NZ") != 0)))
                                                                        {

stack1[n].ASP_Type = 1;
                                                                        }
                                                                        else if
((strncmp(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2")
== 0) || (strcmp(stack1[n].AtmName,"OE2") == 0))) //fully charged group
                                                                        {

stack1[n].ASP_Type = 3;
                                                                        }
                                                                        else if
((strncmp(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2")
!= 0) && (strcmp(stack1[n].AtmName,"OE2") != 0)))
                                                                        {

stack1[n].ASP_Type = 1;
                                                                        }
                                                                        else if
(strncmp(stack1[n].AtmName,"C", 1) == 0)
                                                                        {

stack1[n].ASP_Type = 0;
                                                                        }
                                                                        else if
(strncmp(stack1[n].AtmName,"S", 1) == 0)
                                                                        {

stack1[n].ASP_Type = 2;
                                                                        }
                                                                        else
                                                                        {

stack1[n].ASP_Type = 5;
                                                                        }

                                                                        if ((str
ncmp(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2") == 0)
 || (strcmp(stack2[o].AtmName,"NZ") == 0))) //fully charged group
                                                                        {

stack2[o].ASP_Type = 4;
                                                                        }
                                                                        else if
((strncmp(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2")
!= 0) && (strcmp(stack2[o].AtmName,"NZ") != 0)))
                                                                        {

stack2[o].ASP_Type = 1;
                                                                        }
                                                                        else if
((strncmp(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2")
== 0) || (strcmp(stack2[o].AtmName,"OE2") == 0))) //fully charged group
                                                                        {

stack2[o].ASP_Type = 3;
```



```
                                                                        }
                                                                        else if
((strncmp(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2")
!= 0) && (strcmp(stack2[o].AtmName,"OE2") != 0)))
                                                                        {

stack2[o].ASP_Type = 1;
                                                                        }
                                                                        else if
(strncmp(stack2[o].AtmName,"C", 1) == 0)
                                                                        {

stack2[o].ASP_Type = 0;
                                                                        }
                                                                        else if
(strncmp(stack2[o].AtmName,"S", 1) == 0)
                                                                        {

stack2[o].ASP_Type = 2;
                                                                        }


                                                                        else
                                                                        {

stack2[o].ASP_Type = 5;
                                                                        }
                                                                        asp0[cou
nter] = stack1[n].ASP_Type;
                                                                        asp1[cou
nter] = stack2[o].ASP_Type;
                                                                        ro0[coun
ter] = stack1[n].r0;
                                                                        ro1[coun
ter] = stack2[o].r0;
                                                                        ri0[coun
ter] = stack1[n].NodeIndex;
                                                                        ri1[coun
ter] = stack2[o].NodeIndex;
                                                                        rd[count
er] = sqrt(dist.rsq);
                                                                                }
                                                                        }
                                                                        counter++;


                                                                }
                                                                else if (q0[counter] > 0.0 & q1[
counter] > 0.0)
                                                                {
                                                                        electrostatic++;
                                                                        it[counter] = 1; /*possi
ble electrostatic interaction*/
                                                                        ct0[counter] = stack1[n]
.Res_Type;
                                                                        ct1[counter] = stack2[o]
.Res_Type;
                                                                        ro0[counter] = stack1[n]
.r0;
                                                                        ro1[counter] = stack2[o]
.r0;
                                                                        ri0[counter] = stack1[n]
.NodeIndex;
                                                                        ri1[counter] = stack2[o]
.NodeIndex;
                                                                        rd[counter] = sqrt(dist.
rsq);

                                                                        //HYDROPHOBIC interactio
ns between non−hydrogen side−chain atoms
```

```
                                                    if (((strcmp(stack1[n].A
tmName,"N") != 0) && (strcmp(stack1[n].AtmName,"H") != 0) && (strcmp(stack1[n].
AtmName,"H1") != 0) &&
                                                        (strcmp(stack1[n].AtmNam
e,"H2") != 0) && (strcmp(stack1[n].AtmName,"H3") != 0) && (strcmp(stack1[n].Atm
Name,"CA") != 0) &&
                                                        (strcmp(stack1[n].AtmNam
e,"C") != 0) && (strcmp(stack1[n].AtmName,"O") != 0) && (strcmp(stack1[n].AtmNa
me,"O1") != 0) &&
                                                        (strcmp(stack1[n].AtmNam
e,"O2") != 0)) && ((strcmp(stack2[o].AtmName,"N") != 0) && (strcmp(stack2[o].At
mName,"H") != 0) && (strcmp(stack2[o].AtmName,"H1") != 0) &&
                                                        (strcmp(stack2[o].AtmNam
e,"H2") != 0) && (strcmp(stack2[o].AtmName,"H3") != 0) && (strcmp(stack2[o].Atm
Name,"CA") != 0) &&
                                                        (strcmp(stack2[o].AtmNam
e,"C") != 0) && (strcmp(stack2[o].AtmName,"O") != 0) && (strcmp(stack2[o].AtmNa
me,"O1") != 0) && (strcmp(stack2[o].AtmName,"O2") != 0)))
                                                    {
                                                        if (strncmp(stac
k1[n].AtmName,"H", 1) != 0 && strncmp(stack2[o].AtmName,"H", 1) != 0)
                                                        {
                                                            it[count
er] = 3; /*possible electrostatic and hydropathic interaction*/
                                                            ct0[coun
ter] = stack1[n].Res_Type;
                                                            ct1[coun
ter] = stack2[o].Res_Type;
                                                            at0[coun
ter] = stack1[n].Atom_Type;
                                                            at1[coun
ter] = stack2[o].Atom_Type;

                                                            //Assign
ing the ATOMIC SOLVATION PARAMETERS(Cummings, 1995) to the different atom types

                                                            if ((str
ncmp(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2") == 0)
 || (strcmp(stack1[n].AtmName,"NZ") == 0))) //fully charged group
                                                            {

stack1[n].ASP_Type = 4;
                                                            }
                                                            else if
((strncmp(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2")
!= 0) && (strcmp(stack1[n].AtmName,"NZ") != 0)))
                                                            {

stack1[n].ASP_Type = 1;
                                                            }
                                                            else if
((strncmp(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2")
== 0) || (strcmp(stack1[n].AtmName,"OE2") == 0))) //fully charged group
                                                            {

stack1[n].ASP_Type = 3;
                                                            }
                                                            else if
((strncmp(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2")
!= 0) && (strcmp(stack1[n].AtmName,"OE2") != 0)))
                                                            {

stack1[n].ASP_Type = 1;
                                                            }
                                                            else if
(strncmp(stack1[n].AtmName,"C", 1) == 0)
                                                            {
```



```
stack1[n].ASP_Type = 0;
                                                            }
                                                            else if
(strncmp(stack1[n].AtmName,"S", 1) == 0)
                                                            {

stack1[n].ASP_Type = 2;
                                                            }
                                                            else
                                                            {

stack1[n].ASP_Type = 5;
                                                            }

                                                            if ((str
ncmp(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2") == 0)
 || (strcmp(stack2[o].AtmName,"NZ") == 0))) //fully charged group
                                                            {

stack2[o].ASP_Type = 4;
                                                            }
                                                            else if
((strncmp(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2")
!= 0) && (strcmp(stack2[o].AtmName,"NZ") != 0)))
                                                            {

stack2[o].ASP_Type = 1;
                                                            }
                                                            else if
((strncmp(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2")
== 0) || (strcmp(stack2[o].AtmName,"OE2") == 0))) //fully charged group
                                                            {

stack2[o].ASP_Type = 3;
                                                            }
                                                            else if
((strncmp(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2")
!= 0) && (strcmp(stack2[o].AtmName,"OE2") != 0)))
                                                            {

stack2[o].ASP_Type = 1;
                                                            }
                                                            else if
(strncmp(stack2[o].AtmName,"C", 1) == 0)
                                                            {

stack2[o].ASP_Type = 0;
                                                            }
                                                            else if
(strncmp(stack2[o].AtmName,"S", 1) == 0)
                                                            {

stack2[o].ASP_Type = 2;
                                                            }

                                                            else
                                                            {

stack2[o].ASP_Type = 5;
                                                            }
                                                            asp0[cou
nter] = stack1[n].ASP_Type;
                                                            asp1[cou
nter] = stack2[o].ASP_Type;
                                                            ro0[coun
ter] = stack1[n].r0;
                                                            ro1[coun
ter] = stack2[o].r0;
                                                            ri0[coun
```

```
ter] = stack1[n].NodeIndex;
                                                        ri1[coun
ter] = stack2[o].NodeIndex;
                                                        rd[count
er] = sqrt(dist.rsq);
                                                }
                                        }
                                        counter++;


                                }
                                else if (q0[counter] < 0.0 & q1[
counter] > 0.0)
                                {
                                        electrostatic++;
                                        it[counter] = 1; /*possi
ble electrostatic interaction*/
                                        ct0[counter] = stack1[n]
.Res_Type;
                                        ct1[counter] = stack2[o]
.Res_Type;
                                        ro0[counter] = stack1[n]
.r0;
                                        ro1[counter] = stack2[o]
.r0;
                                        ri0[counter] = stack1[n]
.NodeIndex;
                                        ri1[counter] = stack2[o]
.NodeIndex;
                                        rd[counter] = sqrt(dist.
rsq);

                                        //ASSESSING the presence
 of HYDROGEN BONDS: CHECKING if the DONOR−ACCEPTOR DISTANCE IS <=3.2 Angstrom

                                        if (((strncmp(stack1[n].
AtmName,"O",1) == 0) || (strncmp(stack1[n].AtmName,"N",1) == 0) || (strncmp(sta
ck1[n].AtmName,"S",1) == 0)) && (strncmp(stack2[o].AtmName,"H",1) == 0))
                                        {
                                                if ((strcmp(stac
k2[o].AtmName,"H2") == 0) || (strcmp(stack2[o].AtmName,"HH12") == 0) || (strcmp
(stack2[o].AtmName,"HH22") == 0) || (strcmp(stack2[o].AtmName,"HD22") == 0) ||
(strcmp(stack2[o].AtmName,"HE22") == 0) || (strcmp(stack2[o].AtmName,"HZ2") == 0
))
                                                {
                                                        if (strn
cmp(stack2[o−2].AtmName,"N",1) == 0)
                                                        {

AtomIndex_1 = stack1[n].AtomIndex;

AtomIndex_2 = stack2[o−2].AtomIndex;

if ((strcmp(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB") != 0) && (strcmp(Moved[Ato
mIndex_2−1].AtmHBlabel,"HB") != 0))

{

dDA = sqrt((stack2[o−2].x − stack1[n].x)*(stack2[o−2].x − stack1[n].x) + (stack2
[o−2].y −

stack1[n].y)*(stack2[o−2].y − stack1[n].y) + (stack2[o−2].z − stack1[n].z)*(stac
k2[o−2].z −

stack1[n].z));

if (dDA<=3.2)

{
```



```
        it[counter] = 4; /*possible electrostatic and hydrogen bond interaction*
/

        strcpy(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB");

        strcpy(Moved[AtomIndex_2−1].AtmHBlabel,"HB");

}

}
                                                        }
                                                }
                                                else if ((strcmp
(stack2[o].AtmName,"H3") == 0) || (strcmp(stack2[o].AtmName,"HZ3") == 0))
                                                {
                                                        if (strn
cmp(stack2[o−3].AtmName,"N",1) == 0)
                                                        {

AtomIndex_1 = stack1[n].AtomIndex;

AtomIndex_2 = stack2[o−3].AtomIndex;

if ((strcmp(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB") != 0) && (strcmp(Moved[Ato
mIndex_2−1].AtmHBlabel,"HB") != 0))

{

dDA = sqrt((stack2[o−3].x − stack1[n].x)*(stack2[o−3].x − stack1[n].x) + (stack2
[o−3].y −

stack1[n].y)*(stack2[o−3].y − stack1[n].y) + (stack2[o−3].z − stack1[n].z)*(stac
k2[o−3].z −

stack1[n].z));

if (dDA<=3.2)

{

        it[counter] = 4; /*possible electrostatic and hydrogen bond interaction*
/

        strcpy(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB");

        strcpy(Moved[AtomIndex_2−1].AtmHBlabel,"HB");

}

}
                                                        }
                                                }
                                                else
                                                {
                                                        if ((str
ncmp(stack2[o−1].AtmName,"N",1) == 0) || (strncmp(stack2[o−1].AtmName,"O",1) ==
 0))
                                                        {

AtomIndex_1 = stack1[n].AtomIndex;

AtomIndex_2 = stack2[o−1].AtomIndex;

if ((strcmp(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB") != 0) && (strcmp(Moved[Ato
mIndex_2−1].AtmHBlabel,"HB") != 0))

{
```

```
dDA = sqrt((stack2[o−1].x − stack1[n].x)*(stack2[o−1].x − stack1[n].x) + (stack2
[o−1].y −

stack1[n].y)*(stack2[o−1].y − stack1[n].y) + (stack2[o−1].z − stack1[n].z)*(stac
k2[o−1].z −

stack1[n].z));

if (dDA<=3.2)

{

        it[counter] = 4; /*possible electrostatic and hydrogen bond interaction*
/

        strcpy(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB");

        strcpy(Moved[AtomIndex_2−1].AtmHBlabel,"HB");

}

}
                                                                    }
                                                                }

                                                            }

                                                            //HYDROPHOBIC interactio
ns between non−hydrogen side−chain atoms

                                                            if (((strcmp(stack1[n].A
tmName,"N") != 0) && (strcmp(stack1[n].AtmName,"H") != 0) && (strcmp(stack1[n].
AtmName,"H1") != 0) &&
                                                                (strcmp(stack1[n].AtmNam
e,"H2") != 0) && (strcmp(stack1[n].AtmName,"H3") != 0) && (strcmp(stack1[n].Atm
Name,"CA") != 0) &&
                                                                (strcmp(stack1[n].AtmNam
e,"C") != 0) && (strcmp(stack1[n].AtmName,"O") != 0) && (strcmp(stack1[n].AtmNa
me,"O1") != 0) &&
                                                                (strcmp(stack1[n].AtmNam
e,"O2") != 0)) && ((strcmp(stack2[o].AtmName,"N") != 0) && (strcmp(stack2[o].At
mName,"H") != 0) && (strcmp(stack2[o].AtmName,"H1") != 0) &&
                                                                (strcmp(stack2[o].AtmNam
e,"H2") != 0) && (strcmp(stack2[o].AtmName,"H3") != 0) && (strcmp(stack2[o].Atm
Name,"CA") != 0) &&
                                                                (strcmp(stack2[o].AtmNam
e,"C") != 0) && (strcmp(stack2[o].AtmName,"O") != 0) && (strcmp(stack2[o].AtmNa
me,"O1") != 0) && (strcmp(stack2[o].AtmName,"O2") != 0)))
                                                                {
                                                                    if (strncmp(stac
k1[n].AtmName,"H", 1) != 0 && strncmp(stack2[o].AtmName,"H", 1) != 0)
                                                                    {
                                                                        it[count
er] = 3; /*possible electrostatic and hydropathic interaction*/
                                                                        ct0[coun
ter] = stack1[n].Res_Type;
                                                                        ct1[coun
ter] = stack2[o].Res_Type;
                                                                        at0[coun
ter] = stack1[n].Atom_Type;
                                                                        at1[coun
ter] = stack2[o].Atom_Type;

