Jihočeská univerzita v Českých Budějovicích
Přírodovědecká fakulta

Molekulární simulace proteinů

Bakalářská práce

Matyáš Suchý

Školitel: RNDr. Zdeněk Futera, Ph.D.

České Budějovice 2023

Suchý M., 2023: Molekulární simulace proteinů. [Molecular simulations of proteins. Bc. Thesis, in Czech.] – 69 p., Faculty of Science, University of South Bohemia, České Budějovice, Czech Republic.

Abstract

The bachelor thesis is devoted to computer simulations of proteins, which are introduced and described here in a way so the work can serve as a primary explanatory text for students of non-physical, biological fields. The work reacts to a current situation where a majority of available textbooks, tutorials, software manuals, or scientific reviews are written from physics or, in some cases, chemistry points of view, and they discourage students from biological fields by their rather technical style. Here, attention to the practical aspects of basic protein simulations is paid as an introduction to this research approach. In the thesis, protein structure databases, visualization techniques, molecular-mechanical description, molecular dynamics simulations, and basic data analyses are described and discussed. The methods are demonstrated on a small globular protein actin, and the most popular visualization and simulation software is employed for the practical parts.

Prohlašuji, že svoji bakalářskou práci jsem vypracoval samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

12. 4. 2023
Datum

Matyas Suchy
Digitálně podepsal Matyas Suchy
Datum: 2023.04.12 14:10:02 +02'00'

Matyáš Suchý

Rád bych tímto poděkoval RNDr. Zdeňkovi Futerovi, Ph.D., za vedení bakalářské práce, veškerou pomoc a velmi trpělivý přístup. Za pomoc s bakalářskou prací také děkuji týmu vedeným RNDr. Zdeňkem Futerou, Ph.D.. Za poskytnutí výpočetních prostředků a datového úložiště patří dík katedře fyziky na Přírodovědecké fakulty Jihočeské univerzity v Českých Budějovicích a MetaCentru VO. V neposlední řadě děkuji své rodině za porozumění a podporu, které se mi dostalo během psaní.

Obsah

Úvod

Proteiny jsou velmi významné v biologii živých buněk. Jsou v každé buňce organismu a jsou stále obnovovány. Bílkoviny slouží jako základ stavby a vyvíjení se živých struktur. Jsou používány pro obnovování a zachovávání těl organismů. Proteiny jsou významné v mnoha směrech, mezi jejich další význam patří stavba enzymů a hormonů na které se podílejí. Napomáhají držet konstantní osmotický tlak, a podílejí se na spoustě dalších významných činnostech v živých organismech. Významnou roli hrají proteiny také v biotechnologických aplikacích. Bílkoviny jsou využívány například v biokatalýze, kde slouží jako katalyzátory, tedy zrychlují chemické reakce a snižují aktivační energii. Mohou být užívány také jako specifické senzory. Dnes jsou bílkoviny využívány také v oboru bioelektroniky, kde jsou zkoumány jejich vodivé vlastnosti.

Bílkoviny jsou definovány strukturami, z kterých jsou složeny. Máme čtyři strukturní úrovně (primární, sekundární, terciární a kvartérní). Těmito úrovněmi máme popsán každý protein, který je znám. Od těchto struktur se odvíjí také funkce jednotlivých proteinů. Funkce proteinů může být transportní, enzymatická, zásobní, ochranná, kontraktilní, hormonální, strukturní a toxiny. Tyto funkce jsou vždy definovány podle struktury, kterou protein zaujímá. Například díky specifické geometrii aktivních míst proteinů víme, kde mají enzymy svá aktivní místa. Protein může být jednoduchý (obsahuje jen aminokyseliny) nebo složený (obsahuje nebílkovinovou složku). Díky struktuře metaloproteinu můžeme například zjistit, jak je ovlivněn redoxní potenciál. Dále můžeme ze struktury jednotlivých proteinů vyčíst, jak svou geometrií ovlivňují propustnost iontových kanálů.

Význam počítačových simulací pro studium proteinů spočívá v tom, že můžeme vzít atomární strukturu proteinu získanou z experimentu, která nemusí být vždy dostatečně detailní a nasimulovat si, jak se bude struktura pohybovat nebo co se s ní bude dít, pokud bude například ve vodném roztoku. Získáme tak atomární rozlišení proteinu, který se pohybuje. Dále můžeme zanalyzovat simulaci a získat tak detailní informace například o vibračním spektru daného proteinu, jeho polohách během simulace oproti experimentálním výsledkům, enzymatické reakční mechanismy, mechanismy přenosu náboje v reakčních centrech.

Obsahem této práce je shrnutí jednotlivých kroků, které je potřeba vykonat při analýze proteinů pomocí počítačových simulací. Od získání experimentálních dat z databáze, přes vizualizaci a optimalizaci, až po základní simulace a jejich analýzy. Tato bakalářská práce má sloužit jako manuál a příručka pro další studenty a začátečníky v počítačových simulacích.

Tato práce je uspořádána, tak jak se běžně postupuje při studiu proteinů pomocí počítačových simulací. Nejprve jsou stručně shrnuty základní vlastnosti proteinů a představen systém, na kterém budou jednotlivé postupy demonstrovány. Dále se věnujeme proteinové databázi, kde je možné získat atomovou strukturu proteinu, jejíž znalost je nezbytná k provedení simulací. Dále jsou představeny vizualizační nástroje, tedy software, který se obvykle používá k zobrazení proteinových struktur i simulačních trajektorií. Následně popisujeme, co je to molekulárně mechanický popis, jaké jsou jeho přednosti a limity, a zmiňujeme dostupná silová pole. Poté jsou představeny samotné simulační techniky a jejich konkrétní provedení v programových balících Amber a Gromacs. Nakonec diskutujeme základní analýzy, které se běžně při studiu proteinů provádějí. V závěru práce je shrnutí celé metodologie počítačových simulací proteinů, tak jak je zde představena včetně doporučení dalších postupů.

1 Proteiny

Proteiny, v české literatuře známé také jako bílkoviny, jsou biologické makromolekuly přítomné ve všech živých buňkách. Zde mají strukturní a zásobní funkce. Z pohledu struktury dělíme proteiny na jednoduché a složené. Jednoduché jsou tvořeny pouze aminokyselinami. Složené (konjugované) obsahují kromě aminokyselin v molekule také nebílkovinovou složku, což může být sacharid, nukleová kyselina, lipid, ion kovu, nebo nějaký jiný organický kofaktor. Podle toho se složené proteiny dělí na glykoproteiny, nukleoproteiny, lipoproteiny, metaloproteiny, nebo obecné holoproteiny. Z funkčního hlediska rozlišujeme enzymy, které katalyzují chemické reakce, zásobní proteiny (např. ovalbumin ve vaječném bílku), transportní proteiny (např. hemoglobin přenášející kyslík v červených krvinkách), ochranné proteiny (např. imunoglobulin), kontraktilní proteiny (např. myozin podílející se na svalových kontrakcích), hormony (např. inzulín), toxiny (např. hadí jed) a strukturní proteiny (aktin nebo kolagen) [1-3].

Základními stavebními kameny proteinů jsou aminokyseliny, což jsou malé chirální chemické sloučeniny s obecným vzorcem NH_2-CH(-R)-COOH, které se liší svým postranním řetězcem (R, tzv. reziduum). Přestože v přírodě existuje velké množství různých aminokyselin, pouze 20 tzv. proteinogenních aminokyselin tvoří proteinové struktury (seznam těchto aminokyselin je uveden v Tabulce 1). Všechny proteinogenní aminokyseliny mají L-enantiomerní (tj. levotočivou) formu. Na základě chemických vlastností jejich postranních skupin dělíme aminokyseliny na alifatické (Ala, Gly, Ile, Leu, Pro, Val), aromatické (His, Phe, Trp, Tyr), kyselé (Asp, Glu), bazické (Arg, Lys), aminokyseliny obsahující síru (Cys, Met), aminy (Asn, Gln), anebo hydroxyly (Ser, Thr). Někdy se mezi proteinogenní aminokyseliny řadí také selenocystein (Sec), což je cystein, kde je síra nahrazena selenem. Tato aminokyselina se ale v proteinových strukturách vyskytuje poměrně zřídka (např. glutathionperoxidáza) [1].

Sekvence aminokyselin, která tvoří tzv. primární strukturu proteinů (viz. níže) je kódovaná v deoxyribonukleové kyselině (DNA). Ta se v první fázi syntézy bílkovin (tzv. genová exprese) přepisuje na mediátorovou ribonukleovou kyselinu (messenger RNA nebo jen mRNA), což je proces známý jako transkripce DNA, který katalyzovaný RNA polymerázou uvnitř buněčného jádra. Následně mRNA přechází z jádra do cytoplasmy a pomocí ribozomu dochází k samotné proteinové syntéze, kdy se jednotlivé aminokyseliny spojují pomocí peptidických vazeb (R_1-NH_2 + HOOC-R_2 → R_1-NH-CO-R_2 + H_2O) do polymerních řetězců.

Pořadí aminokyselin je určeno sekvencí nukleových bázích v DNA, resp. mRNA, a naznačený proces genové exprese tvoří centrální dogma molekulární biologie [1-2].

***Tabulka 1**: Seznam proteinogenních aminokyselin, jejich třípísmenných (3L) a jednopísmenných (1L) kódů a chemického složení jejich postranních řetězců.*

Aminokyselina	**3L kód**	**1L kód**	**Postranní řetězec**
Alanin	Ala	A	$-CH_3$
Arginin	Arg	R	$-(CH_2)_3-NH-C(NH)NH_2$
Leucin	Leu	L	$-CH_2-CH-(CH_3)_2$
Isoleucin	Ile	I	$-CH(CH_3)-CH_2-CH_3$
Valin	Val	V	$-CH-(CH_3)_2$
Prolin	Pro	P	$-C_5H_9N$
Fenylalanin	Phe	F	$-CH_2-C_6H_5$
Tryptofan	Trp	W	$-CH_2-C_8H_5NH$
Methionin	Met	M	$-CH_2-CH_2-S-CH_3$
Glycin	Gly	G	-H
Serin	Ser	S	$-CH_2-OH$
Threonin	Thr	T	$-CH(OH)-CH_3$
Tyrosin	Tyr	Y	$-CH_2-C_6H_4-OH$
Asparagin	Asn	N	$-CH_2-C(O)-H_2N$
Glutamin	Gln	Q	$-CH_2-CH_2-C(O)-H_2N$
Cystein	Cys	C	$-CH_2-HS$
Lysin	Lys	K	$-(CH_2)_4-H_2N$
Histidin	His	H	$-CH_2-C_3H_3N_2$
Kys. Asparagová	Asp	D	$-CH_2$
Kys. Glutamová	Glu	E	$-(CH_2)_2-COOH$

1.1 Struktury proteinů

U proteinů rozlišujeme čtyři úrovně strukturního uspořádání atomů do výsledné 3D geometrie. Ty jsou známe jako primární, sekundární, terciární a kvartérní struktura protein.