                                                                        //Assign
ing the ATOMIC SOLVATION PARAMETERS(Cummings, 1995) to the different atom types

                                                                        if ((str
ncmp(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2") == 0)
 || (strcmp(stack1[n].AtmName,"NZ") == 0))) //fully charged group
```



```
                                                                        {

stack1[n].ASP_Type = 4;

                                                                        }
                                                                        else if
((strncmp(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2")
!= 0) && (strcmp(stack1[n].AtmName,"NZ") != 0)))
                                                                        {

stack1[n].ASP_Type = 1;

                                                                        }
                                                                        else if
((strncmp(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2")
== 0) || (strcmp(stack1[n].AtmName,"OE2") == 0))) //fully charged group
                                                                        {

stack1[n].ASP_Type = 3;

                                                                        }
                                                                        else if
((strncmp(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2")
!= 0) && (strcmp(stack1[n].AtmName,"OE2") != 0)))
                                                                        {

stack1[n].ASP_Type = 1;

                                                                        }
                                                                        else if
(strncmp(stack1[n].AtmName,"C", 1) == 0)
                                                                        {

stack1[n].ASP_Type = 0;

                                                                        }
                                                                        else if
(strncmp(stack1[n].AtmName,"S", 1) == 0)
                                                                        {

stack1[n].ASP_Type = 2;

                                                                        }
                                                                        else
                                                                        {

stack1[n].ASP_Type = 5;

                                                                        }

                                                                        if ((str
ncmp(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2") == 0)
 || (strcmp(stack2[o].AtmName,"NZ") == 0))) //fully charged group
                                                                        {

stack2[o].ASP_Type = 4;

                                                                        }
                                                                        else if
((strncmp(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2")
!= 0) && (strcmp(stack2[o].AtmName,"NZ") != 0)))
                                                                        {

stack2[o].ASP_Type = 1;

                                                                        }
                                                                        else if
((strncmp(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2")
== 0) || (strcmp(stack2[o].AtmName,"OE2") == 0))) //fully charged group
                                                                        {

stack2[o].ASP_Type = 3;

                                                                        }
                                                                        else if
((strncmp(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2")
!= 0) && (strcmp(stack2[o].AtmName,"OE2") != 0)))
                                                                        {
```

```
stack2[o].ASP_Type = 1;
                                                                }
                                                                else if
(strncmp(stack2[o].AtmName,"C", 1) == 0)
                                                                {
stack2[o].ASP_Type = 0;
                                                                }
                                                                else if
(strncmp(stack2[o].AtmName,"S", 1) == 0)
                                                                {
stack2[o].ASP_Type = 2;
                                                                }
                                                                else
                                                                {
stack2[o].ASP_Type = 5;
                                                                }
                                                                asp0[cou
nter] = stack1[n].ASP_Type;                                     asp1[cou
nter] = stack2[o].ASP_Type;                                     ro0[coun
ter] = stack1[n].r0;                                            ro1[coun
ter] = stack2[o].r0;                                            ri0[coun
ter] = stack1[n].NodeIndex;                                     ri1[coun
ter] = stack2[o].NodeIndex;                                     rd[count
er] = sqrt(dist.rsq);
                                                        }
                                                }
                                                counter++;


                                        }
                                        else if (q0[counter] > 0.0 & q1[
counter] < 0.0)
                                        {
                                                electrostatic++;
                                                it[counter] = 1; /*possi
ble electrostatic interaction*/
                                                ct0[counter] = stack1[n]
.Res_Type;
                                                ct1[counter] = stack2[o]
.Res_Type;
                                                ro0[counter] = stack1[n]
.r0;
                                                ro1[counter] = stack2[o]
.r0;
                                                ri0[counter] = stack1[n]
.NodeIndex;
                                                ri1[counter] = stack2[o]
.NodeIndex;
                                                rd[counter] = sqrt(dist.
rsq);

                                                //ASSESSING the presence
 of HYDROGEN BONDS: CHECKING if the DONOR−ACCEPTOR DISTANCE IS <=3.2 Angstrom

                                                if ((strncmp(stack1[n].A
tmName,"H",1) == 0) && ((strncmp(stack2[o].AtmName,"O",1) == 0) || (strncmp(sta
ck2[o].AtmName,"N",1) == 0) || (strncmp(stack2[o].AtmName,"S",1) == 0)))
                                                {
                                                        if ((strcmp(stac
```



```
k1[n].AtmName,"H2") == 0) || (strcmp(stack1[n].AtmName,"HH12") == 0) || (strcmp
(stack1[n].AtmName,"HH22") == 0) || (strcmp(stack1[n].AtmName,"HD22") == 0) ||
(strcmp(stack1[n].AtmName,"HE22") == 0) || (strcmp(stack1[n].AtmName,"HZ2") == 0
))
                                                                {
                                                                        if (strn
cmp(stack1[n−2].AtmName,"N",1) == 0)
                                                                        {
AtomIndex_1 = stack1[n−2].AtomIndex;

AtomIndex_2 = stack2[o].AtomIndex;

if ((strcmp(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB") != 0) && (strcmp(Moved[Ato
mIndex_2−1].AtmHBlabel,"HB") != 0))

{

dDA = sqrt((stack2[o].x − stack1[n−2].x)*(stack2[o].x − stack1[n−2].x) + (stack2
[o].y −

stack1[n−2].y)*(stack2[o].y − stack1[n−2].y) + (stack2[o].z − stack1[n−2].z)*(st
ack2[o].z −

stack1[n−2].z));

if (dDA<=3.2)

{

        it[counter] = 4; /*possible electrostatic and hydrogen bond interaction*
/

        strcpy(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB");

        strcpy(Moved[AtomIndex_2−1].AtmHBlabel,"HB");

}

}
                                                                        }
                                                                }
                                                                else if ((strcmp
(stack1[n].AtmName,"H3") == 0) || (strcmp(stack1[n].AtmName,"HZ3") == 0))
                                                                {
                                                                        if (strn
cmp(stack1[n−3].AtmName,"N",1) == 0)
                                                                        {

AtomIndex_1 = stack1[n−3].AtomIndex;

AtomIndex_2 = stack2[o].AtomIndex;

if ((strcmp(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB") != 0) && (strcmp(Moved[Ato
mIndex_2−1].AtmHBlabel,"HB") != 0))

{

dDA = sqrt((stack2[o].x − stack1[n−3].x)*(stack2[o].x − stack1[n−3].x) + (stack2
[o].y −

stack1[n−3].y)*(stack2[o].y − stack1[n−3].y) + (stack2[o].z − stack1[n−3].z)*(st
ack2[o].z −

stack1[n−3].z));

if (dDA<=3.2)

{
```

```
        it[counter] = 4; /*possible electrostatic and hydrogen bond interaction*
/

        strcpy(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB");

        strcpy(Moved[AtomIndex_2−1].AtmHBlabel,"HB");

}

}
                                                                    }
                                                                }
                                                                else
                                                                {
                                                                    if ((str
ncmp(stack1[n−1].AtmName,"N",1) == 0) || (strncmp(stack1[n−1].AtmName,"O",1) ==
 0))
                                                                    {

AtomIndex_1 = stack1[n−1].AtomIndex;

AtomIndex_2 = stack2[o].AtomIndex;

if ((strcmp(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB") != 0) && (strcmp(Moved[Ato
mIndex_2−1].AtmHBlabel,"HB") != 0))

{

dDA = sqrt((stack2[o].x − stack1[n−1].x)*(stack2[o].x − stack1[n−1].x) + (stack2
[o].y −

stack1[n−1].y)*(stack2[o].y − stack1[n−1].y) + (stack2[o].z − stack1[n−1].z)*(st
ack2[o].z −

stack1[n−1].z));

if (dDA<=3.2)

{

        it[counter] = 4; /*possible electrostatic and hydrogen bond interaction*
/

        strcpy(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB");

        strcpy(Moved[AtomIndex_2−1].AtmHBlabel,"HB");

}

}
                                                                    }
                                                                }
                                                            }

                                                            //HYDROPHOBIC interactio
ns between non−hydrogen side−chain atoms

                                                            if (((strcmp(stack1[n].A
tmName,"N") != 0) && (strcmp(stack1[n].AtmName,"H") != 0) && (strcmp(stack1[n].
AtmName,"H1") != 0) &&
                                                                (strcmp(stack1[n].AtmNam
e,"H2") != 0) && (strcmp(stack1[n].AtmName,"H3") != 0) && (strcmp(stack1[n].Atm
Name,"CA") != 0) &&
                                                                (strcmp(stack1[n].AtmNam
e,"C") != 0) && (strcmp(stack1[n].AtmName,"O") != 0) && (strcmp(stack1[n].AtmNa
me,"O1") != 0) &&
                                                                (strcmp(stack1[n].AtmNam
e,"O2") != 0)) && ((strcmp(stack2[o].AtmName,"N") != 0) && (strcmp(stack2[o].At
```



```
mName,"H") != 0) && (strcmp(stack2[o].AtmName,"H1") != 0) &&
                                                                (strcmp(stack2[o].AtmNam
e,"H2") != 0) && (strcmp(stack2[o].AtmName,"H3") != 0) && (strcmp(stack2[o].Atm
Name,"CA") != 0) &&
                                                                (strcmp(stack2[o].AtmNam
e,"C") != 0) && (strcmp(stack2[o].AtmName,"O") != 0) && (strcmp(stack2[o].AtmNa
me,"O1") != 0) && (strcmp(stack2[o].AtmName,"O2") != 0)))
                                                                {
                                                                    if (strncmp(stac
k1[n].AtmName,"H", 1) != 0 && strncmp(stack2[o].AtmName,"H", 1) != 0)
                                                                    {
                                                                        it[count
er] = 3; /*possible electrostatic and hydropathic interaction*/
                                                                        ct0[coun
ter] = stack1[n].Res_Type;
                                                                        ct1[coun
ter] = stack2[o].Res_Type;
                                                                        at0[coun
ter] = stack1[n].Atom_Type;
                                                                        at1[coun
ter] = stack2[o].Atom_Type;

                                                                        //Assign
ing the ATOMIC SOLVATION PARAMETERS(Cummings, 1995) to the different atom types

                                                                        if ((str
ncmp(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2") == 0)
 || (strcmp(stack1[n].AtmName,"NZ") == 0))) //fully charged group
                                                                        {

stack1[n].ASP_Type = 4;
                                                                        }
                                                                        else if
((strncmp(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2")
!= 0) && (strcmp(stack1[n].AtmName,"NZ") != 0)))
                                                                        {

stack1[n].ASP_Type = 1;
                                                                        }
                                                                        else if
((strncmp(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2")
== 0) || (strcmp(stack1[n].AtmName,"OE2") == 0))) //fully charged group
                                                                        {

stack1[n].ASP_Type = 3;
                                                                        }
                                                                        else if
((strncmp(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2")
!= 0) && (strcmp(stack1[n].AtmName,"OE2") != 0)))
                                                                        {

stack1[n].ASP_Type = 1;
                                                                        }
                                                                        else if
(strncmp(stack1[n].AtmName,"C", 1) == 0)
                                                                        {

stack1[n].ASP_Type = 0;
                                                                        }
                                                                        else if
(strncmp(stack1[n].AtmName,"S", 1) == 0)
                                                                        {

stack1[n].ASP_Type = 2;
                                                                        }
                                                                        else
                                                                        {


stack1[n].ASP_Type = 5;
```

```
                                                                        }

                                                                        if ((str
ncmp(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2") == 0)
 || (strcmp(stack2[o].AtmName,"NZ") == 0))) //fully charged group
                                                                        {

stack2[o].ASP_Type = 4;
                                                                        }
                                                                        else if
((strncmp(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2")
!= 0) && (strcmp(stack2[o].AtmName,"NZ") != 0)))
                                                                        {

stack2[o].ASP_Type = 1;
                                                                        }
                                                                        else if
((strncmp(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2")
== 0) || (strcmp(stack2[o].AtmName,"OE2") == 0))) //fully charged group
                                                                        {

stack2[o].ASP_Type = 3;
                                                                        }
                                                                        else if
((strncmp(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2")
!= 0) && (strcmp(stack2[o].AtmName,"OE2") != 0)))
                                                                        {

stack2[o].ASP_Type = 1;
                                                                        }
                                                                        else if
(strncmp(stack2[o].AtmName,"C", 1) == 0)
                                                                        {

stack2[o].ASP_Type = 0;
                                                                        }
                                                                        else if
(strncmp(stack2[o].AtmName,"S", 1) == 0)
                                                                        {

stack2[o].ASP_Type = 2;
                                                                        }



                                                                        else
                                                                        {

stack2[o].ASP_Type = 5;
                                                                        }
                                                                        asp0[cou
nter] = stack1[n].ASP_Type;                                             asp1[cou

nter] = stack2[o].ASP_Type;                                             ro0[coun

ter] = stack1[n].r0;                                                    ro1[coun

ter] = stack2[o].r0;                                                    ri0[coun

ter] = stack1[n].NodeIndex;                                             ri1[coun

ter] = stack2[o].NodeIndex;                                             rd[count

er] = sqrt(dist.rsq);
                                                                                }
                                                                        }
                                                                        counter++;



                                                                }
```



```
                                                                        //HYDROPHOBIC interactions betwe
en non−hydrogen side−chain atoms