Primární struktura

Primární struktura proteinů je určena pořadím aminokyselin v polypeptidickém řetězci. Může tam být zahrnuta popřípadě i poloha disulfidových vazeb tedy kovalentních vazeb mezi cysteiny, které zpravidla nejsou v rámci peptidického řetězce v těsné blízkosti. Primární struktura se experimentálně získává sekvenováním daného polypeptidického řetězce. Pořadí aminokyselin určuje konformaci a biologickou aktivitu proteinů [1].

Sekundární struktura

Sekundární struktura určuje základní typy 3D uspořádání primárního peptidického řetězce, které zpravidla stabilizováno vodíkovými vazbami. Jde o pravidelně se opakující struktury, které mají nějaký typický geometrický tvar. Sekundární struktura byla nejdříve teoretická a později byla potvrzena experimentálně metodami rentgenové krystalografie [1]. Nejznámějšími a nejčastějšími typy sekundárních struktur jsou tzv. α-helix (šroubovice) a β-sheet (skládaný list).

α-helix je často se vyskytujícím strukturním motivem u globulárních bílkovin. Jde většinou o pravotočivou šroubovici, která je stabilizována vodíkovými můstky. Její rozměry jsou přesně dané: 3,6 aminokyselinových zbytků na jednu otáčku (závit). Výška jednoho závitu je 0,54 nm (podél osy spirály). Vzdálenost mezi ekvivalentními atomy sousedních zbytků, tvořících hlavní řetězec, měření podél osy spirály = 0,15 nm [1].

β-sheet je geometrický útvar, který vypadá jako harmonika. Úhel, který svírají jednotlivé plochy je 153° (jde o pohled z boku na strukturu). α-uhlíky jsou zde zastoupeny z více oblastí na rozdíl od α-helixu. Jde o úseky 5-10 aminokyselin. Stabilizátorem polypeptidů jsou vodíkové vazby mezi vodíkem amidu a kyslíkem karbonylu sousedních řetězců [1].

Terciární struktura

Bílkoviny obsahující sekundární struktury se organizují do kompaktních doménových jednotek, které spojuje polypeptidická kostra (backbone). Vztahy těchto domén popisuje právě terciální struktura. Dále popisuje různé způsoby, kterými se různě vzdálené aminokyseliny dostávají do těsné blízkosti v rámci primární struktury. Terciální struktura také popisuje vazby stabilizující konformace [1-2].

Kvartérní struktura

Některé globulární bílkoviny jsou složeny z více polypeptidických řetězců. Tyto bílkoviny se nazývají oligomerní. Jednotlivé řetězce pak tvoří proteinové podjednotky. Vzájemná orientace těchto podjednotek v molekule oligomerního proteinu se označuje kvartérní struktura [1-2].

1.2 Aktin

Aktin je nejvýše zastoupeným proteinem ve všech eukaryotických buňkách. Má nejvíce účastí v interakcích protein-protein ze všech proteinů. Má schopnost přecházet mezi dvěma stavy - monomerním (G-aktin) a vláknitým (F-aktin), za kontroly nukleotidové hydrolýzy, iontů a aktin-vazebných proteinů. Aktin je kritická součást v buněčných funkcích, např. buněčná mobilita, udržování tvaru a polarity buněk, regulaci transkripce. Interakce mezi myozinem a vláknitým aktinem je základem svalové kontrakce [2-5].

Na tomto proteinu budou v rámci této práce demonstrovány jednotlivé kroky, které se při simulacích biomolekul provádějí. Tento protein byl vybrán proto, že je velmi dobře prozkoumaný a jeho data jsou velmi přesné. Konkrétně vybraná struktura proteinu Aktin v databázi PDB, je velmi dobrá pro simulace, protože nemá žádné kofaktory, takže se snadno parametrizuje, zároveň je velmi malý, díky tomu jsou simulace a veškerá práce s proteinem velmi rychlá. Rozlišení je 1.54 Å (angströmů). Protein byl popsán metodou rentgenové difrakce. Použitá struktura aktinu (PDB id 1J6Z) pochází z *Oryctolagus cuniculus* (Králík divoký) [6].

2 Strukturní databáze

2.1 Detekce a predikce proteinových struktur

Pro studium proteinů, analýzu jejich vlastností i provádění počítačových simulací je zásadní znát atomární strukturu těchto makromolekul. Ta se většinou detekuje experimentálně pomocí rentgenové difrakce, popř. nukleární magnetické rezonance či kryogenní mikroskopie [7, 8]. V případě, že je známá jen část struktury, je možné chybějící data doplnit také pomocí tzv. homologního modelování [9, 10]. V poslední době je také možné predikovat proteinové struktury pouze ze znalosti jejich aminokyselinové sekvence, a to pomocí umělé inteligence (AI) [11, 12], kde došlo k výraznému vývoji a zlepšení přesnosti s rozvojem strojového učení a neuronových sítí.

Rentgenová difrakce

Tato metoda slouží k určení atomové a molekulární struktury krystalu. Rentgenové paprsky, které dopadají na krystal, jsou rozptýleny do mnoha směrů. Po změření úhlů a intenzit paprsků po rozptýlení je možné vytvořit 3D obraz hustoty elektronů v krystalu. Díky tomuto obrazu můžeme určit dále střední polohy atomů, chemické vazby a mnoho dalších informací. Touto metodou byla určena převážná část struktur, které se nacházejí v proteinové databázi PDB [7].

Nukleární magnetická rezonance (NMR)

Metoda sloužící k určení struktur proteinů a jejich komplexů. Poskytuje také informace o interakční a konformační dynamice v časovém intervalu pikosekund až sekund (někdy i dní), různých roztocích, ale i v živých buňkách. NMR je založena na 4 krocích, od přípravy vzorku, přes analýzu dat NMR, strukturální výpočty a nakonec posuzování kvality. Samotní nukleární magnetická rezonance využívá magnet, kterému je vystaven určitý protein a na něj jsou vysílány radiofrekvenční signály, zde se pak měří jejich absorbce [7].

Kryogenní mikroskopie (CryoEM)

Metoda pracující s elektronovou mikroskopií, využívající extrémně rychlé zmrazení vzorku ve vodné bázi. Díky velmi rychlému podchlazení, se voda neuspořádá do krystalické mřížky, ale vznikne amorfní led v okolí vzorku, který připomíná sklo. Díky tomu je možné vidět vzorek ve velmi kvalitním rozlišení. Tato metoda nezobrazuje pohyb ani interakce molekul [8].

Homologní modelování

Uváděna jako jedna z nejpřesnějších metod na predikci struktur. Využívá znalost sekvence proteinu a znalost jeho homologního proteinu, tento protein musí mít alespoň 30% schodu v sekvenci s neznámým proteinem. Poté se nechá pracovat software, který dopočítá celkovou strukturu [9, 10].

Metody umělé inteligence (AI)

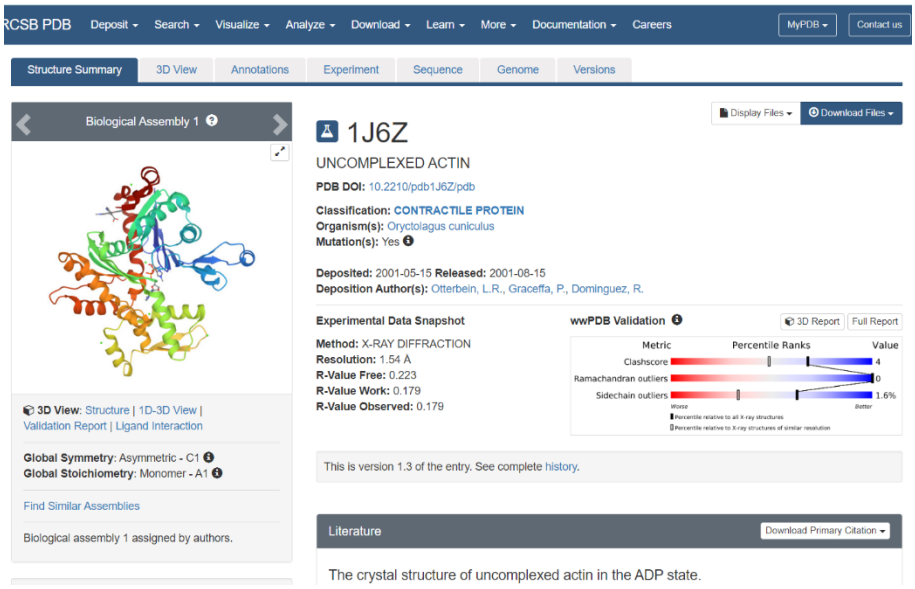
Nová metoda na určování proteinových struktur, která využívá umělou inteligenci. Její boom přišel v roce 2021, kdy byl zpřístupněn software AlphaFold skupiny DeepMind. Tato metoda určuje struktury na základě umělé inteligence, která využívá aminokyselinovou sekvenci a díky informacím z několika sekvenčních databází vytvoří jejich 3D strukturu díky evolučně příbuzným proteinům. Metoda umělé inteligence má velkou budoucnost, rapidně se rozvýjí, ale stále je zde hodně prostoru pro zlepšení [11, 12].

2.2 Proteinová databáze PDB

Proteinová databáze (Protein Data Bank, PDB) [6] je celosvětová online databáze, kam se ukládají 3D strukturní data velkých biologických molekul, jako jsou proteiny a nukleové kyseliny (RNA, DNA). PDB založil v roce 1971 Walter Hamilton v Brookhaven National Laboratory a na počátku obsahovala pouze 7 struktur. Od r. 1998 je PDB spravována organizací Research Collaboratory for Structural Bioinformatics (RCSB), kterou v době jejího založení vedla Helen M. Berman. V r. 2003 byla pak vytvořena Worldwide Protein Data Bank foundation (wwPDB), aby fungovala jako správce jediného archivu PDB makromolekulárních dat. Tyto data jsou volně přístupné celosvětové komunitě. Patří pod ní organizace, které ukládají, zpracovávají a distribuují data PDB [6].

V současné době je v databázi uloženo více než 200 000 záznamů experimentálně určených struktur proteinů a nukleových kyselin. Kromě toho obsahuje databáze nově také struktury predikované pomocí umělé inteligence (AlphaFold, více než 1 000 000 záznamů) [12]. K prohledávání databáze se používá webovské dostupné na hlavní stránce RCSB PBD [6]. Při hledání je kromě jména požadované molekuly možno specifikovat také rozlišení, chemické atributy, či podobnost s jinou strukturou. Jako výsledek vyhledávání se zobrazí seznam struktur splňujících daná kritéria (typicky hledaný protein a všechny jeho dostupné varianty, např. různé mutace, hybridní komplexy s dalšími navázanými molekulami či organickými kofaktory,

super-komplexy několika proteinů, či struktury z různých měření). Po vybrání preferované struktury, se zobrazí její stránka s detailními informacemi o dané struktuře. Ukázka takového záznamu je uvedena na Obrázku 1 [6].



***Obrázek 1**: Ukázka záznamu z proteinové databáze PDB. Na levé straně je zobrazena struktura daného proteinů (v tomto případě aktin), v prostřední části jsou uvedeny základní údaje o této struktuře (původ, detekční metoda, rozlišení, odkaz na původní publikaci) a v horní pravé části je menu umožňující stažení strukturních souřadnic v požadovaném formátu.*

Hlavními informacemi je název např. „Uncomplexed actin“, PBD ID (tento čtyřmístný kód je originální pro každou strukturu, je možné podle něj strukturu vyhledat, ale i ji díky tomu zadat a nechat si jí zobrazit v různých systémech, jako je VMD a další). Dalšími informacemi jsou klasifikace proteinu, organismus, v kterém se daná struktura nachází, jestli je struktura po mutaci. U každé struktury je několik obrázků jako 3D struktury a jsou zde informace o metodě a výsledcích určení struktury (popis metod pro určování struktur viz část 2.1). Dále jsou zde uvedeny i parametry dané struktury. Další důležitou informací je celá sekvence dané struktury, a kde se nachází mutace a další struktury navíc, pokud jsou přítomny. Další blok zahrnuje uvedení informací ohledně navázaných struktur na protein (organické ligandy, ionty, apod.). Poslední blok zahrnuje experimentální data (délka, úhly…).

3 Vizualizační nástroje

3.1 Počítačová grafika

Počítačová grafika se zabývá zobrazováním 3D objektů na 2D obrazovce, přičemž nejde jen o jednoduchou projekci, ale zohledňuje se zde i perspektiva. Dále se zabývá vykreslováním a překrýváním objektů na obrazovce. Dalším důležitým pojmem v počítačové grafice je tzv. teselace povrchů, to znamená, že je vyplněn prostor nepřekrývajícími se geometrickými objekty [13].

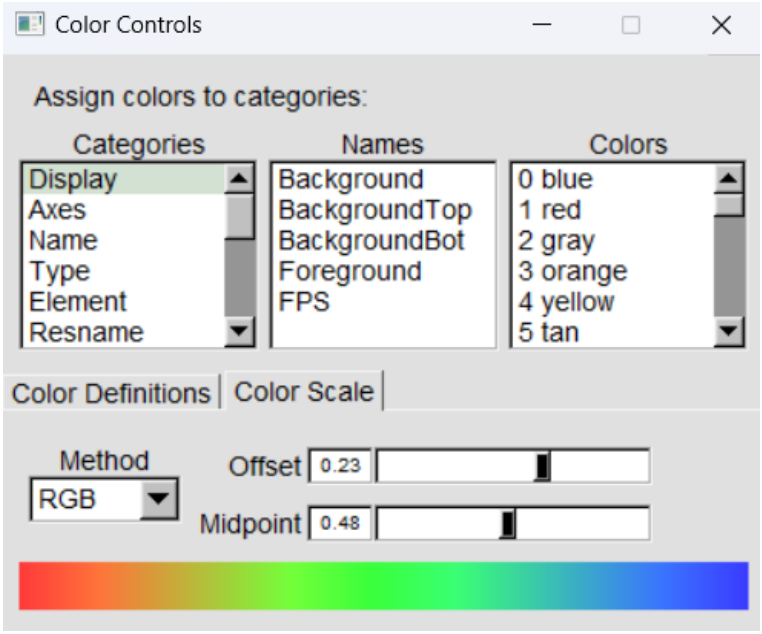
Matematicky se problém zobrazení řeší pomocí maticových transformací (používají se matice 4x4 a quaternionový formalismus, který zjednodušuje výpočty, lineární transformace jsou vhodné implementaci na grafických kartách) [13].

V programech použitých v této práci jsou důležité tři grafické pojmy, které slouží k vizualizaci proteinů, a těmi jsou Coloring (obarvování), Drawing (vykreslování) a Material (typ povrchu a textury).

Coloring

Vizualizační programy mají poměrně hodně možností, co a jak obarvit. Barva na počítači je tvořena pomocí pixelů na monitoru. Abychom viděli určité barvy na monitoru počítače je potřeba aby bylo vysíláno správné množství světla, tří základních barev (červená, zelená a modrá). Vektor těchto barev se nazývá RGB (red, green, blue). Existují i jiné reprezentace barev jako HSV (Hue, Saturation, Value), CMYK (Cyan, Magenta, Yellow, blacK) [13].

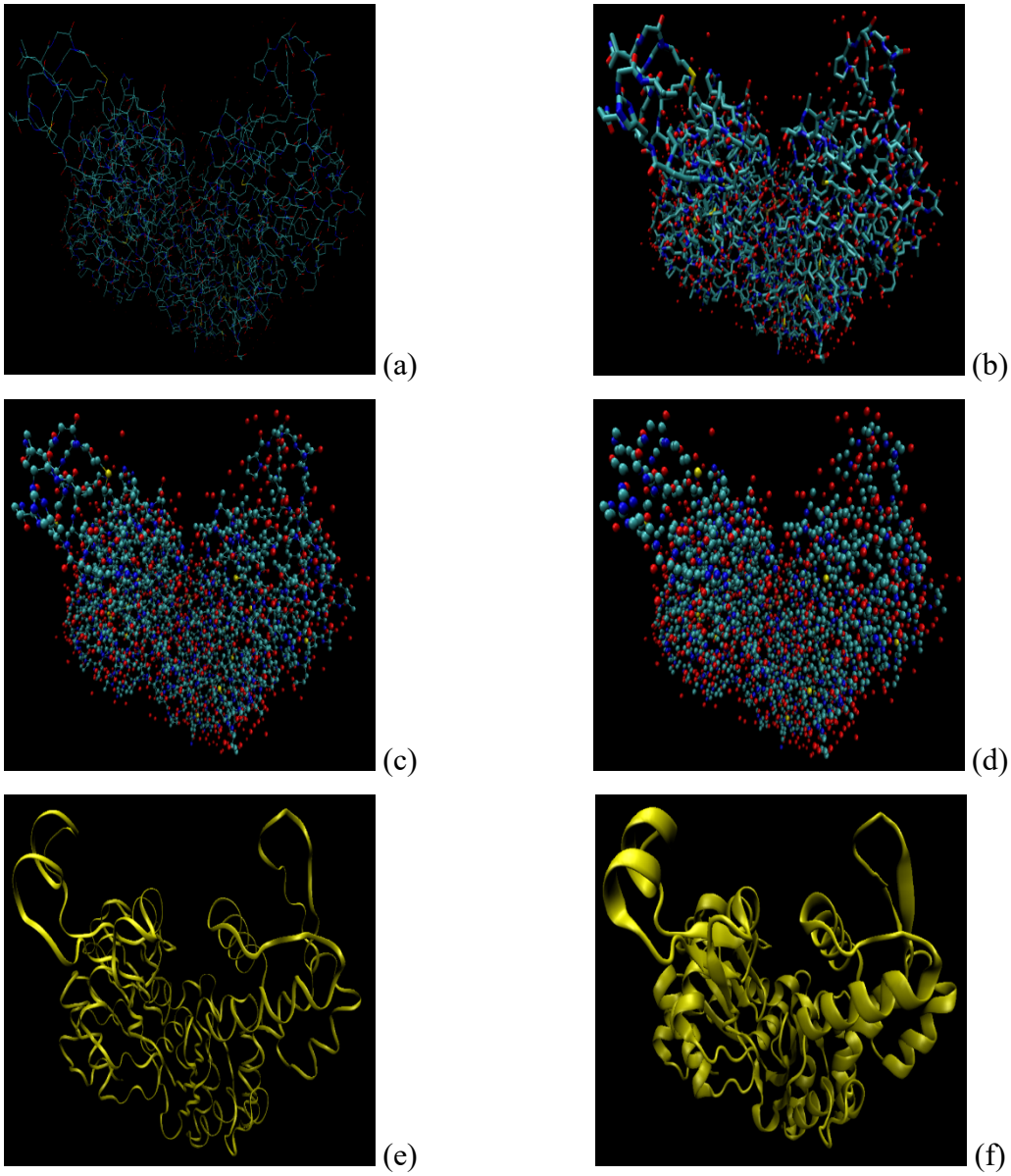
Všechny grafické zobrazovací programy mají přednastavené barvy, které můžete používat na obarvení struktur, nebo jejich částí. Většina programů má možnost navolení a nastavení barev, podle toho, co se komu líbí.



Obrázek 2: *Nastavení vlastních barev, na barevné škále. Program VMD. Kategorie, co chceme obarvovat (Categories). Přednastavené barvy (Colors).*

Drawing

Drawing neboli vykreslování slouží k výběru, jak chceme daný objekt zobrazit, jakým stylem. Různé metody zobrazují různé věci, něco zobrazuje lépe vazby a jednotlivé molekuly, a jiné zobrazení zase lépe ukáže sekundární struktury [14]. Základní vykreslovací metodou je metoda „Lines“, kdy se molekulární struktura zobrazí jako pomocí čar, které znázorňují chemické vazby mezi atomy. Samotné atomy se ale nijak nezvýrazňují. Podobně je tomu i v metodě „Licorice“ (tzv. tyčinkový model), kde jsou ovšem tyto čáry vykreslené pomocí 3D válců o dané tloušťce, na které je možné (na rozdíl od čárového modelu) namapovat vybranou texturu či povrch. Metoda „CPK“ (pojmenovaná po svých tvůrcích: Corey, Pauling, Koltun) je rozšířením tyčinkového modelu o sférické zobrazení atomů (jde o typické zobrazení molekul jako koulí spojených tyčemi reprezentujícími chemické vazby). Naopak metoda „VDW“ zobrazuje pouze atomy pomocí koulí, jejichž velikost odpovídá jejich van der Wallsovu poloměru. To je vhodné pro ilustraci vyplnění prostoru molekulární hmotou, i když u proteinů se více využívá povrchové vykreslení („Surface“). Nejčastější metody využívané pro vizualizaci proteinů jsou „Ribbons“ (popř. „Cartoons“ a jejich obdoby), které zobrazují velmi dobře prvky sekundární struktury, jako jsou šroubovice (α-helix) a skládané listy (β-sheet).