                                                                        else if (((strcmp(stack1[n].AtmN
ame,"N") != 0) && (strcmp(stack1[n].AtmName,"H") != 0) && (strcmp(stack1[n].Atm
Name,"H1") != 0) &&
                                                                         (strcmp(stack1[n].AtmName,"H2")
 != 0) && (strcmp(stack1[n].AtmName,"H3") != 0) && (strcmp(stack1[n].AtmName,"C
A") != 0) &&
                                                                         (strcmp(stack1[n].AtmName,"C")
!= 0) && (strcmp(stack1[n].AtmName,"O") != 0) && (strcmp(stack1[n].AtmName,"O1"
) != 0) &&
                                                                         (strcmp(stack1[n].AtmName,"O2")
 != 0)) && ((strcmp(stack2[o].AtmName,"N") != 0) && (strcmp(stack2[o].AtmName,"
H") != 0) && (strcmp(stack2[o].AtmName,"H1") != 0) &&
                                                                         (strcmp(stack2[o].AtmName,"H2")
 != 0) && (strcmp(stack2[o].AtmName,"H3") != 0) && (strcmp(stack2[o].AtmName,"C
A") != 0) &&
                                                                         (strcmp(stack2[o].AtmName,"C")
!= 0) && (strcmp(stack2[o].AtmName,"O") != 0) && (strcmp(stack2[o].AtmName,"O1"
) != 0) && (strcmp(stack2[o].AtmName,"O2") != 0)))
                                                                        {
                                                                                if (strncmp(stack1[n].At
mName,"H", 1) != 0 && strncmp(stack2[o].AtmName,"H", 1) != 0)
                                                                                {
                                                                                        it[counter] = 2;
 /*possible hydropathic interaction*/
                                                                                        ct0[counter] = s
tack1[n].Res_Type;
                                                                                        ct1[counter] = s
tack2[o].Res_Type;
                                                                                        at0[counter] = s
tack1[n].Atom_Type;
                                                                                        at1[counter] = s
tack2[o].Atom_Type;

                                                                                        //Assigning the
ATOMIC SOLVATION PARAMETERS(Cummings, 1995) to the different atom types

                                                                                        if ((strncmp(sta
ck1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2") == 0) || (str
cmp(stack1[n].AtmName,"NZ") == 0))) //fully charged group
                                                                                        {
                                                                                                stack1[n
].ASP_Type = 4;
                                                                                        }
                                                                                        else if ((strncm
p(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2") != 0) &&
 (strcmp(stack1[n].AtmName,"NZ") != 0)))
                                                                                        {
                                                                                                stack1[n
].ASP_Type = 1;
                                                                                        }
                                                                                        else if ((strncm
p(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2") == 0) ||
 (strcmp(stack1[n].AtmName,"OE2") == 0))) //fully charged group
                                                                                        {
                                                                                                stack1[n
].ASP_Type = 3;
                                                                                        }
                                                                                        else if ((strncm
p(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2") != 0) &&
 (strcmp(stack1[n].AtmName,"OE2") != 0)))
                                                                                        {
                                                                                                stack1[n
].ASP_Type = 1;
                                                                                        }
```

```
                                                        else if (strncmp
(stack1[n].AtmName,"C", 1) == 0)
                                                        {
                                                                stack1[n
].ASP_Type = 0;
                                                        }
                                                        else if (strncmp
(stack1[n].AtmName,"S", 1) == 0)
                                                        {
                                                                stack1[n
].ASP_Type = 2;
                                                        }
                                                        else
                                                        {
                                                                stack1[n
].ASP_Type = 5;
                                                        }

                                                        if ((strncmp(sta
ck2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2") == 0) || (str
cmp(stack2[o].AtmName,"NZ") == 0))) //fully charged group
                                                        {
                                                                stack2[o
].ASP_Type = 4;
                                                        }
                                                        else if ((strncm
p(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2") != 0) &&
 (strcmp(stack2[o].AtmName,"NZ") != 0)))
                                                        {
                                                                stack2[o
].ASP_Type = 1;
                                                        }
                                                        else if ((strncm
p(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2") == 0) ||
 (strcmp(stack2[o].AtmName,"OE2") == 0))) //fully charged group
                                                        {
                                                                stack2[o
].ASP_Type = 3;
                                                        }
                                                        else if ((strncm
p(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2") != 0) &&
 (strcmp(stack2[o].AtmName,"OE2") != 0)))
                                                        {
                                                                stack2[o
].ASP_Type = 1;
                                                        }
                                                        else if (strncmp
(stack2[o].AtmName,"C", 1) == 0)
                                                        {
                                                                stack2[o
].ASP_Type = 0;
                                                        }
                                                        else if (strncmp
(stack2[o].AtmName,"S", 1) == 0)
                                                        {
                                                                stack2[o
].ASP_Type = 2;
                                                        }

                                                        else
                                                        {
                                                                stack2[o
].ASP_Type = 5;
                                                        }
                                                        asp0[counter] =
stack1[n].ASP_Type;
                                                        asp1[counter] =
stack2[o].ASP_Type;
                                                        ro0[counter] = s
```



```
tack1[n].r0;
                                                        ro1[counter] = s
tack2[o].r0;
                                                        ri0[counter] = s
tack1[n].NodeIndex;
                                                        ri1[counter] = s
tack2[o].NodeIndex;
                                                        rd[counter] = sq
rt(dist.rsq);
                                                        counter++;
                                                                }
                                                        }
                                                }
                                        }
                                }

                        }
                }

        }

        return counter;
        fclose(test);
}


int CONTACTSMOVED(void)
{
        int n,o,counter;

        counter=0;
        for (n=0;n<nAtom[0];n++)
        {
                for (o=0;o<nAtom[1];o++)
                {
                        dist.r01 = HARDCORE_R*(Atoms[n][0].r0+Moved[o].r0);
                        dist.r02 = CONTACT_R*(Atoms[n][0].r0+Moved[o].r0);
                        dist.rsq = (Moved[o].x−Atoms[n][0].x)*(Moved[o].x−Atoms[
n][0].x)+(Moved[o].y−Atoms[n][0].y)*(Moved[o].y−Atoms[n][0].y)+(Moved[o].z−Atoms
[n][0].z)*(Moved[o].z−Atoms[n][0].z);

                        if ((dist.rsq>=dist.r01*dist.r01)&&(dist.rsq<=dist.r02*d
ist.r02))
                        {
                                counter++;
                        }

                }

        }
        return counter;
}


int CLASHESMOVED(void)
{
        int n,o,counter;

        counter=0;
        for (n=0;n<nAtom[0];n++)
        {
                for (o=0;o<nAtom[1];o++)
                {
                        dist.r01 = HARDCORE_R*(Atoms[n][0].r0+Moved[o].r0);
                        dist.r02 = CONTACT_R*(Atoms[n][0].r0+Moved[o].r0);
                        dist.rsq = (Moved[o].x−Atoms[n][0].x)*(Moved[o].x−Atoms[
n][0].x)+(Moved[o].y−Atoms[n][0].y)*(Moved[o].y−Atoms[n][0].y)+(Moved[o].z−Atoms
[n][0].z)*(Moved[o].z−Atoms[n][0].z);

                        if (dist.rsq<(dist.r01*dist.r01))
```

```
                        {
                                counter++;
                        }


                }

        }
        return counter;
}


int TYPECONTACTS(int ct0[MAXCONTACT], int ct1[MAXCONTACT], int at0[MAXCONTACT],
int at1[MAXCONTACT], int ri0[MAXCONTACT], int ri1[MAXCONTACT], double ro0[MAXCON
TACT], double ro1[MAXCONTACT], double rd[MAXCONTACT], int it[MAXCONTACT], float
q0[MAXCONTACT], float q1[MAXCONTACT], int asp0[MAXCONTACT], int asp1[MAXCONTACT]
)
{
        FILE *test;

        int i, k, l, n, o, p, r, maxcontactflag, counter, len1, len2, residue_in
dex_0, residue_index_1, s, t, ss, tt;
        int AtomIndex_1, AtomIndex_2;
        int electrostatic = 0;
        float dDA, dc1, dc2, dc3;


        for (k = 0; k<nAtom[0]; k++)
        {
                strcpy(Atoms[k][0].AtmHBlabel, "NoHB");
        }

        for (l = 0; l<nAtom[1]; l++)
        {
                strcpy(Atoms[l][1].AtmHBlabel, "NoHB");
        }


        counter = 0;
        s = 0;
        t = 0;
        for (i = Atoms[0][0].NodeIndex; i <= Atoms[nAtom[0] − 1][0].NodeIndex; i
++)
        {
                //picking coordinates of residue with index i.

                len1 = 0;
                s = 0;
                for (k = 0; k<nAtom[0]; k++)
                {
                        if (Atoms[k][0].NodeIndex == i)
                        {
                                stack1[len1].x = Atoms[k][0].x;
                                stack1[len1].y = Atoms[k][0].y;
                                stack1[len1].z = Atoms[k][0].z;
                                stack1[len1].r0 = Atoms[k][0].r0;
                                stack1[len1].Res_Type = Atoms[k][0].Res_Type;
                                stack1[len1].Atom_Type = Atoms[k][0].Atom_Type;
                                stack1[len1].NodeIndex = Atoms[k][0].NodeIndex;
                                stack1[len1].AtomIndex = Atoms[k][0].AtomIndex;
                                strcpy(stack1[len1].AtmName, Atoms[k][0].AtmName
);
                                len1++;
                        }
                }
                for (k = Atoms[0][1].NodeIndex; k <= Atoms[nAtom[1] − 1][1].Node
Index; k++)
                {
                        //picking coordinates of residue with index k
```



```
                len2 = 0;
                t = 0;
                for (l = 0; l<nAtom[1]; l++)
                {
                        if (Atoms[l][1].NodeIndex == k)
                        {
                                stack2[len2].x = Atoms[l][1].x;
                                stack2[len2].y = Atoms[l][1].y;
                                stack2[len2].z = Atoms[l][1].z;
                                stack2[len2].r0 = Atoms[l][1].r0;
                                stack2[len2].Res_Type = Atoms[l][1].Res_
Type;
                                stack2[len2].Atom_Type = Atoms[l][1].Ato
m_Type;
                                stack2[len2].NodeIndex = Atoms[l][1].Nod
eIndex;
                                stack2[len2].AtomIndex = Atoms[l][1].Ato
mIndex;
                                strcpy(stack2[len2].AtmName, Atoms[l][1]
.AtmName);
                                len2++;
                        }
                }
                for (n=0;n<len1;n++)
                {
                        for (o=0;o<len2;o++)
                        {
                                dist.r01 = HARDCORE_R*(stack1[n].r0 + st
ack2[o].r0);
                                dist.r02 = CONTACT_R*(stack1[n].r0 + sta
ck2[o].r0);
                                dist.rsq = (stack2[o].x − stack1[n].x)*(
stack2[o].x − stack1[n].x) + (stack2[o].y − stack1[n].y)*(stack2[o].y − stack1[n
].y) + (stack2[o].z − stack1[n].z)*(stack2[o].z − stack1[n].z);

                                if ((dist.rsq>=dist.r01*dist.r01)&&(dist
.rsq<=dist.r02*dist.r02))
                                {
                                        for (p = 0; p < LENCHARGESFILE;
p++)
                                        {
                                                if ((strcmp(stack1[n].At
mName,Charges[p].AtmName) == 0) && (stack1[n].Res_Type == Charges[p].Res_Type))
                                                {
                                                        for (r = 0; r <
LENCHARGESFILE; r++)
                                                        {
                                                                if ((str
cmp(stack2[o].AtmName,Charges[r].AtmName) == 0) && (stack2[o].Res_Type == Charge
s[r].Res_Type))
                                                                {

q0[counter] = Charges[p].Atm_Charge; //GROMOS54A7 point charges of each interact
ing atom

q1[counter] = Charges[r].Atm_Charge; //GROMOS54A7 point charges of each interact
ing atom
                                                                }
                                                        }
                                                }

                                        }
                                        if (q0[counter] < 0.0 & q1[count
er] < 0.0)
                                        {
                                                electrostatic++;
                                                it[counter] = 1; /*possi
ble electrostatic interaction*/
```

```
                                                    ct0[counter] = stack1[n]
.Res_Type;
                                                    ct1[counter] = stack2[o]
.Res_Type;
                                                    ro0[counter] = stack1[n]
.r0;
                                                    ro1[counter] = stack2[o]
.r0;
                                                    ri0[counter] = stack1[n]
.NodeIndex;
                                                    ri1[counter] = stack2[o]
.NodeIndex;
                                                    rd[counter] = sqrt(dist.
rsq);

                                                    //HYDROPHOBIC interactio
ns between non−hydrogen side−chain atoms

                                                    if (((strcmp(stack1[n].A
tmName,"N") != 0) && (strcmp(stack1[n].AtmName,"H") != 0) && (strcmp(stack1[n].
AtmName,"H1") != 0) &&
                                                    (strcmp(stack1[n].AtmNam
e,"H2") != 0) && (strcmp(stack1[n].AtmName,"H3") != 0) && (strcmp(stack1[n].Atm
Name,"CA") != 0) &&
                                                    (strcmp(stack1[n].AtmNam
e,"C") != 0) && (strcmp(stack1[n].AtmName,"O") != 0) && (strcmp(stack1[n].AtmNa
me,"O1") != 0) &&
                                                    (strcmp(stack1[n].AtmNam
e,"O2") != 0)) && ((strcmp(stack2[o].AtmName,"N") != 0) && (strcmp(stack2[o].At
mName,"H") != 0) && (strcmp(stack2[o].AtmName,"H1") != 0) &&
                                                    (strcmp(stack2[o].AtmNam
e,"H2") != 0) && (strcmp(stack2[o].AtmName,"H3") != 0) && (strcmp(stack2[o].Atm
Name,"CA") != 0) &&
                                                    (strcmp(stack2[o].AtmNam
e,"C") != 0) && (strcmp(stack2[o].AtmName,"O") != 0) && (strcmp(stack2[o].AtmNa
me,"O1") != 0) && (strcmp(stack2[o].AtmName,"O2") != 0)))
                                                    {
                                                            if (strncmp(stac
k1[n].AtmName,"H", 1) != 0 && strncmp(stack2[o].AtmName,"H", 1) != 0)
                                                            {
                                                                    it[count
er] = 3; /*possible electrostatic and hydropathic interaction*/
                                                                    ct0[coun
ter] = stack1[n].Res_Type;
                                                                    ct1[coun
ter] = stack2[o].Res_Type;
                                                                    at0[coun
ter] = stack1[n].Atom_Type;
                                                                    at1[coun
ter] = stack2[o].Atom_Type;

                                                                    //Assign
ing the ATOMIC SOLVATION PARAMETERS(Cummings, 1995) to the different atom types

                                                                    if ((str
ncmp(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2") == 0)
 || (strcmp(stack1[n].AtmName,"NZ") == 0))) //fully charged group
                                                                    {

stack1[n].ASP_Type = 4;
                                                                    }
                                                                    else if
((strncmp(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2")
!= 0) && (strcmp(stack1[n].AtmName,"NZ") != 0)))
                                                                    {

stack1[n].ASP_Type = 1;
                                                                    }
                                                                    else if
```



```
((strncmp(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2")
== 0) || (strcmp(stack1[n].AtmName,"OE2") == 0))) //fully charged group
                                                                    {

stack1[n].ASP_Type = 3;
                                                                    }
                                                                    else if
((strncmp(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2")
!= 0) && (strcmp(stack1[n].AtmName,"OE2") != 0)))
                                                                    {

stack1[n].ASP_Type = 1;
                                                                    }
                                                                    else if
(strncmp(stack1[n].AtmName,"C", 1) == 0)
                                                                    {

stack1[n].ASP_Type = 0;
                                                                    }
                                                                    else if
(strncmp(stack1[n].AtmName,"S", 1) == 0)
                                                                    {

stack1[n].ASP_Type = 2;
                                                                    }
                                                                    else
                                                                    {


stack1[n].ASP_Type = 5;
                                                                    }


                                                                    if ((str
ncmp(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2") == 0)
 || (strcmp(stack2[o].AtmName,"NZ") == 0))) //fully charged group
                                                                    {

stack2[o].ASP_Type = 4;
                                                                    }
                                                                    else if
((strncmp(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2")
!= 0) && (strcmp(stack2[o].AtmName,"NZ") != 0)))
                                                                    {

stack2[o].ASP_Type = 1;
                                                                    }
                                                                    else if
((strncmp(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2")
== 0) || (strcmp(stack2[o].AtmName,"OE2") == 0))) //fully charged group
                                                                    {

stack2[o].ASP_Type = 3;
                                                                    }
                                                                    else if
((strncmp(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2")
!= 0) && (strcmp(stack2[o].AtmName,"OE2") != 0)))
                                                                    {

stack2[o].ASP_Type = 1;
                                                                    }
                                                                    else if
(strncmp(stack2[o].AtmName,"C", 1) == 0)
                                                                    {

stack2[o].ASP_Type = 0;
                                                                    }
                                                                    else if
(strncmp(stack2[o].AtmName,"S", 1) == 0)
                                                                    {
```

```
stack2[o].ASP_Type = 2;
                                                        }

                                                        else
                                                        {

stack2[o].ASP_Type = 5;
                                                        }
                                                        asp0[cou
nter] = stack1[n].ASP_Type;
                                                        asp1[cou
nter] = stack2[o].ASP_Type;
                                                        ro0[coun
ter] = stack1[n].r0;
                                                        ro1[coun
ter] = stack2[o].r0;
                                                        ri0[coun
ter] = stack1[n].NodeIndex;
                                                        ri1[coun
ter] = stack2[o].NodeIndex;
                                                        rd[count
er] = sqrt(dist.rsq);
                                                }
                                        }
                                        counter++;



                                }
                                else if (q0[counter] > 0.0 & q1[
counter] > 0.0)
                                {
                                        electrostatic++;
                                        it[counter] = 1; /*possi
ble electrostatic interaction*/
                                        ct0[counter] = stack1[n]
.Res_Type;
                                        ct1[counter] = stack2[o]
.Res_Type;
                                        ro0[counter] = stack1[n]
.r0;
                                        ro1[counter] = stack2[o]
.r0;
                                        ri0[counter] = stack1[n]
.NodeIndex;
                                        ri1[counter] = stack2[o]
.NodeIndex;
                                        rd[counter] = sqrt(dist.
rsq);

                                        //HYDROPHOBIC interactio
ns between non−hydrogen side−chain atoms

                                        if (((strcmp(stack1[n].A
tmName,"N") != 0) && (strcmp(stack1[n].AtmName,"H") != 0) && (strcmp(stack1[n].
AtmName,"H1") != 0) &&
                                        (strcmp(stack1[n].AtmNam
e,"H2") != 0) && (strcmp(stack1[n].AtmName,"H3") != 0) && (strcmp(stack1[n].Atm
Name,"CA") != 0) &&
                                        (strcmp(stack1[n].AtmNam
e,"C") != 0) && (strcmp(stack1[n].AtmName,"O") != 0) && (strcmp(stack1[n].AtmNa
me,"O1") != 0) &&
                                        (strcmp(stack1[n].AtmNam
e,"O2") != 0)) && ((strcmp(stack2[o].AtmName,"N") != 0) && (strcmp(stack2[o].At
mName,"H") != 0) && (strcmp(stack2[o].AtmName,"H1") != 0) &&
                                        (strcmp(stack2[o].AtmNam
e,"H2") != 0) && (strcmp(stack2[o].AtmName,"H3") != 0) && (strcmp(stack2[o].Atm
Name,"CA") != 0) &&
                                        (strcmp(stack2[o].AtmNam
e,"C") != 0) && (strcmp(stack2[o].AtmName,"O") != 0) && (strcmp(stack2[o].AtmNa
```



```
me,"O1") != 0) && (strcmp(stack2[o].AtmName,"O2") != 0)))
                                                        {
                                                                if (strncmp(stac
k1[n].AtmName,"H", 1) != 0 && strncmp(stack2[o].AtmName,"H", 1) != 0)
                                                                {
                                                                        it[count
er] = 3; /*possible electrostatic and hydropathic interaction*/
                                                                        ct0[coun
ter] = stack1[n].Res_Type;
                                                                        ct1[coun
ter] = stack2[o].Res_Type;
                                                                        at0[coun
ter] = stack1[n].Atom_Type;
                                                                        at1[coun
ter] = stack2[o].Atom_Type;

                                                                        //Assign
ing the ATOMIC SOLVATION PARAMETERS(Cummings, 1995) to the different atom types

                                                                        if ((str
ncmp(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2") == 0)
 || (strcmp(stack1[n].AtmName,"NZ") == 0))) //fully charged group
                                                                        {

stack1[n].ASP_Type = 4;
                                                                        }
                                                                        else if
((strncmp(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2")
!= 0) && (strcmp(stack1[n].AtmName,"NZ") != 0)))
                                                                        {

stack1[n].ASP_Type = 1;
                                                                        }
                                                                        else if
((strncmp(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2")
== 0) || (strcmp(stack1[n].AtmName,"OE2") == 0))) //fully charged group
                                                                        {

stack1[n].ASP_Type = 3;
                                                                        }
                                                                        else if
((strncmp(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2")
!= 0) && (strcmp(stack1[n].AtmName,"OE2") != 0)))
                                                                        {

stack1[n].ASP_Type = 1;
                                                                        }
                                                                        else if
(strncmp(stack1[n].AtmName,"C", 1) == 0)
                                                                        {

stack1[n].ASP_Type = 0;
                                                                        }
                                                                        else if
(strncmp(stack1[n].AtmName,"S", 1) == 0)
                                                                        {

stack1[n].ASP_Type = 2;
                                                                        }
                                                                        else
                                                                        {

stack1[n].ASP_Type = 5;
                                                                        }


                                                                        if ((str
ncmp(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2") == 0)
 || (strcmp(stack2[o].AtmName,"NZ") == 0))) //fully charged group
                                                                        {
```

```
stack2[o].ASP_Type = 4;
                                                                }
                                                                else if
((strncmp(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2")
!= 0) && (strcmp(stack2[o].AtmName,"NZ") != 0)))
                                                                {

stack2[o].ASP_Type = 1;
                                                                }
                                                                else if
((strncmp(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2")
== 0) || (strcmp(stack2[o].AtmName,"OE2") == 0))) //fully charged group
                                                                {

stack2[o].ASP_Type = 3;
                                                                }
                                                                else if
((strncmp(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2")
!= 0) && (strcmp(stack2[o].AtmName,"OE2") != 0)))
                                                                {

stack2[o].ASP_Type = 1;
                                                                }
                                                                else if
(strncmp(stack2[o].AtmName,"C", 1) == 0)
                                                                {

stack2[o].ASP_Type = 0;
                                                                }
                                                                else if
(strncmp(stack2[o].AtmName,"S", 1) == 0)
                                                                {

stack2[o].ASP_Type = 2;
                                                                }


                                                                else
                                                                {

stack2[o].ASP_Type = 5;
                                                                }
                                                                asp0[cou

nter] = stack1[n].ASP_Type;
                                                                asp1[cou

nter] = stack2[o].ASP_Type;
                                                                ro0[coun

ter] = stack1[n].r0;
                                                                ro1[coun

ter] = stack2[o].r0;
                                                                ri0[coun

ter] = stack1[n].NodeIndex;
                                                                ri1[coun

ter] = stack2[o].NodeIndex;
                                                                rd[count

er] = sqrt(dist.rsq);
                                                                        }
                                                                }
                                                                counter++;



                                                }
                                                else if (q0[counter] < 0.0 & q1[

counter] > 0.0)
                                                {
                                                        electrostatic++;
                                                        it[counter] = 1; /*possi

ble electrostatic interaction*/
                                                        ct0[counter] = stack1[n]
```



```
.Res_Type;
                                                        ct1[counter] = stack2[o]

.Res_Type;
                                                        ro0[counter] = stack1[n]

.r0;
                                                        ro1[counter] = stack2[o]

.r0;
                                                        ri0[counter] = stack1[n]

.NodeIndex;
                                                        ri1[counter] = stack2[o]

.NodeIndex;
                                                        rd[counter] = sqrt(dist.

rsq);

                                                        //ASSESSING the presence
 of HYDROGEN BONDS: CHECKING if the DONOR−ACCEPTOR DISTANCE IS <=3.2 Angstrom
                                                        if (((strncmp(stack1[n].

AtmName,"O",1) == 0) || (strncmp(stack1[n].AtmName,"N",1) == 0) || (strncmp(sta

ck1[n].AtmName,"S",1) == 0)) && (strncmp(stack2[o].AtmName,"H",1) == 0))
                                                        {
                                                                if ((strcmp(stac

k2[o].AtmName,"H2") == 0) || (strcmp(stack2[o].AtmName,"HH12") == 0) || (strcmp

(stack2[o].AtmName,"HH22") == 0) || (strcmp(stack2[o].AtmName,"HD22") == 0) ||

(strcmp(stack2[o].AtmName,"HE22") == 0) || (strcmp(stack2[o].AtmName,"HZ2") == 0

))
                                                                {
                                                                        if (strn

cmp(stack2[o−2].AtmName,"N",1) == 0)
                                                                        {

AtomIndex_1 = stack1[n].AtomIndex;

AtomIndex_2 = stack2[o−2].AtomIndex;

if ((strcmp(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB") != 0) && (strcmp(Atoms[Ato

mIndex_2−1][1].AtmHBlabel,"HB") != 0))

{

dDA = sqrt((stack2[o−2].x − stack1[n].x)*(stack2[o−2].x − stack1[n].x) + (stack2

[o−2].y −

stack1[n].y)*(stack2[o−2].y − stack1[n].y) + (stack2[o−2].z − stack1[n].z)*(stac

k2[o−2].z −

stack1[n].z));

if (dDA<=3.2)

{

        it[counter] = 4; /*possible electrostatic and hydrogen bond interaction*

/

        strcpy(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB");

        strcpy(Atoms[AtomIndex_2−1][1].AtmHBlabel,"HB");

}

}
                                                                        }
                                                                }
                                                                else if ((strcmp

(stack2[o].AtmName,"H3") == 0) || (strcmp(stack2[o].AtmName,"HZ3") == 0))
                                                                {
                                                                        if (strn

cmp(stack2[o−3].AtmName,"N",1) == 0)
                                                                        {
```

```
AtomIndex_1 = stack1[n].AtomIndex;

AtomIndex_2 = stack2[o−3].AtomIndex;

if ((strcmp(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB") != 0) && (strcmp(Atoms[Ato
mIndex_2−1][1].AtmHBlabel,"HB") != 0))

{

dDA = sqrt((stack2[o−3].x − stack1[n].x)*(stack2[o−3].x − stack1[n].x) + (stack2
[o−3].y −

stack1[n].y)*(stack2[o−3].y − stack1[n].y) + (stack2[o−3].z − stack1[n].z)*(stac
k2[o−3].z −

stack1[n].z));

if (dDA<=3.2)

{

        it[counter] = 4; /*possible electrostatic and hydrogen bond interaction*
/

        strcpy(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB");

        strcpy(Atoms[AtomIndex_2−1][1].AtmHBlabel,"HB");

}

}
                                                                    }
                                                                }
                                                                else
                                                                {
                                                                    if ((str
ncmp(stack2[o−1].AtmName,"N",1) == 0) || (strncmp(stack2[o−1].AtmName,"O",1) ==
 0))
                                                                    {

AtomIndex_1 = stack1[n].AtomIndex;

AtomIndex_2 = stack2[o−1].AtomIndex;

if ((strcmp(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB") != 0) && (strcmp(Atoms[Ato
mIndex_2−1][1].AtmHBlabel,"HB") != 0))

{

dDA = sqrt((stack2[o−1].x − stack1[n].x)*(stack2[o−1].x − stack1[n].x) + (stack2
[o−1].y −

stack1[n].y)*(stack2[o−1].y − stack1[n].y) + (stack2[o−1].z − stack1[n].z)*(stac
k2[o−1].z −

stack1[n].z));

if (dDA<=3.2)

{

        it[counter] = 4; /*possible electrostatic and hydrogen bond interaction*
/

        strcpy(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB");

        strcpy(Atoms[AtomIndex_2−1][1].AtmHBlabel,"HB");
```



```
}

}
                                                                    }
                                                                }
                                                            }

                                                            //HYDROPHOBIC interactio
ns between non−hydrogen side−chain atoms

                                                            if (((strcmp(stack1[n].A
tmName,"N") != 0) && (strcmp(stack1[n].AtmName,"H") != 0) && (strcmp(stack1[n].
AtmName,"H1") != 0) &&
                                                            (strcmp(stack1[n].AtmNam
e,"H2") != 0) && (strcmp(stack1[n].AtmName,"H3") != 0) && (strcmp(stack1[n].Atm
Name,"CA") != 0) &&
                                                            (strcmp(stack1[n].AtmNam
e,"C") != 0) && (strcmp(stack1[n].AtmName,"O") != 0) && (strcmp(stack1[n].AtmNa
me,"O1") != 0) &&
                                                            (strcmp(stack1[n].AtmNam
e,"O2") != 0)) && ((strcmp(stack2[o].AtmName,"N") != 0) && (strcmp(stack2[o].At
mName,"H") != 0) && (strcmp(stack2[o].AtmName,"H1") != 0) &&
                                                            (strcmp(stack2[o].AtmNam
e,"H2") != 0) && (strcmp(stack2[o].AtmName,"H3") != 0) && (strcmp(stack2[o].Atm
Name,"CA") != 0) &&
                                                            (strcmp(stack2[o].AtmNam
e,"C") != 0) && (strcmp(stack2[o].AtmName,"O") != 0) && (strcmp(stack2[o].AtmNa
me,"O1") != 0) && (strcmp(stack2[o].AtmName,"O2") != 0)))
                                                            {
                                                                if (strncmp(stac
k1[n].AtmName,"H", 1) != 0 && strncmp(stack2[o].AtmName,"H", 1) != 0)
                                                                {
                                                                    it[count
er] = 3; /*possible electrostatic and hydropathic interaction*/
                                                                    ct0[coun
ter] = stack1[n].Res_Type;
                                                                    ct1[coun
ter] = stack2[o].Res_Type;
                                                                    at0[coun
ter] = stack1[n].Atom_Type;
                                                                    at1[coun
ter] = stack2[o].Atom_Type;