Obrázek 3: *Různé metody vykreslovací metody jako Lines (a), Licorice (b), CPK (c), VDW (d), NewRibbons (e), NewCartoons (f)*

Material

Ve všech třech programech je možné vybrat si materiál z kterého bude struktura vymodelována. Toto je důležité, protože pak si můžeme vybírat jak se bude odrážet světlo, jak bude struktura průhledná a jak se bude lesknout. Jsou dvě základní možnosti, které se nedají upravovat dále. Opaque, neboli neprůhledná a Transparent, neboli průhledný. Všechny materiály jsou definovány podle pěti nastavení: neprůhlednost (opacity), ambientní číslo které říká, jak moc odráží materiál okolní světlo, difuze (diffuse)- odrazy závislé na umístění

osvětlení a ne na směru pohledu, Specular- číslo, které popisuje intenzitu zrcadlového odrazu, lesk- číslo, které popisuje jak velké úhly svírají zrcadlové odrazy [13, 14].

3.2 Program Mol* Viewer

Tento program je interaktivní vizualizační software pro zobrazení molekulárních struktur. Software je dostupný online. Umožňuje rychle a snadno vizualizovat a analyzovat molekulární struktury v 3D prostoru. Program umožňuje zobrazení molekul v různých reprezentacích, jako jsou kuličkové modely (CPK), tyčinkové modely (licorice), plošné reprezentace nebo elektrostatické potenciály [15, 16].

Mol*Viewer je propojen s webovým prohlížečem a je na něm možné sdílet své výsledky s kolegy nebo veřejností pomocí URL adresy [16].

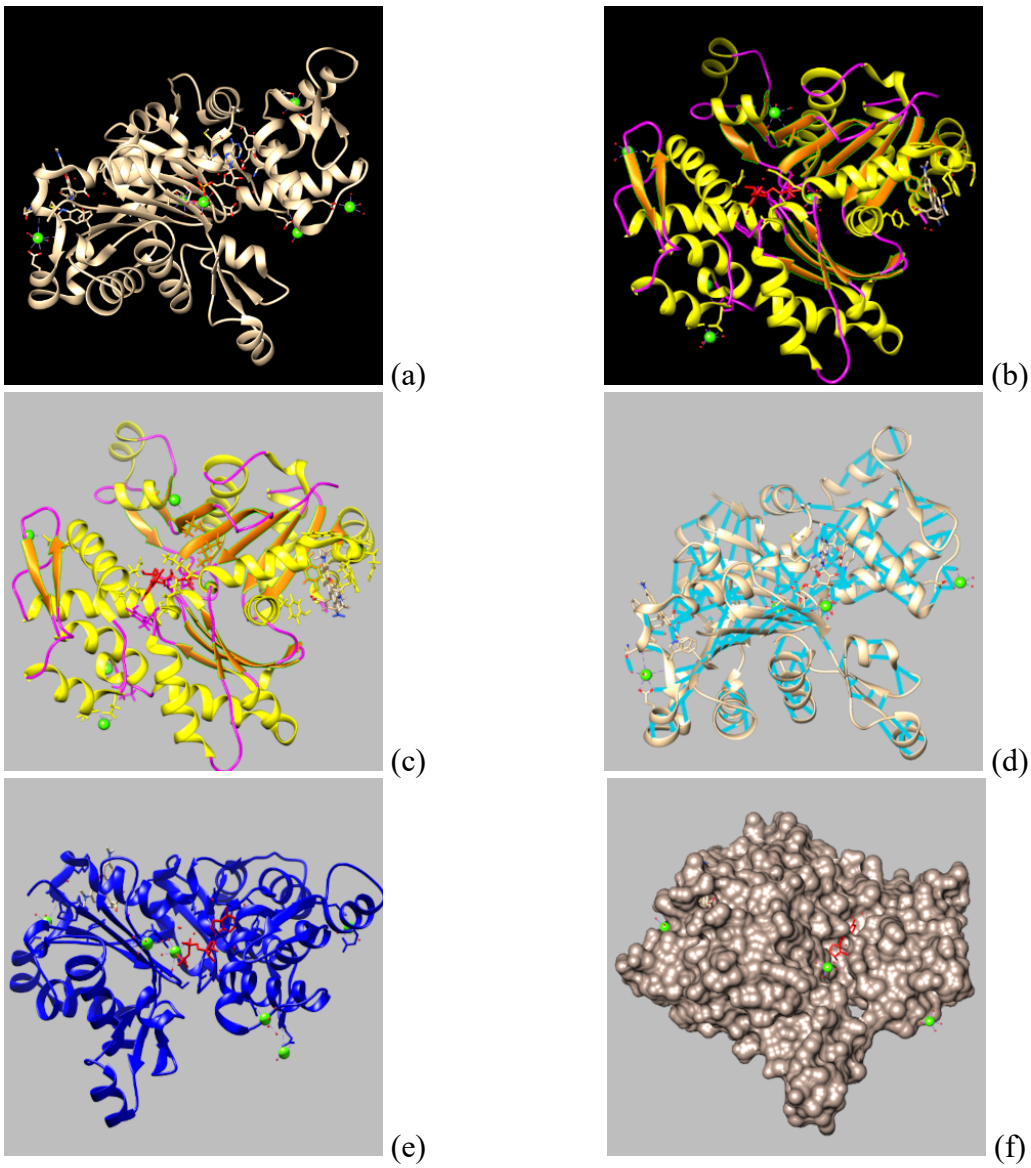
***Obrázek 4**: Uživatelské prostředí programu Mol* Viewer. Levý panel obsahuje různé nástroje pro úpravu zobrazení struktury. Horní panel nad strukturou obsahuje primární sekvenci daná struktury. Panel v pravé části obrazovky obsahuje informace o struktuře.*

3.3 Program Chimera

UCSF Chimera je program pro zobrazování a interaktivní analýzu dat molekulárních struktur, poskytuje data důležitá pro práci s molekulárními strukturami. V tomto programu je možné využít různé typy analýz. Je možné nechat si zobrazit úhly nebo délky jednotlivých vazeb. Dále je možné strukturu graficky upravovat, abychom dostali publikovatelné snímky různých struktur. Chimera byla vytvořena na University of California. Nově je možnost stažení programů ChimeraX, který má lepší výkon a dokáže pracovat s většími strukturami. Od roku 2018 není již program Chimera v aktivním vývoji. Tento program je zdarma stažitelný na webové stránce https://www.cgl.ucsf.edu/chimera/ [17, 18].

Chimera spolupracuje s PDB databází proteinových struktur.

***Obrázek 5**: Uživatelské prostředí programu Chimera. GUI tvoří jedno velké vizualizační okno, kde je zobrazena struktura (zde aktin), se kterou je možné volně otáčet. Nastavení a dostupné nástroje jsou dostupné přes menu umístněném v levém horním rohu.*

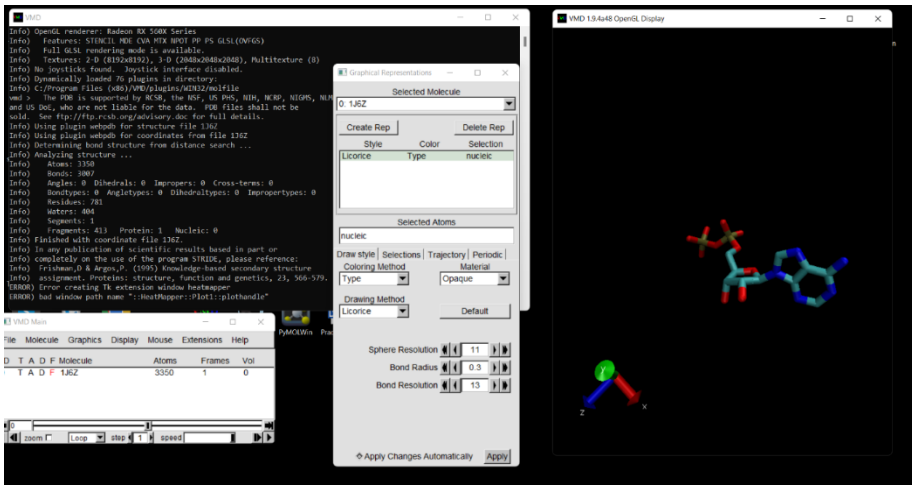


Obrázek 6: *Snímek aktinu z programu Chimera. Je zde možné vidět celý protein (zobrazený pomocí listových smyček) s navázanou nukleovou kyselinou (tyčkový model s různým obarvením atomů: červený kyslík, modrý dusík, žlutá síra). (a) Základní nastavení, (b) barevné rozlišení různých částí proteinu a organických kofaktorů (ADP), (c) barevné odlišení pozadí, (d) indikace vodíkových můstků, systém zvýraznění polohy ligandu, (f) zobrazení povrchu proteinu s patrnou pozicí ADP a iontů, toto zobrazení je dobré proto, abychom viděli tunely, kterými mohou být prováděny různé interakce s molekulou.*

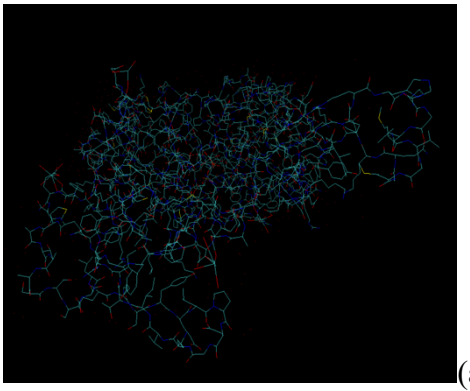
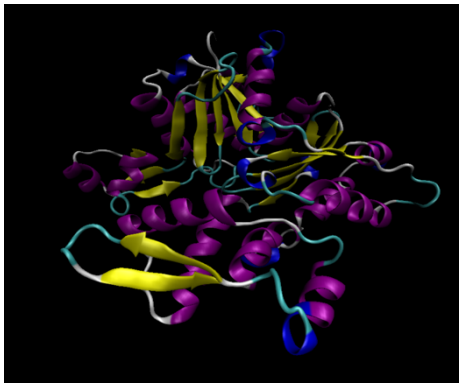
3.4 Program VMD

VMD je program pro zobrazování a následnou práci s velkými biomolekulárními systémy. Může být však použit také ke zobrazení obecnějších struktur, protože je propojen s Protein Data Bank (PDB) a dokáže číst všechny soubory v ní obsažené. VMD obsahuje pestrou škálu vykreslovacích a vybarvovacích metod, od jednoduchých čar až po metody NewCartoons. Tento program se také používá pro zobrazování a analýzu molekulárně dynamických simulací. Dokáže zobrazit simulace jak z programu AMBER, tak z programu GROMACS. Vyhotovené snímky je možné v programu rendrovat, abychom získali publikovatelné snímky. VMD podporuje své uživatele formou tutoriálů a manuálů na svých webových stránkách (odkaz níže). [14, 19]

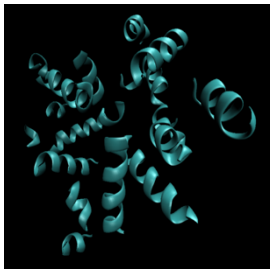
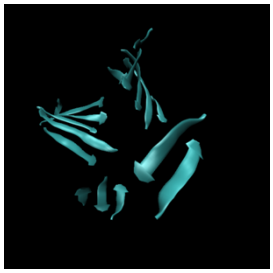
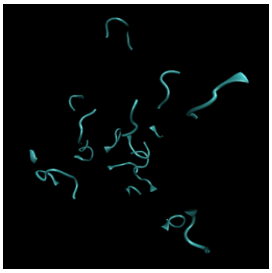
Tento program je podporován MacOS X, Unix a Windows systémy. Je zdarma stažitelný na stránkách http://www.ks.uiuc.edu/Research/vmd/ [19]. Spolupracuje s proteinovou databází PDB.