                                                                    //Assign
ing the ATOMIC SOLVATION PARAMETERS(Cummings, 1995) to the different atom types

                                                                    if ((str
ncmp(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2") == 0)
 || (strcmp(stack1[n].AtmName,"NZ") == 0))) //fully charged group
                                                                    {

stack1[n].ASP_Type = 4;
                                                                    }
                                                                    else if
((strncmp(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2")
!= 0) && (strcmp(stack1[n].AtmName,"NZ") != 0)))
                                                                    {

stack1[n].ASP_Type = 1;
                                                                    }
                                                                    else if
((strncmp(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2")
== 0) || (strcmp(stack1[n].AtmName,"OE2") == 0))) //fully charged group
                                                                    {

stack1[n].ASP_Type = 3;
                                                                    }
                                                                    else if
```

```
((strncmp(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2")
!= 0) && (strcmp(stack1[n].AtmName,"OE2") != 0)))
                                                                {

stack1[n].ASP_Type = 1;

                                                                }
                                                                else if
(strncmp(stack1[n].AtmName,"C", 1) == 0)
                                                                {

stack1[n].ASP_Type = 0;

                                                                }
                                                                else if
(strncmp(stack1[n].AtmName,"S", 1) == 0)
                                                                {

stack1[n].ASP_Type = 2;

                                                                }
                                                                else
                                                                {


stack1[n].ASP_Type = 5;

                                                                }


                                                                if ((str
ncmp(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2") == 0)
 || (strcmp(stack2[o].AtmName,"NZ") == 0))) //fully charged group
                                                                {

stack2[o].ASP_Type = 4;

                                                                }
                                                                else if
((strncmp(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2")
!= 0) && (strcmp(stack2[o].AtmName,"NZ") != 0)))
                                                                {

stack2[o].ASP_Type = 1;

                                                                }
                                                                else if
((strncmp(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2")
== 0) || (strcmp(stack2[o].AtmName,"OE2") == 0))) //fully charged group
                                                                {

stack2[o].ASP_Type = 3;

                                                                }
                                                                else if
((strncmp(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2")
!= 0) && (strcmp(stack2[o].AtmName,"OE2") != 0)))
                                                                {

stack2[o].ASP_Type = 1;

                                                                }
                                                                else if
(strncmp(stack2[o].AtmName,"C", 1) == 0)
                                                                {

stack2[o].ASP_Type = 0;

                                                                }
                                                                else if
(strncmp(stack2[o].AtmName,"S", 1) == 0)
                                                                {

stack2[o].ASP_Type = 2;

                                                                }


                                                                else
                                                                {


stack2[o].ASP_Type = 5;
```



```
                                                                }
                                                                asp0[cou
nter] = stack1[n].ASP_Type;
                                                                asp1[cou
nter] = stack2[o].ASP_Type;
                                                                ro0[coun
ter] = stack1[n].r0;
                                                                ro1[coun
ter] = stack2[o].r0;
                                                                ri0[coun
ter] = stack1[n].NodeIndex;
                                                                ri1[coun
ter] = stack2[o].NodeIndex;
                                                                rd[count
er] = sqrt(dist.rsq);
                                                                        }
                                                                }
                                                                counter++;



                                                        }
                                                        else if (q0[counter] > 0.0 & q1[
counter] < 0.0)
                                                        {
                                                                electrostatic++;
                                                                it[counter] = 1; /*possi
ble electrostatic interaction*/
                                                                ct0[counter] = stack1[n]
.Res_Type;
                                                                ct1[counter] = stack2[o]
.Res_Type;
                                                                at0[counter] = stack1[n]
.Atom_Type;
                                                                at1[counter] = stack2[o]
.Atom_Type;
                                                                ro0[counter] = stack1[n]
.r0;
                                                                ro1[counter] = stack2[o]
.r0;
                                                                ri0[counter] = stack1[n]
.NodeIndex;
                                                                ri1[counter] = stack2[o]
.NodeIndex;
                                                                rd[counter] = sqrt(dist.
rsq);

                                                                //ASSESSING the presence
 of HYDROGEN BONDS: CHECKING if the DONOR−ACCEPTOR DISTANCE IS <=3.2 Angstrom

                                                                if ((strncmp(stack1[n].A
tmName,"H",1) == 0) && ((strncmp(stack2[o].AtmName,"O",1) == 0) || (strncmp(sta
ck2[o].AtmName,"N",1) == 0) || (strncmp(stack2[o].AtmName,"S",1) == 0)))
                                                                {
                                                                        if ((strcmp(stac
k1[n].AtmName,"H2") == 0) || (strcmp(stack1[n].AtmName,"HH12") == 0) || (strcmp
(stack1[n].AtmName,"HH22") == 0) || (strcmp(stack1[n].AtmName,"HD22") == 0) ||
(strcmp(stack1[n].AtmName,"HE22") == 0) || (strcmp(stack1[n].AtmName,"HZ2") == 0
))
                                                                        {
                                                                                if (strn
cmp(stack1[n−2].AtmName,"N",1) == 0)
                                                                                {

AtomIndex_1 = stack1[n−2].AtomIndex;

AtomIndex_2 = stack2[o].AtomIndex;

if ((strcmp(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB") != 0) && (strcmp(Atoms[Ato
mIndex_2−1][1].AtmHBlabel,"HB") != 0))
```

```
{

dDA = sqrt((stack2[o].x − stack1[n−2].x)*(stack2[o].x − stack1[n−2].x) + (stack2
[o].y −

stack1[n−2].y)*(stack2[o].y − stack1[n−2].y) + (stack2[o].z − stack1[n−2].z)*(st
ack2[o].z −

stack1[n−2].z));

if (dDA<=3.2)

{

        it[counter] = 4; /*possible electrostatic and hydrogen bond interaction*
/

        strcpy(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB");

        strcpy(Atoms[AtomIndex_2−1][1].AtmHBlabel,"HB");

}

}
                                                                        }
                                                                }
                                                                else if ((strcmp
(stack1[n].AtmName,"H3") == 0) || (strcmp(stack1[n].AtmName,"HZ3") == 0))
                                                                {
                                                                        if (strn
cmp(stack1[n−3].AtmName,"N",1) == 0)
                                                                        {

AtomIndex_1 = stack1[n−3].AtomIndex;

AtomIndex_2 = stack2[o].AtomIndex;

if ((strcmp(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB") != 0) && (strcmp(Atoms[Ato
mIndex_2−1][1].AtmHBlabel,"HB") != 0))

{

dDA = sqrt((stack2[o].x − stack1[n−3].x)*(stack2[o].x − stack1[n−3].x) + (stack2
[o].y −

stack1[n−3].y)*(stack2[o].y − stack1[n−3].y) + (stack2[o].z − stack1[n−3].z)*(st
ack2[o].z −

stack1[n−3].z));

if (dDA<=3.2)

{

        it[counter] = 4; /*possible electrostatic and hydrogen bond interaction*
/

        strcpy(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB");

        strcpy(Atoms[AtomIndex_2−1][1].AtmHBlabel,"HB");

}

}
                                                                        }
                                                                }
                                                                else
                                                                {
```



```
                                                                        if ((str
ncmp(stack1[n−1].AtmName,"N",1) == 0) || (strncmp(stack1[n−1].AtmName,"O",1) ==
 0))
                                                                        {

AtomIndex_1 = stack1[n−1].AtomIndex;

AtomIndex_2 = stack2[o].AtomIndex;

if ((strcmp(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB") != 0) && (strcmp(Atoms[Ato
mIndex_2−1][1].AtmHBlabel,"HB") != 0))

{

dDA = sqrt((stack2[o].x − stack1[n−1].x)*(stack2[o].x − stack1[n−1].x) + (stack2
[o].y −

stack1[n−1].y)*(stack2[o].y − stack1[n−1].y) + (stack2[o].z − stack1[n−1].z)*(st
ack2[o].z −

stack1[n−1].z));

if (dDA<=3.2)

{

        it[counter] = 4; /*possible electrostatic and hydrogen bond interaction*
/

        strcpy(Atoms[AtomIndex_1−1][0].AtmHBlabel,"HB");

        strcpy(Atoms[AtomIndex_2−1][1].AtmHBlabel,"HB");

}

}
                                                                        }
                                                                }
                                                        }

                                                        //HYDROPHOBIC interactio
ns between non−hydrogen side−chain atoms

                                                        if (((strcmp(stack1[n].A
tmName,"N") != 0) && (strcmp(stack1[n].AtmName,"H") != 0) && (strcmp(stack1[n].
AtmName,"H1") != 0) &&
                                                        (strcmp(stack1[n].AtmNam
e,"H2") != 0) && (strcmp(stack1[n].AtmName,"H3") != 0) && (strcmp(stack1[n].Atm
Name,"CA") != 0) &&
                                                        (strcmp(stack1[n].AtmNam
e,"C") != 0) && (strcmp(stack1[n].AtmName,"O") != 0) && (strcmp(stack1[n].AtmNa
me,"O1") != 0) &&
                                                        (strcmp(stack1[n].AtmNam
e,"O2") != 0)) && ((strcmp(stack2[o].AtmName,"N") != 0) && (strcmp(stack2[o].At
mName,"H") != 0) && (strcmp(stack2[o].AtmName,"H1") != 0) &&
                                                        (strcmp(stack2[o].AtmNam
e,"H2") != 0) && (strcmp(stack2[o].AtmName,"H3") != 0) && (strcmp(stack2[o].Atm
Name,"CA") != 0) &&
                                                        (strcmp(stack2[o].AtmNam
e,"C") != 0) && (strcmp(stack2[o].AtmName,"O") != 0) && (strcmp(stack2[o].AtmNa
me,"O1") != 0) && (strcmp(stack2[o].AtmName,"O2") != 0)))
                                                        {
                                                                if (strncmp(stac
k1[n].AtmName,"H", 1) != 0 && strncmp(stack2[o].AtmName,"H", 1) != 0)
                                                                {
                                                                        it[count
er] = 3; /*possible electrostatic and hydropathic interaction*/
                                                                        ct0[coun
ter] = stack1[n].Res_Type;
```

```
                                                                        ct1[coun
ter] = stack2[o].Res_Type;
                                                                        at0[coun
ter] = stack1[n].Atom_Type;
                                                                        at1[coun
ter] = stack2[o].Atom_Type;

                                                                        //Assign
ing the ATOMIC SOLVATION PARAMETERS(Cummings, 1995) to the different atom types

                                                                        if ((str
ncmp(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2") == 0)
 || (strcmp(stack1[n].AtmName,"NZ") == 0))) //fully charged group
                                                                        {

stack1[n].ASP_Type = 4;
                                                                        }
                                                                        else if
((strncmp(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2")
!= 0) && (strcmp(stack1[n].AtmName,"NZ") != 0)))
                                                                        {

stack1[n].ASP_Type = 1;
                                                                        }
                                                                        else if
((strncmp(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2")
== 0) || (strcmp(stack1[n].AtmName,"OE2") == 0))) //fully charged group
                                                                        {

stack1[n].ASP_Type = 3;
                                                                        }
                                                                        else if
((strncmp(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2")
!= 0) && (strcmp(stack1[n].AtmName,"OE2") != 0)))
                                                                        {

stack1[n].ASP_Type = 1;
                                                                        }
                                                                        else if
(strncmp(stack1[n].AtmName,"C", 1) == 0)
                                                                        {

stack1[n].ASP_Type = 0;
                                                                        }
                                                                        else if
(strncmp(stack1[n].AtmName,"S", 1) == 0)
                                                                        {

stack1[n].ASP_Type = 2;
                                                                        }
                                                                        else
                                                                        {


stack1[n].ASP_Type = 5;
                                                                        }

                                                                        if ((str
ncmp(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2") == 0)
 || (strcmp(stack2[o].AtmName,"NZ") == 0))) //fully charged group
                                                                        {

stack2[o].ASP_Type = 4;
                                                                        }
                                                                        else if
((strncmp(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2")
!= 0) && (strcmp(stack2[o].AtmName,"NZ") != 0)))
                                                                        {

stack2[o].ASP_Type = 1;
```



```
                                                                        }
                                                                        else if
((strncmp(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2")
== 0) || (strcmp(stack2[o].AtmName,"OE2") == 0))) //fully charged group
                                                                        {

stack2[o].ASP_Type = 3;
                                                                        }
                                                                        else if
((strncmp(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2")
!= 0) && (strcmp(stack2[o].AtmName,"OE2") != 0)))
                                                                        {

stack2[o].ASP_Type = 1;
                                                                        }
                                                                        else if
(strncmp(stack2[o].AtmName,"C", 1) == 0)
                                                                        {

stack2[o].ASP_Type = 0;
                                                                        }
                                                                        else if
(strncmp(stack2[o].AtmName,"S", 1) == 0)
                                                                        {

stack2[o].ASP_Type = 2;
                                                                        }


                                                                        else
                                                                        {

stack2[o].ASP_Type = 5;
                                                                        }
                                                                        asp0[cou
nter] = stack1[n].ASP_Type;
                                                                        asp1[cou
nter] = stack2[o].ASP_Type;
                                                                        ro0[coun
ter] = stack1[n].r0;
                                                                        ro1[coun
ter] = stack2[o].r0;
                                                                        ri0[coun
ter] = stack1[n].NodeIndex;
                                                                        ri1[coun
ter] = stack2[o].NodeIndex;
                                                                        rd[count
er] = sqrt(dist.rsq);
                                                                                }
                                                                        }
                                                                        counter++;


                                                                }

                                                                //HYDROPHOBIC interactions betwe
en non−hydrogen side−chain atoms



                                                                else if (((strcmp(stack1[n].AtmN
ame,"N") != 0) && (strcmp(stack1[n].AtmName,"H") != 0) && (strcmp(stack1[n].Atm
Name,"H1") != 0) &&
                                                                (strcmp(stack1[n].AtmName,"H2")
 != 0) && (strcmp(stack1[n].AtmName,"H3") != 0) && (strcmp(stack1[n].AtmName,"C
A") != 0) &&
                                                                (strcmp(stack1[n].AtmName,"C")
!= 0) && (strcmp(stack1[n].AtmName,"O") != 0) && (strcmp(stack1[n].AtmName,"O1"
) != 0) &&
                                                                (strcmp(stack1[n].AtmName,"O2")
```