Obrázek 7: *Uživatelské prostředí VMD. V pravém horním rohu je terminálové okno, kde jsou vypsány data. V levém dolním rohu je hlavní ovládací panel. Uprostřed je okno pro specifikaci grafického zobrazení a samotná zobrazená struktura je vidět na pravé straně.*


(a)

(b)

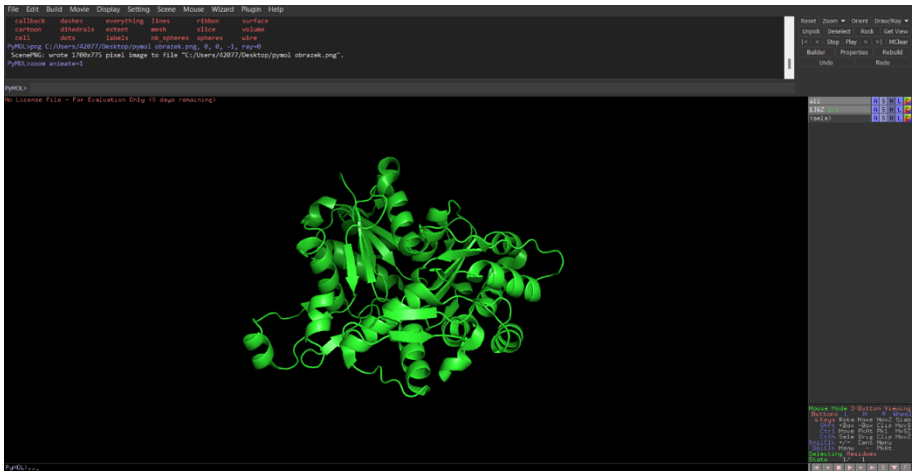
Obrázek 8: *Tento snímek je základní obrázek proteinu aktin, který je bez úprav. Je zde zobrazen celý protein. Použitá vykreslování pomocí jednoduchých čas (lines) spojující interagující atomy. Jednotlivé atomy jsou obarvené na základě chemických prvků*


(a)

(b)

(c)

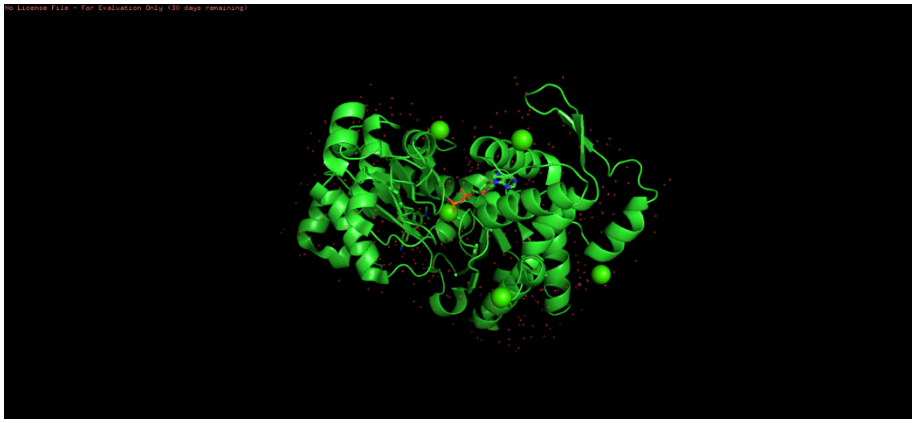
Obrázek 9: *Zde jsou byla použita metoda výběru jednotlivých struktur. Na třech snímcích jsou uvedeny jednotlivé struktury jako (a) helix, (b) beta sheet a (c) coils.*

3.5 Program PyMol

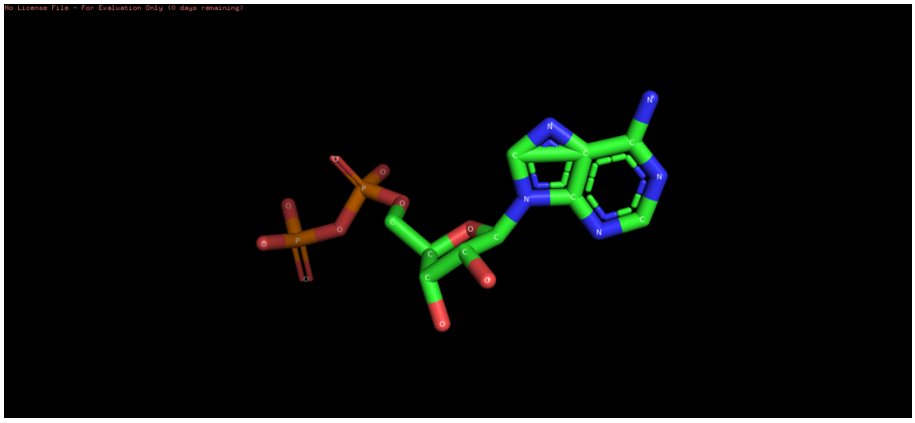
PyMOL je uživateli sponzorovaný open-source program pro vizualizaci molekulárních systémů. Vyvinula jej firma Schrödinger. V programu PyMol lze rendrovat, abychom dosáhli publikovatelných snímku. Také je možné v PyMolu spustit animace vycházející z molekulárně dynamických simulací. Je zdarma stažitelný na stránce https://pymol.org/. Spolupracuje s databází PDB protein databází [20].



Obrázek 10: *Uživatelské prostředí PyMol, v horní části obrazovky je příkazový řádek a menu. V pravé části máme ovládání obrazu samotného, jako otáčení struktury a zoom.*

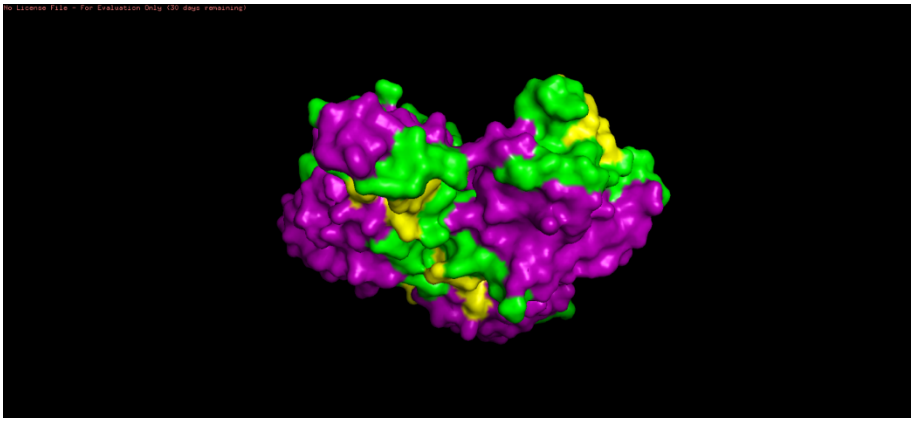


Obrázek 11: *Tento snímek je bez úprav. Je na něm zobrazen protein Aktin. Je zde vidět navázaná nukleová kyselina.*

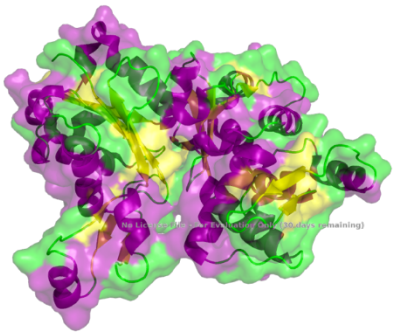


Obrázek 12: *Na tomto snímku máme ligand Adenosin difosfát. Byla použita selekce jenom daného ligandu, dále byl vybrán materiál opaque a styl licorice. Použili jsme barvení na základě elementů a přidali jsme označení jednotlivých elementů. Toto je dobré proto, aby bylo vidět z čeho se daný ligand skládá.*

Obrázek 13: *Zde jsou upraveny barvy. Fialově jsou zobrazeny šroubovice (helix). Žlutě jsou vlákna (strand). Zeleně jsou spirály (coil)*



Obrázek 14: *Na snímku číslo 14 je přidán na předchozí snímek povrch (surface). Je probarven podle struktur, které jsou pod povrchem*



Obrázek 15: *Zde je finální vyrendrovaný snímek proteinu aktin s povrchem o průhlednosti 0,5.*

4 Simulační techniky

Počítačové simulace jsou důležitou součástí studia proteinů i jiných makromolekulárních struktur (např. nukleové kyseliny, membrány, supermolekulární komplexy atd.). Simulace proteinů mají jasně daný postup, začínající v PDB databázi, kde získáme informace o daném proteinu, dále musíme protein převést do matematického modelu, abychom s ním mohli provádět simulace. Dalším krokem je pak tzv. parametrizace a následná optimalizace proteinu. Pak dochází k provádění samotných molekulárně dynamických simulací, které se pak následně analyzují a interpretují.

K provádění jednotlivých kroků zmíněných výše se používá různý vizualizační software (např. PyMol [20] či VMD [14, 19]) a specializovaný simulační software (např. Amber [21, 22] nebo Gromacs [23, 24]), jehož součástí jsou zpravidla i nástroje pro analýzu dat. Počítačové simulace a jejich přípravy se obvykle provádějí v počítačovém prostředí operačního systému UNIX/Linux. Je tomu tak proto, že většina simulačních programů je pro tento operační systém napsáno, a to jak z historických důvodů (počítačové simulace se začaly provádět ještě před vznikem operačního systému Windows), tak kvůli efektivitě (UNIX od svého vzniku podporuje multitasking i multi-user přístupy, a proto se používá na výpočetních klastrech).

Cílem počítačových simulací je získání znalostí o atomárních detailech proteinů, které nejsou jednoduší měřitelné experimentálně. Typicky se studuje strukturní stabilita a flexibilita těchto makromolekul (tj. jak moc se daný protein může měnit, zda jsou jeho podjednotky stále ve stejné poloze nebo se může přeskupovat, jak moc tyto pohyby ovlivňují aktivní místa proteinů apod.). Simulace se dále využívají ke studiu reakčních mechanismů (kolik energie je potřeba k tomu, aby se daný ligand/kofaktor dostal aktivního místa proteinu, jak dlouho to trvá, jakým způsobem je tam orientovaný a stabilizovaný, proč dochází k přerušení/vytvoření chemických vazeb, popř. přenosu náboje atd.). Kromě základního výzkumu se tedy počítačové simulace proteinů využívají také v bioinženýrství či farmakologii, kde slouží k predikci aktivity modifikovaných proteinových struktur či designu nových látek, které s proteiny interagují (léčiva, herbicidy, pesticidy apod.).

4.1 Operační systém UNIX/Linux

Operační systém Linux, který je vybudován na původním designu systému UNIX, byl vytvořen L. Torvaldsem a R. Stallmanem v r. 1991 [25]. Zatímco UNIX, tedy systém podporující multitasking a multi-user přístupy, vznikl již na konci 60. let minulého století v Bellových laboratořích a poté našel své uplatnění v akademickém prostředí, Linux měl snahu rozšířit se

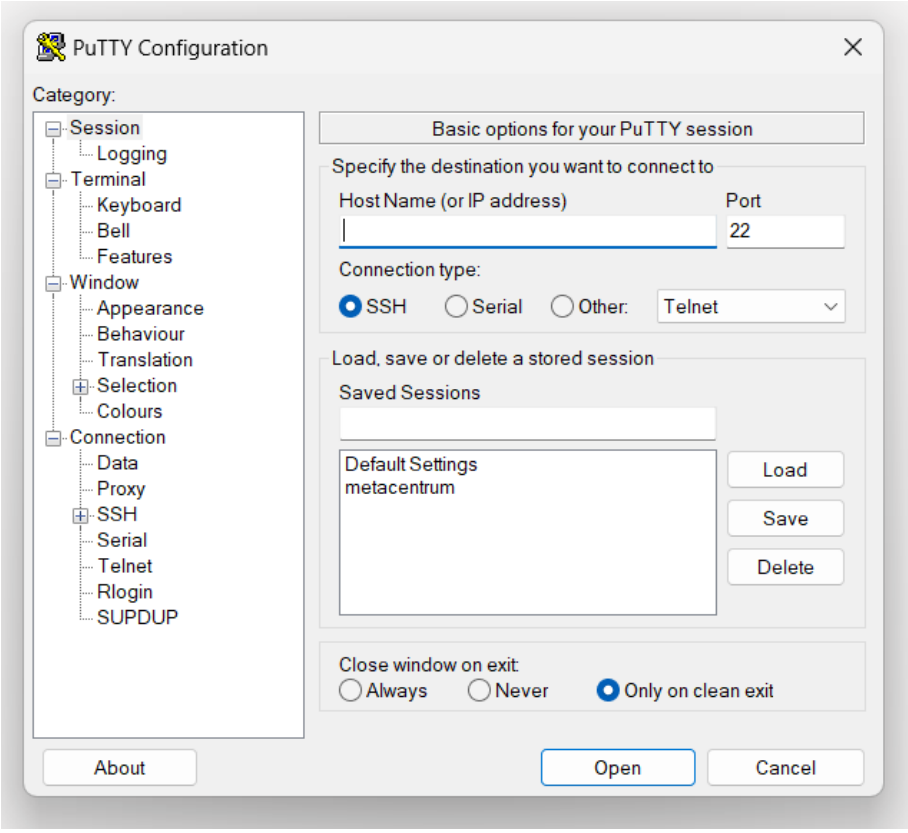
mezi běžné uživatele. Dnes jde o moderní operační systém, který je v mnohém konkurenceschopný a srovnatelný se systémy Windows a MacOS. Na rozdíl od těchto komerčních systému je ale zdrojový kód Linuxu volně dostupný a modifikovatelný (tzv. open source), což umožnilo vznik mnoha různých linuxových distribucí (např. Fedora, Debian, Ubuntu, Gentoo, atd.). Ty mají společné tzv. linuxové jádro (označované jako kernel – jde o základní infrastrukturu operačního systému přímo komunikující s hardwarem), ale liší se v uživatelské nadstavbě (grafické prostředí, systém správy apod.). Na rozdíl od Windows podporuje ale Linux i ovládání pomocí příkazové řádky v terminálovém okně, které sice může působit až trochu archaicky, umožňuje ale pokročilému uživateli velkou flexibilitu při ovládání systému (především díky možnosti skriptování a snadnému programování). Mezi běžnými uživateli je Linux známý podle svého loga – tučňáka.

Počítačové simulace (nejen) proteinů se provádějí zpravidla na počítačových klastrech (což je soustava počítačů, které jsou propojené pomocí internetových kabelů a ovládají se přes řídící uzel – tzv. frontend) nebo HPC architekturách (superpočítače designované na velké výpočty, kde je potřeba masivní paralelizace, rychlý přenos dat a často i velká operační paměť). V České republice existuje tzv. národní počítačová infrastruktura (tedy celorepubliková síť počítačových klastrů), která je známá jako virtuální organizaci MetaCentrum (MetaVO) [26]. Jednotlivé klastry jsou fyzicky umístěny na různých místech (většinou na univerzitách a vědeckých institutech jako je Akademie věd ČR), navzájem jsou propojené optickými kabely a jsou přístupně z několika frontendů. MetaCentrum VO je volně přístupné pro vědecké pracovníky, akademické zaměstnance, i univerzitní studenty. Mezi instituce zapojené do MetaCentra patří i Přírodovědecká fakulta Jihočeské univerzity. Studenti této fakulty mohou tedy získat k těmto výpočetním zdrojům pouhým vyplněním přihlášky pomocí online formuláře na webu MetaCentra.

Putty

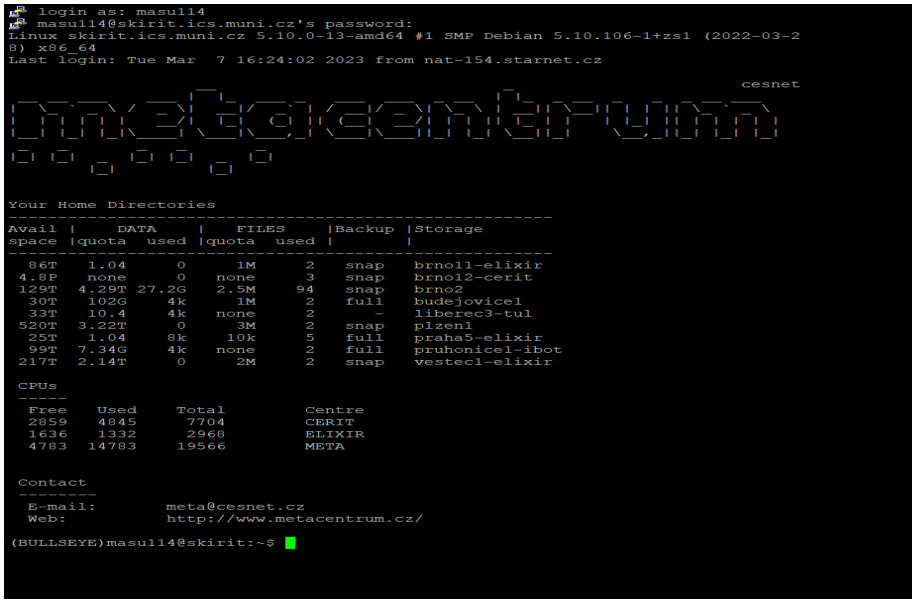
Frontendy MetaCentra jsou spravovány operačním systémem Linux a přistupuje se k nim pomocí SSH (tj. secure shell) protokolu [25]. Pracuje-li uživatel běžně na operačním systému Linux, je toto první překážkou na cestě k počítačovým simulacím, protože (1) pro SSH přístup z Windows je nutné použít specializovaný program známý jako Putty [27] a (2) na frontendu se pracuje pomocí linuxové příkazové řádky v terminálovém okně. Protože systém Windows je velmi rozšířený a v biologických oborech zcela běžný, popisujeme zde jak ovládání programu Putty, tak základní linuxové příkazy.

Putty je volně dostupný SSH klient, který zprostředkovává spojení mezi počítačem uživatele (tedy strojem, na kterém Putty spustíme) a počítačem, na kterým běží tzv. SSH server (v tomto případě frontendy MetaCentra. Putty je bezplatný, volně stažitelný a není nutné jej instalovat do systému (tj. stažený binární EXE soubor můžeme rovnou spustit). K připojení je nutné znát přihlašovací údaje na klastry MetaCentra, které uživatel obdrží po vyplnění přihlášky a jejím následném schválení [26].



Obrázek 16: *Rozhraní programu Putty, které umožňuje zadání přihlašovacích údajů a následné spojení s počítačovým klastrem pomocí SSH protokolu.*

V zásadě je nutné zadat pouze jméno serveru (tzv. Host Name, např. „skirit.ics.muni.cz“ v případě frontendu Skirit), vše ostatní je přednastaveno automaticky (např. SSH spojení přes port 22). Po stisknutí tlačítka „open“ se otevře černé terminálové okno, kam je nutné zadat uživatelské jméno a heslo (při zadávání hesla se z bezpečnostních důvodů v okně neobjevují žádné znaky). Po přihlášení se v okně objeví informace o stroji, na kterém pracujeme a tzv. „prompt“ (místo, kam se píší linuxové příkazy, je označeno zvýrazněným nebo blikajícím kurzorem).