```
 != 0)) && ((strcmp(stack2[o].AtmName,"N") != 0) && (strcmp(stack2[o].AtmName,"
H") != 0) && (strcmp(stack2[o].AtmName,"H1") != 0) &&
                                                (strcmp(stack2[o].AtmName,"H2")
 != 0) && (strcmp(stack2[o].AtmName,"H3") != 0) && (strcmp(stack2[o].AtmName,"C
A") != 0) &&
                                                (strcmp(stack2[o].AtmName,"C")
!= 0) && (strcmp(stack2[o].AtmName,"O") != 0) && (strcmp(stack2[o].AtmName,"O1"
) != 0) && (strcmp(stack2[o].AtmName,"O2") != 0)))
                                                {
                                                        if (strncmp(stack1[n].At
mName,"H", 1) != 0 && strncmp(stack2[o].AtmName,"H", 1) != 0)
                                                        {
                                                                it[counter] = 2;
 /*possible hydropathic interaction*/
                                                                ct0[counter] = s
tack1[n].Res_Type;
                                                                ct1[counter] = s
tack2[o].Res_Type;
                                                                at0[counter] = s
tack1[n].Atom_Type;
                                                                at1[counter] = s
tack2[o].Atom_Type;

                                                                //Assigning the
ATOMIC SOLVATION PARAMETERS(Cummings, 1995) to the different atom types

                                                                if ((strncmp(sta
ck1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2") == 0) || (str
cmp(stack1[n].AtmName,"NZ") == 0))) //fully charged group
                                                                {
                                                                        stack1[n
].ASP_Type = 4;
                                                                }
                                                                else if ((strncm
p(stack1[n].AtmName,"N", 1) == 0) && ((strcmp(stack1[n].AtmName,"NH2") != 0) &&
 (strcmp(stack1[n].AtmName,"NZ") != 0)))
                                                                {
                                                                        stack1[n
].ASP_Type = 1;
                                                                }
                                                                else if ((strncm
p(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2") == 0) ||
 (strcmp(stack1[n].AtmName,"OE2") == 0))) //fully charged group
                                                                {
                                                                        stack1[n
].ASP_Type = 3;
                                                                }
                                                                else if ((strncm
p(stack1[n].AtmName,"O", 1) == 0) && ((strcmp(stack1[n].AtmName,"OD2") != 0) &&
 (strcmp(stack1[n].AtmName,"OE2") != 0)))
                                                                {
                                                                        stack1[n
].ASP_Type = 1;
                                                                }
                                                                else if (strncmp
(stack1[n].AtmName,"C", 1) == 0)
                                                                {
                                                                        stack1[n
].ASP_Type = 0;
                                                                }
                                                                else if (strncmp
(stack1[n].AtmName,"S", 1) == 0)
                                                                {
                                                                        stack1[n
].ASP_Type = 2;
                                                                }
                                                                else
                                                                {
                                                                        stack1[n
```



```
].ASP_Type = 5;
                                                                }

                                                                if ((strncmp(sta
ck2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2") == 0) || (str
cmp(stack2[o].AtmName,"NZ") == 0))) //fully charged group
                                                                {
                                                                        stack2[o
].ASP_Type = 4;
                                                                }
                                                                else if ((strncm
p(stack2[o].AtmName,"N", 1) == 0) && ((strcmp(stack2[o].AtmName,"NH2") != 0) &&
 (strcmp(stack2[o].AtmName,"NZ") != 0)))
                                                                {
                                                                        stack2[o
].ASP_Type = 1;
                                                                }
                                                                else if ((strncm
p(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2") == 0) ||
 (strcmp(stack2[o].AtmName,"OE2") == 0))) //fully charged group
                                                                {
                                                                        stack2[o
].ASP_Type = 3;
                                                                }
                                                                else if ((strncm
p(stack2[o].AtmName,"O", 1) == 0) && ((strcmp(stack2[o].AtmName,"OD2") != 0) &&
 (strcmp(stack2[o].AtmName,"OE2") != 0)))
                                                                {
                                                                        stack2[o
].ASP_Type = 1;
                                                                }
                                                                else if (strncmp
(stack2[o].AtmName,"C", 1) == 0)
                                                                {
                                                                        stack2[o
].ASP_Type = 0;
                                                                }
                                                                else if (strncmp
(stack2[o].AtmName,"S", 1) == 0)
                                                                {
                                                                        stack2[o
].ASP_Type = 2;
                                                                }

                                                                else
                                                                {
                                                                        stack2[o
].ASP_Type = 5;
                                                                }
                                                                asp0[counter] =
stack1[n].ASP_Type;
                                                                asp1[counter] =
stack2[o].ASP_Type;
                                                                ro0[counter] = s
tack1[n].r0;
                                                                ro1[counter] = s
tack2[o].r0;
                                                                ri0[counter] = s
tack1[n].NodeIndex;
                                                                ri1[counter] = s
tack2[o].NodeIndex;
                                                                rd[counter] = sq
rt(dist.rsq);
                                                                counter++;
                                                        }
                                                }
                                        }
```

```
                        }
                }
        }

        return counter;
        //fclose(test);
}


int CONTACTS(void)
{
        int n,o,counter;

        counter=0;
        for (n=0;n<nAtom[0];n++)
        {
                for (o=0;o<nAtom[1];o++)
                {

                        dist.r01 = HARDCORE_R*(Atoms[n][0].r0+Atoms[o][1].r0);
                        dist.r02 = CONTACT_R*(Atoms[n][0].r0+Atoms[o][1].r0);
                        dist.rsq = (Atoms[o][1].x−Atoms[n][0].x)*(Atoms[o][1].x−
Atoms[n][0].x)+(Atoms[o][1].y−Atoms[n][0].y)*(Atoms[o][1].y−Atoms[n][0].y)+(Atom
s[o][1].z−Atoms[n][0].z)*(Atoms[o][1].z−Atoms[n][0].z);

                        if ((dist.rsq>=dist.r01*dist.r01)&&(dist.rsq<=dist.r02*d
ist.r02))
                        {
                                counter++;
                        }

                }

        }
        return counter;
}


int CLASHES(void)
{
        int n,o,counter;

        counter=0;
        for (n=0;n<nAtom[0];n++)
        {
                for (o=0;o<nAtom[1];o++)
                {

                        dist.r01 = HARDCORE_R*(Atoms[n][0].r0+Atoms[o][1].r0);
                        dist.r02 = CONTACT_R*(Atoms[n][0].r0+Atoms[o][1].r0);
                        dist.rsq = (Atoms[o][1].x−Atoms[n][0].x)*(Atoms[o][1].x−
Atoms[n][0].x)+(Atoms[o][1].y−Atoms[n][0].y)*(Atoms[o][1].y−Atoms[n][0].y)+(Atom
s[o][1].z−Atoms[n][0].z)*(Atoms[o][1].z−Atoms[n][0].z);

                        if (dist.rsq<(dist.r01*dist.r01))
                        {
                                counter++;
                        }


                }

        }
        return counter;
}

void READCHARGES(void)
{
```



```
        FILE *input;

        int i = 0;

        char s[256], szLine[MAXLENLINE], szAtomName[8], szResName[8], szAtomType
[8], buff[256],*ReadStatus;

        float AtomCharge;

        sprintf(s, "charges.dat"); //file containing the GROMOS54A7 atomic point ch
arges

        input = fopen(s,"r");

        if(input == NULL)
        {
                printf("Fail to open charges file: %s\nQuit.\n",s);
                exit(0);
        }

        while (!feof(input))
        {
                fscanf(input, "%s %s %s %f\n", &szResName, &szAtomName, &szAtom
Type, &AtomCharge);
                if(strncmp(szResName, "CYS2", 4)==0)
                {
                        strcpy(szResName, "CYS");
                }
                Charges[i].Res_Type = Get_AA_Type(szResName);
                strcpy(Charges[i].AtmName, szAtomName);
                strcpy(Charges[i].Atm_Type, szAtomType);
                Charges[i].Atm_Charge = AtomCharge;
                i++;
        }
        fclose(input);
}


void CENTRALIZER(double a)
{
        int i,j,k;

        //protein centralizer
        for (i=0;i<2;i++)
        {
                for (j=0;j<3;j++)
                {
                        MEAN_R[j][i]=0;
                }
        }


        for (i=0;i<2;i++)
        {
                for (j=0;j<nAtom[i];j++)
                {
                        MEAN_R[0][i]=MEAN_R[0][i]+Atoms[j][i].x;
                        MEAN_R[1][i]=MEAN_R[1][i]+Atoms[j][i].y;
                        MEAN_R[2][i]=MEAN_R[2][i]+Atoms[j][i].z;
                }
                for (j=0;j<3;j++)
                {
                        MEAN_R[j][i]=MEAN_R[j][i]/nAtom[i];
                }
        }

        //ROUNDING ERRORS DUE TO PDB FILE FORMAT
        //for (i=0;i<2;i++)
        //{
```

```
        //      printf("%d      %lf     %lf     %lf\n",i,MEAN_R[0][i],MEAN_R[1][
i],MEAN_R[2][i]);
        //}
        //printf("\n\n");


        for (i=0;i<2;i++)
        {
                for (j=0;j<nAtom[i];j++)
                {
                        Atoms[j][i].x = Atoms[j][i].x − MEAN_R[0][i];
                        Atoms[j][i].y = Atoms[j][i].y − MEAN_R[1][i];
                        Atoms[j][i].z = Atoms[j][i].z − MEAN_R[2][i];
                }
        }
        for (i=0;i<2;i++)
        {
                for (j=0;j<3;j++)
                {
                        MEAN_R[j][i]=0;
                }
        }
        for (i=0;i<2;i++)
        {
                for (j=0;j<nAtom[i];j++)
                {
                        MEAN_R[0][i]=MEAN_R[0][i]+Atoms[j][i].x;
                        MEAN_R[1][i]=MEAN_R[1][i]+Atoms[j][i].y;
                        MEAN_R[2][i]=MEAN_R[2][i]+Atoms[j][i].z;
                }
                for (j=0;j<3;j++)
                {
                        MEAN_R[j][i]=MEAN_R[j][i]/nAtom[i];
                }
        }

        //CONTROL OUTPUT: CENTRALIZED PROTEINS with CMs at (0,0,0):
        //for (i=0;i<2;i++)
        //{
        //      printf("%d      %lf     %lf     %lf\n",i,MEAN_R[0][i],MEAN_R[1][
i],MEAN_R[2][i]);
        //}


        //MOVE STRUCTURE 2 along X/Y/Z−axis for a=Rg1+Rg2 in DIR−direction //
        ////////////////////////////////////////////////////////////////
        if (DIR==1)
        {
                for (j=0;j<nAtom[1];j++)
                {
                        Atoms[j][1].x = Atoms[j][1].x + a;
                }
                INITIALORIENTATOR();
        }
        else
        {
                if (DIR==−1)
                {
                        for (j=0;j<nAtom[1];j++)
                        {
                                Atoms[j][1].x = Atoms[j][1].x − a;
                        }
                        INITIALORIENTATOR();
                }
                else
                {
                        if (DIR == 2)
                        {
                                for (j=0;j<nAtom[1];j++)
```



```
                                {
                                        Atoms[j][1].y = Atoms[j][1].y + a;
                                }
                                INITIALORIENTATOR();
                        }
                        else
                        {
                                if (DIR == −2)
                                {
                                        for (j=0;j<nAtom[1];j++)
                                        {
                                                Atoms[j][1].y = Atoms[j][1].y −
a;
                                        }
                                        INITIALORIENTATOR();
                                }
                                else
                                {
                                        if (DIR == 3)
                                        {
                                                for (j=0;j<nAtom[1];j++)
                                                {
                                                        Atoms[j][1].z = Atoms[j]
[1].z + a;
                                                }
                                                INITIALORIENTATOR();
                                        }
                                        else
                                        {
                                                if (DIR == −3)
                                                {
                                                        for (j=0;j<nAtom[1];j++)
                                                        {
                                                                Atoms[j][1].z =
Atoms[j][1].z − a;
                                                        }
                                                        INITIALORIENTATOR();
                                                }
                                        }

                                }
                        }


                }
        }


        //CONTROL OUTPUT: MOVED PROTEINS − CMs
        //PROTEIN 1: (0,0,0)
        //PROTEIN 2: (restdist,0,0)

        //for (i=0;i<2;i++)
        //{
        //      printf("%d      %lf     %lf     %lf\n",i,MEAN_R[0][i],MEAN_R[1][
i],MEAN_R[2][i]);
        //}

}


void INITIALORIENTATOR(void)
{
        int i,j,k;
        double oldmoved_x[MAXATOMNUM];
        double oldmoved_y[MAXATOMNUM];
        double oldmoved_z[MAXATOMNUM];

        //recalculating the coordinates of the structure stored in Atoms in rela
```

```
tion to its centre of mass

        for (i=0;i<2;i++)
        {
                for (j=0;j<3;j++)
                {
                        MEAN_R[j][i]=0;
                }
        }

        for (i=0;i<2;i++)
        {
                for (j=0;j<nAtom[i];j++)
                {
                        MEAN_R[0][i]=MEAN_R[0][i]+Atoms[j][i].x;
                        MEAN_R[1][i]=MEAN_R[1][i]+Atoms[j][i].y;
                        MEAN_R[2][i]=MEAN_R[2][i]+Atoms[j][i].z;
                }
                for (j=0;j<3;j++)
                {
                        MEAN_R[j][i]=MEAN_R[j][i]/nAtom[i];
                }
        }

        for (i=0;i<nAtom[1];i++)
        {
                RotMoved[i].x = Atoms[i][1].x − MEAN_R[0][1];
                RotMoved[i].y = Atoms[i][1].y − MEAN_R[1][1];
                RotMoved[i].z = Atoms[i][1].z − MEAN_R[2][1];
        }

        //performing −/+90º rotation over each axis on structure stored in RotMo
ved via matrix multiplication


        if (ORIENTATION == 1)
        {
                for (i=0;i<nAtom[1];i++)
                {
                        oldmoved_x[i] = RotMoved[i].x;
                        oldmoved_y[i] = RotMoved[i].y;
                        oldmoved_z[i] = RotMoved[i].z;
                        RotMoved[i].x = oldmoved_x[i];
                        RotMoved[i].y = −oldmoved_z[i];
                        RotMoved[i].z = oldmoved_y[i];
                }
        }
        else if (ORIENTATION == −1)
        {
                for (i=0;i<nAtom[1];i++)
                {
                        oldmoved_x[i] = RotMoved[i].x;
                        oldmoved_y[i] = RotMoved[i].y;
                        oldmoved_z[i] = RotMoved[i].z;
                        RotMoved[i].x = oldmoved_x[i];
                        RotMoved[i].y = oldmoved_z[i];
                        RotMoved[i].z = −oldmoved_y[i];
                }
        }
        else if (ORIENTATION == 2)
        {
                for (i=0;i<nAtom[1];i++)
                {
                        oldmoved_x[i] = RotMoved[i].x;
                        oldmoved_y[i] = RotMoved[i].y;
                        oldmoved_z[i] = RotMoved[i].z;
                        RotMoved[i].x = oldmoved_z[i];
                        RotMoved[i].y = oldmoved_y[i];
                        RotMoved[i].z = −oldmoved_x[i];
```



```
                }
        }
        else if (ORIENTATION == −2)
        {
                for (i=0;i<nAtom[1];i++)
                {
                        oldmoved_x[i] = RotMoved[i].x;
                        oldmoved_y[i] = RotMoved[i].y;
                        oldmoved_z[i] = RotMoved[i].z;
                        RotMoved[i].x = −oldmoved_z[i];
                        RotMoved[i].y = oldmoved_y[i];
                        RotMoved[i].z = oldmoved_x[i];
                }
        }
        else if (ORIENTATION == 3)
        {
                for (i=0;i<nAtom[1];i++)
                {
                        oldmoved_x[i] = RotMoved[i].x;
                        oldmoved_y[i] = RotMoved[i].y;
                        oldmoved_z[i] = RotMoved[i].z;
                        RotMoved[i].x = oldmoved_y[i];
                        RotMoved[i].y = −oldmoved_x[i];
                        RotMoved[i].z = oldmoved_z[i];
                }
        }
        else if (ORIENTATION == −3)
        {
                for (i=0;i<nAtom[1];i++)
                {
                        oldmoved_x[i] = RotMoved[i].x;
                        oldmoved_y[i] = RotMoved[i].y;
                        oldmoved_z[i] = RotMoved[i].z;
                        RotMoved[i].x = −oldmoved_y[i];
                        RotMoved[i].y = oldmoved_x[i];
                        RotMoved[i].z = oldmoved_z[i];
                }
        }
        else
        {
                if (ORIENTATION == 4)
                {
                        for (i=0;i<nAtom[1];i++)
                        {
                                oldmoved_x[i] = RotMoved[i].x;
                                oldmoved_y[i] = RotMoved[i].y;
                                oldmoved_z[i] = RotMoved[i].z;
                                RotMoved[i].x = oldmoved_x[i];
                                RotMoved[i].y = oldmoved_y[i];
                                RotMoved[i].z = oldmoved_z[i];
                        }
                }
        }


        //recalculating the coordinates of the structure stored in RotMoved in r
elation to the origin of the cartesian axes

        for (i=0;i<nAtom[1];i++)
        {
                Atoms[i][1].x = RotMoved[i].x + MEAN_R[0][1];
                Atoms[i][1].y = RotMoved[i].y + MEAN_R[1][1];
                Atoms[i][1].z = RotMoved[i].z + MEAN_R[2][1];
        }

        ////////////////////////////////////////////////////////////
        for (i=0;i<2;i++)
        {
                for (j=0;j<3;j++)
```

```
                {
                        MEAN_R[j][i]=0;
                }
        }
        for (i=0;i<2;i++)
        {
                for (j=0;j<nAtom[i];j++)
                {
                        MEAN_R[0][i]=MEAN_R[0][i]+Atoms[j][i].x;
                        MEAN_R[1][i]=MEAN_R[1][i]+Atoms[j][i].y;
                        MEAN_R[2][i]=MEAN_R[2][i]+Atoms[j][i].z;
                }
                for (j=0;j<3;j++)
                {
                        MEAN_R[j][i]=MEAN_R[j][i]/nAtom[i];
                }
        }
}


void READER(int a, int b)
{
        FILE *input;

        int i,j,k;
        int ReadCoord, HIS_Type;

        int cter;

        char s[256], szLine[MAXLENLINE], szAtomName[8], szResName[8], buff[256],
*ReadStatus;

        for (i=0;i<2;i++)
        {
                input=NULL;
                Conf[i].contEn=0;
                Conf[i].Rg=0;
                Conf[i].RMSD=0;
                ResNum[i]=0;
                nAtom[i]=0;

                if (i==0)
                {
                        j=a;
                }
                else
                {
                        j=b;
                }

                if(j < 10)
                {
                        sprintf(s, "D76N−I1−pH7p2−chain−newcf−000%d.pdb",j);
                }
                else
                        if(j < 100)
                        {
                                sprintf(s, "D76N−I1−pH7p2−chain−newcf−00%d.pdb",j);
                        }
                        else
                                if(j < 1000)
                                {
                                        sprintf(s, "D76N−I1−pH7p2−chain−newcf−0%d.pd
b",j);
                                }
                                else
                                {
                                        sprintf(s, "D76N−I1−pH7p2−chain−newcf−%d.pdb
",j);
```



```
                                }


                //printf("Filename: [%s]\n", s); fflush(stdout);

                ////////////////////////////////////////////////////////////////
////////////////


                input = fopen(s,"r");


                if(input == NULL)
                {
                        printf("Fail to open PDB file: %s\nQuit.\n",s);
                        exit(0);
                }

                k=0;
                while(1)
                {

                        ReadStatus = fgets(szLine, MAXLENLINE, input);

                        if(ReadStatus == NULL)
                        {
                                break;
                        }

                        if (k==1)
                        {
                                ReadCoord=sscanf(szLine+10, "%d", &(Conf[i].con
tEn));
                                ReadCoord=sscanf(szLine+24, "%lf", &(Conf[i].Rg)
);
                                ReadCoord=sscanf(szLine+43, "%lf", &(Conf[i].RMS
D));
                        }

                        if (strncmp(szLine, "ATOM", 4)==0)
                        {
                                sscanf(szLine+5, "%d", &(Atoms[nAtom[i]][i].Ato
mIndex));
                                sscanf(szLine+12, "%s", szAtomName);
                                sscanf(szLine+26, "%d", &(Atoms[nAtom[i]][i].No
deIndex));


                                if(szLine[17] == ' ')
                                {
                                        sscanf(szLine+17, "%s", szResName);
                                }
                                else if(szLine[17] == 'A')      //only read atom
 in chain A
                                {
                                        sscanf(szLine+18, "%s", szResName);
                                }
                                else
                                {
                                        continue;
                                }

                                if(strncmp(szResName, "ARGH", 4)==0)
                                {
                                        strcpy(szResName, "ARG");
                                }
                                if(strncmp(szResName, "LYSH", 4)==0)
```

```
                                {
                                        strcpy(szResName, "LYS");
                                }

                                if((strncmp(szResName, "ILE", 3)==0) && (Atoms[n
Atom[i]][i].NodeIndex == 1)) // N−terminal residue:
                                {
                                        strcpy(szResName, "ILEN");
                                }
                                if((strncmp(szResName, "MET", 3)==0) && (Atoms[
nAtom[i]][i].NodeIndex == 99)) // C−terminal residue;
                                {
                                        strcpy(szResName, "METC");
                                }
                                strcpy(Atoms[nAtom[i]][i].AtmName, szAtomName);

                                //ASSIGNING the atom type for all side−chain ato
ms
                                if((strcmp(szAtomName, "N")!=0) && (strncmp(szA
tomName, "H", 1)!=0) && (strncmp(szAtomName, "1", 1)!=0) && (strncmp(szAtomName
, "2", 1)!=0) && (strcmp(szAtomName, "CA")!=0) && (strcmp(szAtomName, "C")!=0)
&& (strcmp(szAtomName, "O")!=0) && (strcmp(szAtomName, "O1")!=0) && (strcmp(szA
tomName, "O2")!=0))
                                {
                                        Atoms[nAtom[i]][i].Atom_Type = Get_Atom_
Type(szAtomName);
                                }
                                Atoms[nAtom[i]][i].Res_Type = Get_AA_Type(szResN
ame);

                                ReadCoord = sscanf(szLine+30, "%lf %lf %lf", &(Ato
ms[nAtom[i]][i].x), &(Atoms[nAtom[i]][i].y), &(Atoms[nAtom[i]][i].z));

                                if(ReadCoord != 3)
                                {
                                        printf("Fail to read coordinates from PDB file: %s, Ato
m %d AtomName = %s ResName = %s\nQuit\n",
                                        s, nAtom[i]+1, szAtomName, szResName);
                                        fclose(input);
                                        exit(1);
                                }

                                nAtom[i]++;

                        }

                        ResNum[i] = Atoms[nAtom[i]−1][i].NodeIndex − Atoms[0][i]
.NodeIndex + 1;
                        k++;

                }
                fclose(input);
                AssignAtomRadii();

        }

        //printf("READING COMPLETE\n\n");


}

void ExportSnapshot(int Index1, int Index2)     //export the current protein str
ucture. It can be used for resumed simulation from unexpected termination
{
        FILE *fOut1;
        FILE *fOut2;

        int i;
        char szName1[256];
```



```
        char szName2[256];
        char AA_Name_0[5];
        char AA_Name_1[5];


        sprintf(szName1, "D76N_I1_PAIRS_ID1=%d_ID2=%d_DIR=%d_newcf__a.pdb", Index1,I
ndex2,(int)DIR);
        sprintf(szName2, "D76N_I1_PAIRS_ID1=%d_ID2=%d_DIR=%d_newcf__b.pdb", Index1,I
ndex2,(int)DIR);


        fOut1 = fopen(szName1, "w");
        fOut2 = fopen(szName2, "w");

        fprintf(fOut1, "REMARK PDB file generated by GO potential code written by Lei Huang.\n");
        //fprintf(fOut1, "REMARK E = %7.3lf Rg = %7.3lf  CaRMSD = %7.3lf \n", (d
ouble)Conf[0].contEn, Conf[0].Rg, Conf[0].RMSD);
        for(i=0; i<nAtom[0]; i++)
        {
                strcpy(AA_Name_0, AA_Name[Atoms[i][0].Res_Type]);

                if(strncmp(AA_Name_0, "ARGH", 4)==0)
                {
                        strcpy(AA_Name_0, "ARG");
                }
                if(strncmp(AA_Name_0, "LYSH", 4)==0)
                {
                        strcpy(AA_Name_0, "LYS");
                }
                if(strncmp(AA_Name_0, "HISH", 4)==0)
                {
                        strcpy(AA_Name_0, "HIS");
                }
                if(strncmp(AA_Name_0, "METC", 4)==0)
                {
                        strcpy(AA_Name_0, "MET");
                }
                if(strncmp(AA_Name_0, "ILEN", 4)==0)
                {
                        strcpy(AA_Name_0, "ILE");
                }

                if((strncmp(Atoms[i][0].AtmName, "HD11", 4)==0) || (strncmp(Ato
ms[i][0].AtmName, "HD12", 4)==0) || (strncmp(Atoms[i][0].AtmName, "HD21", 4)==0
) || (strncmp(Atoms[i][0].AtmName, "HD22", 4)==0) || (strncmp(Atoms[i][0].AtmNa
me, "HE11", 4)==0) || (strncmp(Atoms[i][0].AtmName, "HE12", 4)==0) || (strncmp(A
toms[i][0].AtmName, "HE21", 4)==0) || (strncmp(Atoms[i][0].AtmName, "HE22", 4)==
0) || (strncmp(Atoms[i][0].AtmName, "HH11", 4)==0) || (strncmp(Atoms[i][0].AtmN
ame, "HH12", 4)==0) || (strncmp(Atoms[i][0].AtmName, "HH21", 4)==0) || (strncmp
(Atoms[i][0].AtmName, "HH22", 4)==0))
                {
                        fprintf(fOut1, "ATOM%7d %−4s %−3s A%4d    %8.3lf%8.3lf%8.3lf  1.00
 0.00\n",
                        i+1, Atoms[i][0].AtmName, AA_Name_0, Atoms[i][0].NodeInd
ex, Atoms[i][0].x,Atoms[i][0].y,Atoms[i][0].z);
                }
                else
                {
                        fprintf(fOut1, "ATOM%7d  %−3s %−3s A%4d    %8.3lf%8.3lf%8.3lf  1.0
0  0.00\n",
                        i+1, Atoms[i][0].AtmName, AA_Name_0, Atoms[i][0].NodeInd
ex, Atoms[i][0].x,Atoms[i][0].y,Atoms[i][0].z);
                }

        }

        fprintf(fOut2, "REMARK PDB file generated by GO potential code written by Lei Huang.\n");
        //fprintf(fOut2, "REMARK E = %7.3lf Rg = %7.3lf  CaRMSD = %7.3lf \n", (d
ouble)Conf[1].contEn, Conf[1].Rg, Conf[1].RMSD);
```

```
        for(i=0; i<nAtom[1]; i++)
        {
                strcpy(AA_Name_1, AA_Name[Atoms[i][1].Res_Type]);

                if(strncmp(AA_Name_1, "ARGH", 4)==0)
                {
                        strcpy(AA_Name_1, "ARG");
                }
                if(strncmp(AA_Name_1, "LYSH", 4)==0)
                {
                        strcpy(AA_Name_1, "LYS");
                }
                if(strncmp(AA_Name_1, "HISH", 4)==0)
                {
                        strcpy(AA_Name_1, "HIS");
                }
                if(strncmp(AA_Name_1, "METC", 4)==0)
                {
                        strcpy(AA_Name_1, "MET");
                }
                if(strncmp(AA_Name_1, "ILEN", 4)==0)
                {
                        strcpy(AA_Name_1, "ILE");
                }

                if((strncmp(Atoms[i][1].AtmName, "HD11", 4)==0) || (strncmp(Ato
ms[i][1].AtmName, "HD12", 4)==0) || (strncmp(Atoms[i][1].AtmName, "HD21", 4)==0
) || (strncmp(Atoms[i][1].AtmName, "HD22", 4)==0) || (strncmp(Atoms[i][1].AtmNa
me, "HE11", 4)==0) || (strncmp(Atoms[i][1].AtmName, "HE12", 4)==0) || (strncmp(A
toms[i][1].AtmName, "HE21", 4)==0) || (strncmp(Atoms[i][1].AtmName, "HE22", 4)==
0) || (strncmp(Atoms[i][1].AtmName, "HH11", 4)==0) || (strncmp(Atoms[i][1].AtmN
ame, "HH12", 4)==0) || (strncmp(Atoms[i][1].AtmName, "HH21", 4)==0) || (strncmp
(Atoms[i][1].AtmName, "HH22", 4)==0))
                {
                        fprintf(fOut2, "ATOM%7d %−4s %−3s A%4d    %8.3lf%8.3lf%8.3lf  1.00
 0.00\n",
                        i+1, Atoms[i][1].AtmName, AA_Name_1, Atoms[i][1].NodeInd
ex, Atoms[i][1].x,Atoms[i][1].y,Atoms[i][1].z);
                }
                else
                {
                        fprintf(fOut2, "ATOM%7d  %−3s %−3s A%4d    %8.3lf%8.3lf%8.3lf  1.0
0  0.00\n",
                        i+1, Atoms[i][1].AtmName, AA_Name_1, Atoms[i][1].NodeInd
ex, Atoms[i][1].x,Atoms[i][1].y,Atoms[i][1].z);
                }

        }

        fclose(fOut1);
        fclose(fOut2);