Obrázek 17: *Terminálové okno po přihlášení na počítačový klastr Skirit, který patří do infrastruktury MetaCentra VO. Okno je zprostředkované programem Putty a ovládání probíhá pomocí příkazového řádku.*

Abychom nemuseli pokaždé zadávat uživatelské jméno, můžeme ho specifikovat už při přihlášení, jako součást Host Name. Uživatelské jméno je nutné napsat před jméno serveru a tyto dvě jméno od sebe oddělit zavináčem (např. „masu114@skirit.ics.muni.cz"). Takto zadané údaje tedy připomínají emailovou adresu. Přihlašovací údaje a jiné nastavení je v Putty také možné uložit (stačí zvolit jméno pro tyto údaje, např. „metacentrum", zadat jej do „Saved Sessions" a uložit tlačítkem „save" – při dalším spuštění je pak možné na toto jméno jen dvakrát kliknout a rovnou se otevře terminálové okno).

Pokud budeme chtít přes SSH server spouštět i nějaký grafický software (např. nějaký jednoduchý vizualizační program nebo nástroje na tvorbu a zobrazování grafů), je potřeba nastavit ještě takzvaný X-Server. Operační systémy Linux totiž používají jiné grafické knihovny než Windows. Aby bylo možné zobrazit grafický program spuštěný na Linux přes Putty ve Windows, je potřeba nejprve tyto knihovny nainstalovat a zařídit, aby se s nimi spouštěný program správně spojil. To přesně dělá X-Server, kterých existuje několik. Pro naše účely dostačuje jednoduchý X-Server známý pod jménem XMing, což je malý, volně dostupný program, který se nainstaluje do Windows a je potřeba jej spustit před spuštěním Putty (XMing běží na pozadí, při jeho spuštění se nezobrazí žádné okno). V Putty je pak potřeba ještě zaškrtnout „Enable X11 Forwarding" v Connection → SSH → X11.

Základní linuxové příkazy

V terminálovém okně, které zprostředkuje program Putty, se pracuje pomocí linuxových příkazů [26]. Ty nám umožňují provádět základní operace se souborovým systémem (tedy vytvářet nové soubory, editovat je nebo mazat, uspořádávat soubory do složek apod.) i spouštět dostupné programy (např. software pro tvorbu grafů). Samotný simulační software se ale nepouští přímo na frontendu (tedy serveru, na který se hlásíme), ale na některém z výpočetních strojů (tzv. nodů) daného klastru. Aby naše výpočty neovlivňovaly práci dalších uživatelů a naopak, zajišťuje spouštění programů na jednotlivých výpočetních strojích tzv. frontový systém, o kterém se zmiňujeme dále v této práci, při popisu samotných simulací.

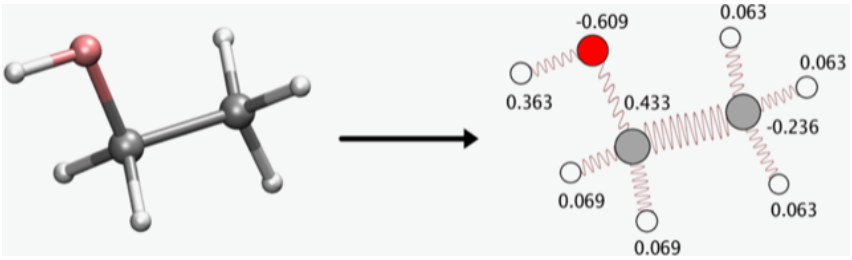
Základní linuxové příkazy, které by měl uživatel znát jsou uvedené v Tabulce 2.

Tabulka 2: *Seznam základních příkazů na ovládání klastru.*

Příkaz	Popis
clear	Vyčistí stránku
ls	Zobrazí všechny složky a soubory v dané složce
ls -l	Zobrazí všechny složky a soubory v dané složce včetně detailních údajů o jejich velikostech a přístupových právech
cd	Přesune nás do složky, jejíž jméno napíšeme za příkaz
cd ..	Přesune nás ze složky, kde pracujeme o jednu úroveň výše
exit	Ukončí připojení a tím i program Putty
rm	Vymaže složku nebo soubor
mv	Přesune složku nebo soubor
cp	Zkopíruje soubor do jiného souboru
head	Zobrazí prvních 10 řádek souboru (počet lze specifikovat)
tail	Zobrazí posledních 10 řádek souboru (počet lze specifikovat)
more	Zobrazí obsah souboru stránku po stránce
nano	Jednoduchý textový editor pro psaní a úpravu souborů
vi	Pokročilý textový editor pro psaní a úpravu souborů
mkdir	Vytvoří soubor nebo složku
pwd	Vytiskne jmeno adresáře, ve kterém se nacházíme
cat	Vypíše celý obsah souboru (dá se použít ke spojování souborů)
grep	Prohledávání textových souborů
history	Zobrazí historii použitých příkazů

4.2 Molekulárně-mechanický popis

Chceme-li studovat vlastnosti molekulárních systémů pomocí počítačových simulací, je potřeba je reprezentovat vhodným fyzikálním modelem. Protože velikosti molekul jsou mikroskopické (typické rozměry středně velkých proteinů jsou 5-10 nm), měli bychom jejich struktury a vlastnosti popisovat pomocí kvantové teorie [28]. Takový přístup by nám sice v principu umožnil studovat např. chemické změny, přenosy náboje či optická spektra, ale kvantový popis je velmi komplikovaný a v praxi aplikovatelný pouze na relativně malé molekuly (do 100-1000 atomů). Pokud se zajímáme pouze o strukturní nebo dynamické vlastnosti, při kterých nedochází ke změnám topologie (tj. nevznikají ani nezanikají žádné vazby), můžeme se omezit na klasické přiblížení, ve kterém molekuly modelují jako systémy nabitých koulí navzájem spojených pružinami. Tento tzv. pružinový model, schématický zobrazený na obrázku 18, je základem molekulárně mechanického (MM) popisu molekulárních systémů [28].



Obrázek 18: *Aplikace molekulárně mechanického popisu na molekulu etanolu. Jednotlivé atomy (stříbrný uhlík, červený kyslík, bílý vodík) se popisují jako nabité koule navzájem spojené mechanickými pružinami. Parciální náboje v atomových jednotkách jsou specifikovány u jednotlivých atomů.*

Popis chemických vazeb pomocí mechanických pružin místo tuhých tyčí zajišťuje, že tyto vazby mohou vibrovat, a to s frekvencí, která je dána tuhostí pružin a hmotnostmi jednotlivých atomů. Aby si molekula zachovávala požadovanou strukturu, je potřeba podobným způsobem kontrolovat také velikosti valenčních a torzních úhlů navzájem svázaných atomů (což, pro jednoduchost, není zahrnuto v obrázku 18). Pružinový model tak umožňuje studovat strukturní změny bez chemických modifikací (např. konformační změny proteinů či protein folding), mezimolekulární interakce (např. nekovalentní vázání ligandů do aktivních míst enzymů, proteinové agregace nebo adsorpce proteinů na různé povrchy) a molekulární pohyb (např.

difuze iontů okolo proteinů či uvnitř proteinových kanálů, penetrace membránových struktur, apod.) [28].

Potenciální funkce

Fyzikální model nám slouží k vytvoření matematického popisu molekulárního systému, jehož základem je tzv. potenciální funkce, což je v podstatě závislost potenciální energie systému na jeho geometrii a dalších vstupních parametrech (např. atomové náboje nebo tuhosti pružin) [28]. Tuto funkci pak můžeme využít k hledání optimální struktury (tzv. geometrická optimalizace) nebo k simulacím časového vývoje systému (tzv. molekulární dynamika), které jsou popsání níže. Jde tedy o ústřední fyzikální veličinu v počítačových simulacích.

Potenciální energie v molekulárně mechanickém popisu, E_{MM} se dá rozdělit na vazebné a nevazebné příspěvky, které jsou navzájem nezávislé [28-30]

$$E_{MM} = E_b + E_a + E_d + E_{vdW} + E_{Cl} \quad (1)$$

Vazebné příspěvky zahrnují energie vazeb (E_b), energie valenčních úhlů (E_a) a energie torzních (tzv. dihedrálních) úhlů (E_d). Tyto příspěvky jsou čistě intra-molekulární, což znamená, že nezávisí na vzájemné poloze různých molekul. Nevazebné příspěvky se dělí na krátkodosahové van der Waalsovské interakce (E_{vdW}) a dalekodosahové elektrostatické Coulombovské interakce (E_{Cl}). Tyto členy mohou mít jak intra-molekulární tak inter-molekulární příspěvky.

Vazebné interakce

V pružinovém molekulárně mechanickém modelu mají energie vazeb a úhlů tvar harmonických potenciálů

$$E_b = \frac{1}{2}\sum_{\text{vazby}} k_b(r - r_0)^2, \quad (2)$$

$$E_a = \frac{1}{2}\sum_{\text{valenční úhly}} k_a(\alpha - \alpha_0)^2, \quad (3)$$

kde r_0 a α_0 jsou rovnovážné hodnoty vazebných délek a valenčních úhlů a k_b, k_a silové konstanty ovlivňující flexibilitu vazeb a úhlů (tyto konstanty mají význam tuhostí pružin v mechanickém modelu). Harmonický potenciál ale není aplikovatelný na torzní úhly, které popisují rotace jednotlivých skupin, a měly by být tedy popsané nějakou periodickou funkcí. Nejčastěji se volí potenciál ve tvaru

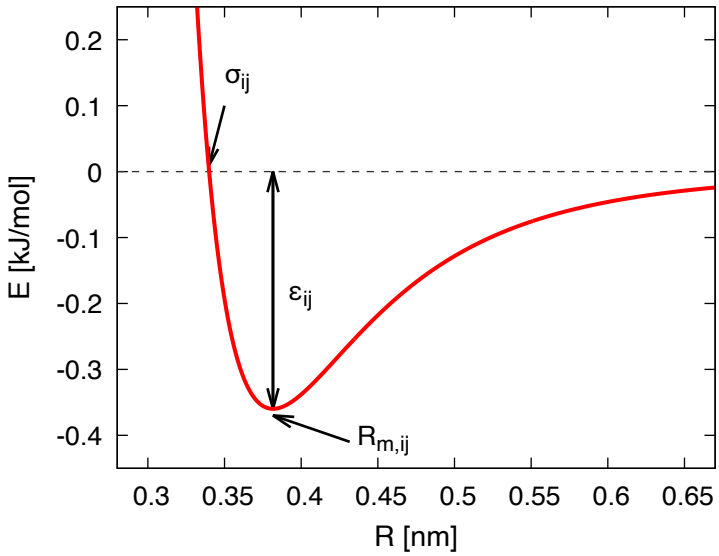
$$E_d = \frac{1}{2}\sum_{\text{torzní úhly}} k_d[1 + \cos(n\vartheta + \vartheta_0)], \quad (4)$$

což jsou první dva členy Fourierovy řady, která se používá pro rozvoj periodických funkcí. V případech, kdy je potřeba popsat složitější tvar torzního potenciálu, je možné řadu rozšířit o další členy. Tyto korekce ale zpomalují MM výpočty, a proto se v praxi moc nepoužívají.