}

//pdb−compatible screen output
//for (i=0;i<nAtom;i++)
//{
//      printf("ATOM%7d %−3s    %−3s    A%4d    %8.3lf  %8.3lf  %8.3lf  1.00 0.0
0\n",
//      i+1, Atoms[i].AtmName, AA_Name[Atoms[i].Res_Type],Atoms[i].NodeIndex, At
oms[i].x,Atoms[i].y,Atoms[i].z);
//}

//output of r0−assignment, tested, works!
//for (i=0;i<2;i++)
//{
//      for (j=0;j<nAtom[i];j++)
//      {
//              printf("%d  ATOM%7d     %−3s    %−3s    A%4d    %8.3lf  %8.3lf
```



```
%8.3lf  1.00 %8.3lf\n",i,j+1, Atoms[j][i].AtmName,
//              AA_Name[Atoms[j][i].Res_Type],Atoms[j][i].NodeIndex, Atoms[j][i]
.x, Atoms[j][i].y, Atoms[j][i].z, Atoms[j][i].r0);
//      }
//}

//control output of protein folding observables
//printf("%d    %d      %d\n",Conf[0].contEn,Conf[1].contEn,Conf[0].contEn+Conf[
1].contEn);
//printf("%lf   %lf     %lf\n",Conf[0].Rg,Conf[1].Rg,Conf[0].Rg+Conf[1].Rg);
//printf("%lf   %lf     %lf\n\n",Conf[0].RMSD,Conf[1].RMSD,Conf[0].RMSD+Conf[1].
RMSD);

//TEST OUTPUT for stack structure
/*for (j=0;j<nAtom[1];j++)
{
        printf("ATOM%7d %−3s    %−3s    A%4d    %8.3lf  %8.3lf  %8.3lf  1.00 %8.
3lf\n",j+1, Moved[j].AtmName,
                AA_Name[Moved[j].Res_Type],Moved[j].NodeIndex, Moved[j].x, Moved
[j].y, Moved[j].z, Moved[j].r0);
}*/

double abbs(double a)
{
        if (a < 0)
                return −a;
        else
                return a;
}
int IsANumber(char c)                   //to check whether c is a number [0−9] o
r not
{
        if((c >= '0') && (c<='9'))
        {
                return 1;
        }
        else
        {
                return 0;
        }
}
int Get_AA_Type(char SzAAName[])
{
        int AA_Type=−1;
        int i;

        for (i=0;i<NUM_AA_TYPE; i++)
        {
                if (strcmp(SzAAName, AA_Name[i])==0)
                {
                        AA_Type=i;
                        break;
                }
        }

        if (AA_Type<0)
        {
                printf("Fatal error in AA matching!");
                exit(1);
        }
        return AA_Type;
}
int Get_Atom_Type(char SzAtomName[])
{
        int Atom_Type=−1;
        int i;

        for (i=0;i<NUM_Atom_TYPE; i++)
        {
```

```
                if (strcmp(SzAtomName, atom_name[i])==0)
                {
                        Atom_Type=i;
                        break;
                }
        }

        if (Atom_Type<0)
        {
                printf("Fatal error in Atom matching!");
                exit(1);
        }
        return Atom_Type;
}
void AssignAtomRadii(void)
{
        int n,i, Type;

        for (i=0;i<2;i++)
        {
                for (n=0;n<nAtom[i];n++)
                {
                        Type = QueryAtomType(Atoms[n][i].AtmName, AA_Name[Atoms[
n][i].Res_Type]);
                        Atoms[n][i].r0=radii[Type];
                }
        }


}
int QueryAtomType(char *s, char *res)   //from the atom name, s, and residue nam
e, res, to the type of atoms. The type is used to assign the radii. This procedu
re is same as the REMC code
{
        if (!strncmp(s,"C",1)) {
                if (!strcmp(s,"C") || (!strcmp(s,"CG") && (!strcmp(res,"ASN") |
| !strcmp(res,"ASP") || !strcmp(res,"ASPH") || !strcmp(res,"HIS") || !strcmp(re
s,"HISH") || !strcmp(res,"PHE") || !strcmp(res,"TYR") ||
                !strcmp(res,"TRP"))) || (!strcmp(s,"CD") && (!strcmp(res,"GLN")
 || !strcmp(res,"GLU") || !strcmp(res,"GLUH"))) || (!strcmp(s,"CZ") && (!strcm
p(res,"ARG") || !strcmp(res,"ARGH") || !strcmp(res,"TYR"))) ||  (!strcmp(s,"CD
2") && !strcmp(res,"TRP")) || (!strcmp(s,"CE2") && !strcmp(res,"TRP")))
                        return 0;
                else if ((!strcmp(s,"CD1") && (!strcmp(res,"PHE") || !strcmp(re
s,"TYR"))) || (!strcmp(s,"CD2") && (!strcmp(res,"HIS") || !strcmp(res,"HISH") |
| !strcmp(res,"PHE") || !strcmp(res,"TYR"))) || (!strcmp(s,"CZ") && !strcmp(res
,"PHE")) || (!strcmp(res,"TRP") && (!strncmp(s,"CH",2) || !strncmp(s,"CZ",2) ||
 !strcmp(s,"CE3") || !strcmp(s,"CD1"))))
                        return 1;
                else if ((!strcmp(s,"CA") && strcmp(res,"GLY")) || (!strcmp(s,"
CB") && (!strcmp(res,"ILE") || !strcmp(res,"ILEN") || !strcmp(res,"THR") || !str
cmp(res,"VAL"))) || (!strcmp(s,"CG") && !strcmp(res,"LEU")))
                        return 2;
                else if ((!strcmp(s,"CB") && !strcmp(res,"ALA")) || (!strcmp(s,
"CD1") && (!strcmp(res,"ILE") || !strcmp(res,"ILEN") || !strcmp(res,"LEU"))) ||
(!strcmp(s,"CD2") &&
                !strcmp(res,"LEU")) || (!strcmp(s,"CG1") && !strcmp(res,"VAL"))
 || (!strcmp(s,"CG2") && (!strcmp(res,"ILE") || !strcmp(res,"ILEN") || !strcmp(r
es,"THR") || !strcmp(res,"VAL"))))
                        return 4;
                else
                        return 3;
        }
        else if (!strncmp(s,"N",1)) {
                if ((!strcmp(s,"N") && !strcmp(res,"PRO")) || (!strcmp(s,"NE2")
 && (!strcmp(res,"HIS") || !strcmp(res,"HISH"))))
                        return 5;
                else if (!strcmp(s,"NZ"))
                        return 8;
```



```
                else if (!strcmp(s,"N") || (!strcmp(s,"ND1") && (!strcmp(res,"H
IS") || !strcmp(res,"HISH"))) || (!strcmp(s,"NE") && (!strcmp(res,"ARG") || !st
rcmp(res,"ARGH"))) || (!strcmp(s,"NE1") && !strcmp(res,"TRP")))
                        return 6;
                else
                        return 7;
        }
        else if (!strncmp(s,"O",1)) {
                if (!strcmp(s,"OH") || !strcmp(s,"OG") || !strcmp(s,"OG1"))
                        return 10;
                else
                        return 9;
        }
        else if (!strncmp(s,"S",1)) {
                if (!strcmp(res,"MET") || !strcmp(res,"METC") || !strcmp(res,"
METN"))
                        return 11;
                else
                        return 12;
        }
        else if (!strncmp(s,"H",1))
        {
                if ((!strcmp(s,"HD1") || !strcmp(s,"HD2") || !strcmp(s,"HE1") |
| !strcmp(s,"HE2") || !strcmp(s,"HZ")) && !strcmp(res,"PHE"))
                {
                        return 14;
                }
                else if ((!strcmp(s,"HD1") || !strcmp(s,"HE3") || !strcmp(s,"HZ
2") || !strcmp(s,"HZ3") || !strcmp(s,"HH2")) && !strcmp(res,"TRP"))
                {
                        return 14;
                }
                else if ((!strcmp(s,"HD1") || !strcmp(s,"HD2") || !strcmp(s,"HE
1") || !strcmp(s,"HE2")) && !strcmp(res,"TYR"))
                {
                        return 14;
                }
                else if ((!strcmp(s,"HD2") || !strcmp(s,"HE1")) && !strcmp(res,
"HIS"))
                {
                        return 14;
                }
                else if ((!strcmp(s,"HD2") || !strcmp(s,"HE1")) && !strcmp(res,
"HISH"))
                {
                        return 14;
                }
                else
                {
                        return 13;
                }
        }
        else
        {
                return -500;
        }
}


//RANDOM NUMBER GENERATOR FUNCTIONS
#define FALSE 0
#define TRUE 1

/*
   This Random Number Generator is based on the algorithm in a FORTRAN
   version published by George Marsaglia and Arif Zaman, Florida State
   University; ref.: see original comments below.
   At the fhw (Fachhochschule Wiesbaden, W.Germany), Dept. of Computer
   Science, we have written sources in further languages (C, Modula-2
```

```
   Turbo−Pascal(3.0, 5.0), Basic and Ada) to get exactly the same test
   results compared with the original FORTRAN version.
   April 1989
   Karl−L. Noell <NOELL@DWIFH1.BITNET>
      and  Helmut  Weber <WEBER@DWIFH1.BITNET>

   This random number generator originally appeared in "Toward a Universal
   Random Number Generator" by George Marsaglia and Arif Zaman.
   Florida State University Report: FSU−SCRI−87−50 (1987)
   It was later modified by F. James and published in "A Review of Pseudo−
   random Number Generators"
   THIS IS THE BEST KNOWN RANDOM NUMBER GENERATOR AVAILABLE.
   (However, a newly discovered technique can yield
   a period of 10^600. But that is still in the development stage.)
   It passes ALL of the tests for random number generators and has a period
   of 2^144, is completely portable (gives bit identical results on all
   machines with at least 24−bit mantissas in the floating point
   representation).
   The algorithm is a combination of a Fibonacci sequence (with lags of 97
   and 33, and operation "subtraction plus one, modulo one") and an
   "arithmetic sequence" (using subtraction).

   Use IJ = 1802 & KL = 9373 to test the random number generator. The
   subroutine RANMAR should be used to generate 20000 random numbers.
   Then display the next six random numbers generated multiplied by 4096*4096
   If the random number generator is working properly, the random numbers
   should be:
           6533892.0  14220222.0  7275067.0
           6172232.0  8354498.0   10633180.0
*/

/* Globals */
double u[97],c,cd,cm;
int i97,j97;
int test = FALSE;

/*
   This is the initialization routine for the random number generator.
   NOTE: The seed variables can have values between:    0 <= IJ <= 31328
                                                        0 <= KL <= 30081
   The random number sequences created by these two seeds are of sufficient
   length to complete an entire calculation with. For example, if sveral
   different groups are working on different parts of the same calculation,
   each group could be assigned its own IJ seed. This would leave each group
   with 30000 choices for the second seed. That is to say, this random
   number generator can create 900 million different subsequences −− with
   each subsequence having a length of approximately 10^30.
*/
void RandomInitialise(int ij,int kl)
{
   double s,t;
   int ii,i,j,k,l,jj,m;

   /*
      Handle the seed range errors
         First random number seed must be between 0 and 31328
         Second seed must have a value between 0 and 30081
   */
   if (ij < 0 || ij > 31328 || kl < 0 || kl > 30081) {
               ij = 1802;
               kl = 9373;
   }

   i = (ij / 177) % 177 + 2;
   j = (ij % 177)       + 2;
   k = (kl / 169) % 178 + 1;
   l = (kl % 169);

   for (ii=0; ii<97; ii++) {
```



```
      s = 0.0;
      t = 0.5;
      for (jj=0; jj<24; jj++) {
         m = (((i * j) % 179) * k) % 179;
         i = j;
         j = k;
         k = m;
         l = (53 * l + 1) % 169;
         if (((l * m % 64)) >= 32)
            s += t;
         t *= 0.5;
      }
      u[ii] = s;
   }

   c    = 362436.0 / 16777216.0;
   cd   = 7654321.0 / 16777216.0;
   cm   = 16777213.0 / 16777216.0;
   i97  = 97;
   j97  = 33;
   test = TRUE;
}

/*
   This is the random number generator proposed by George Marsaglia in
   Florida State University Report: FSU−SCRI−87−50
*/
double RandomUniform(void)
{
   double uni;

   /* Make sure the initialisation routine has been called */
   if (!test)
        RandomInitialise(1802,9373);

   uni = u[i97−1] − u[j97−1];
   if (uni <= 0.0)
      uni++;
   u[i97−1] = uni;
   i97−−;
   if (i97 == 0)
      i97 = 97;
   j97−−;
   if (j97 == 0)
      j97 = 97;
   c −= cd;
   if (c < 0.0)
      c += cm;
   uni −= c;
   if (uni < 0.0)
      uni++;

   return(uni);
}

/*
  ALGORITHM 712, COLLECTED ALGORITHMS FROM ACM.
  THIS WORK PUBLISHED IN TRANSACTIONS ON MATHEMATICAL SOFTWARE,
  VOL. 18, NO. 4, DECEMBER, 1992, PP. 434−435.
  The function returns a normally distributed pseudo−random number
  with a given mean and standard devaiation.  Calls are made to a
  function subprogram which must return independent random
  numbers uniform in the interval (0,1).
  The algorithm uses the ratio of uniforms method of A.J. Kinderman
  and J.F. Monahan augmented with quadratic bounding curves.
*/
double RandomGaussian(double mean,double stddev)
{
   double  q,u,v,x,y;
```

```
      /*
              Generate P = (u,v) uniform in rect. enclosing acceptance region
      Make sure that any random numbers <= 0 are rejected, since
      gaussian() requires uniforms > 0, but RandomUniform() delivers >= 0.
      */
   do {
      u = RandomUniform();
      v = RandomUniform();
        if (u <= 0.0 || v <= 0.0) {
        u = 1.0;
        v = 1.0;
        }
      v = 1.7156 * (v − 0.5);

      /*  Evaluate the quadratic form */
      x = u − 0.449871;
        y = fabs(v) + 0.386595;
      q = x * x + y * (0.19600 * y − 0.25472 * x);

      /* Accept P if inside inner ellipse */
      if (q < 0.27597)
                        break;

      /*  Reject P if outside outer ellipse, or outside acceptance region */
    } while ((q > 0.27846) || (v * v > −4.0 * log(u) * u * u));

    /*  Return ratio of P's coordinates as the normal deviate */
    return (mean + stddev * v / u);
}

/*
   Return random integer within a range, lower −> upper INCLUSIVE
*/
int RandomInt(int lower,int upper)
{
   return((int)(RandomUniform() * (upper − lower + 1)) + lower);
}

/*
   Return random float within a range, lower −> upper
*/
double RandomDouble(double lower,double upper)
{
   return((upper − lower) * RandomUniform() + lower);
}
```