Nevazebné interakce

Van der Waalsovy interakce popisují disperzní působení mezi jednotlivými atomy, které klesá s šestou mocninou vzdálenosti. Tyto interakce jsou tedy krátkodosahové, což znamená, že se uplatňují jen v nejbližším okolí (do ~1 nm) jednotlivých atomů. Pokud se k sobě atomy ale přiblíží na velmi krátkou vzdálenost, kdy začne docházet k překryvu jejich elektronových obalů, začnou se tyto atomy silně odpuzovat [28]. Tento efekt je zohledněn v tzv. Lennard-Jonesově potenciálu, který obsahuje repulzní člen rychle klesající s dvanáctou mocninou meziatomové vzdálenosti R

$$E_{Cl} = \sum_{i<j}\left[\frac{A_{ij}}{R_{ij}^{12}} - \frac{B_{ij}}{R_{ij}^{6}}\right] \quad (6)$$



Obrázek 19: *Lennard-Jonesův potenciál pro interakci aromatických uhlíků (atomový typ CA v silových polích Amber, = 0.38 nm, = 0.34 nm, = 0.36 kJ/mol).*

Průběh LJ potenciálu je zobrazen na obrázku 19. Pozice minima ($R_{m,ij}$) a jeho hloubka (ε_{ij}), které odpovídají interakční vzdálenosti a energii u daného páru atomů *i* a *j*, je dána hodnotami konstant A_{ij} a B_{ij}. V praxi se neurčují tyto konstanty pro každý pár, ale udávají se atomové parametry ε_i a $R_{m,i}$ (popř. σ_i) pro různé atomové typy. Potřebné párové konstanty se pak z atomových parametrů spočtou pomocí tzv. mixovacích pravidel. Např. v potenciálech typu Amber se používají Lorentz Berthelotovy pravidla [29, 30]:

$$R_{m,ij} = \frac{1}{2}\left(R_{m,i} + R_{m,j}\right) \tag{6}$$

$$\varepsilon_{ij} = \sqrt{\varepsilon_i \varepsilon_j} \tag{7}$$

Tyto parametry souvisí s konstantami A_{ij} a B_{ij} v rovnici (6) vztahy

$$A_{ij} = \varepsilon_{ij} R_{ij}^{12} \tag{8}$$

$$B_{ij} = 2\varepsilon_{ij} R_{ij}^{6} \tag{9}$$

Druhým typem nevazebného působení jsou elektrostatické interakce mezi bodovými náboji q_i, které nesou jednotlivé atomy. Tato interakce se řídí klasickým Coulombovým zákonem

$$E_{Cl} = \frac{1}{4\pi\varepsilon_0} \sum_{i<j} \frac{q_i q_j}{R_{ij}} \tag{8}$$

kde R_{ij} je vzdálenost mezi atomy *i*, *j* a konstanta ε_0 má význam permitivity vakua. V praxi se často v MM simulacích využívají tzv. periodické okrajové podmínky (PBC), které umožňují virtuální rozšíření studovaného systému o jeho periodické obrazy. V těchto případech by ale vyčíslení elektrostatické energie přímou sumací (8) bylo velmi neefektivní, a proto se používá tzv. Ewaldova sumace a její modifikace (především „smooth particle mesh Ewald“ algoritmus známý jako PME). V této metoda se, zjednodušeně řečeno, přímou sumací vyčíslí jen interakce s nejbližším okolí jednotlivých atomů (až po specifikovanou vzdálenost označovanou jako „cutoff“) a interakce se vzdálenějšími atomy se vysčítá v tzv. reciprokém prostoru.

Pro určení potenciální energie molekulárního systému potřebuje znát nejen jeho strukturu (tedy souřadnice jednotlivých atomů $\vec{r}_i$, z kterých lze dopočítat vazebné délky r a hodnoty valenčních úhlů α a torzních úhlů ϑ), ale i všechny parametry jednotlivých potenciálů (rovnovážné hodnoty vazebných délek r_0 a úhlů α_0, ϑ_0, příslušné silové konstanty k_a, k_b, k_d, periodicity torzních potenciálů n, LJ atomové parametry $R_{m,i}$, ε_i a náboje q_i. Soubor těchto konstant, společně s hmotnosti jednotlivých atomů m_i, se nazývá silové pole. Pro popis biomolekul se nejčastěji používají silové pole typu Amber [21], CHARMM [31], OPLS [32]

a GROMOS [33], které se liší v detailech potenciálové funkce i způsobu, jakým jsou v nich určovány hodnoty jednotlivých parametrů. V této práci používáme výhradně silové pole typu Amber.

4.3 Simulační software

Simulační techniky založené na molekulárně mechanickém popisu (tj. na parametrizace molekul pomocí silových polí), jsou implementované v několika programových balících. Mezi nejznámější programy, které se využívají pro studium proteinů, patří Amber [21, 22], CHARMM [31], Gromacs [23, 24], GROMOS [34], NAMD [35] a Yasara [36]. V této práci se porovnáváme použití programu Amber s programe Gromacs, které jsou v dané komunitě nejběžnější.

Amber

Amber (Assisted Model Building with Energy Refinement) [21, 22] je software určený pro simulace biomolekul jako jsou nukleové kyseliny, proteiny a sacharidy. Kromě molekulárně mechanických simulací umožňuje program provádět také semi-empirické kvantové výpočty a hybridní molekulárně mechanické/kvantově mechanické (QM/MM) výpočty [28], které jsou často využívány ke studiu biomolekul a jejich reakcí. Program Amber byl vytvořen v roce 1981 na University of California v San Francisku a od té doby je neustále aktualizován a vylepšován. Nejnovější verze Amber21 z roku 2021 je optimalizovaná pro masivní paralelizaci (OpenMP a MPI) i akceleraci na GPU. Kromě standardních molekulárně dynamických simulací program umožňuje výpočty volných energií pomocí různých metod (termodynamická integrace, umbrella sampling, replica exchange, steered MD, string methods), simulace ve specifických prostředích a podmínkách (konstantní pH, konstantní redoxní potenciál) a refinement dat z rentgenové difrakce a NMR měření.

Programový balík Amber [21, 22] je z části komerční (použití pokročilých metod a efektivních optimalizovaných modulů vyžaduje koupenou licenci), ale má i svou volně dostupnou část (dříve distribuována jako AmberTools). Volná část obsahuje programy, pomocí kterých je možné parametrizovat standardní i atypické biomolekulární systémy a vytvořit tak tzv. topologii (soubor obsahující informace o vazbách mezi jednotlivými atomy a všechny parametry silového pole, potřebné k vyčíslení potenciálové funkce). Tyto programy jsou velmi efektivní a často se používají i k parametrizaci nejrůznějších organických molekul, které jsou

pak simulovány jinými softwarovými balíky (např. Gromacs [23, 24]). Volně je možné provádět i nenáročné simulace pomocí modulu známého jako Sander.

V této práci využíváme verzi Amber14, která sice není nejnovější, ale pro tuto práci zcela dostačující. Důvodem použití této verze je dostupnost její plné licence na MetaCentrum VO, což nám umožní využít GPU akcelerovaný modul PMEMD. Díky tomu můžeme simulovat dostatečně dlouhou trajektorii pro následné analýzy i porovnat efektivnost programu Amber s programem Gromacs.

Gromacs

Podobně jako Amber, je i Gromacs (GROninge Machine for Chemical Simulations) softwarový balík určený pro simulace biomolekul [23, 24], jako jsou lipidy, proteiny a sacharidy. Program byl vytvořen na University of Groningen v Nizozemsku v roce 1995 a od té doby se stále vyvíjí a aktualizuje (poslední verze byla zveřejněna na začátku roku 2023). Gromacs také podporuje hybridní QM/MM simulace, výpočty volných energií, simulace různých prostředí a podmínek [28]. Stejně jako Amber je i Gromacs velmi dobře paralelizován a podporuje akceleraci na GPU. Na rozdíl od Amberu je ale Gromacs volně přístupný a modifikovatelný (tzv. open-source), a proto je jeho rozšíření ve vědecké komunitě o větší [24].

Gromacs používá jiné datové formáty pro ukládání struktur, topologií a trajektorií než Amber, což je ale typické i pro ostatní simulační balíky. Parametrizace nestandardních systémů (proteiny s kovalentně vázanými kofaktory, specifické anorganické části – např. FeS klastry, apod.) je o poznání složitější Gromacsu než v Amberu, a proto se často použití těchto programů kombinuje, tj. systém se parametrizuje pomocí volných nástrojů balíku Amber, výsledná topologie se pak konvertuje do formátu, který podporuje Gromacs, a ten se použije k samotným simulacím. V této práci využíváme k simulacím verzi 2020.3, která je k dispozici na MetaCentru VO [26].

4.4 Příprava struktury a stavba modelu

Nejprve si musíme stáhnout soubor s daty, který najdeme na stránce daného proteinu (kapitola 2.2, obrázek 1). Poté za pomocí příkazu grep vystřihneme řádky se slovem ATOM na začátku. Tyto řádky obsahují data o jednotlivých atomech, o jejich pozici. Posledním krokem je parametrizace, přidání molekul vody a ionty. Ionty jsou přidány, aby byl vyrovnán náboj na

nulový. Na konci tohoto procesu dostaneme dva soubory. Jeden s topologiemi a druhý s koordináty.

Příprava struktury v programu Amber

V softwarovém balíku Amber se parametrizace provádí pomocí programu TLeap [29]. Program komunikuje s uživatelem buď interaktivně anebo zpracovává poskytnutý skript s příkazy. Pomocí TLeap je možné nejen parametrizovat daný protein (tj. vytvořit seznam konektivit a přiřadit jednotlivým atomů, vazbám a úhlům příslušné parametry ze silového pole), ale také ho obklopit vodou a neutralizovat jeho náboj pomocí iontů. To je ukázáno v následujícím skriptu:

```
source leaprc.ff03.r1
```
Načtení parametrů silového pole FF03.R1

```
x = loadpdb actin.pdb
```
Načtení struktury proteinu z upraveného PDB souboru (struktura se pro další použití označí jako „x“)

```
solvatebox x TIP3BOX 10.0
```
Vytvoření simulačního boxu okolo proteinu (vzdálenost 10 A mezi proteinem a krajem boxu na každé straně). Volné místo v tomto boxu se zaplní molekulami vody (model TIP3P).

```
addions x Na+ 0.0
```
Neutralizace celkového náboje přidáním iontů sodíku

```
saveamberparm x actin.top actin.crd
```
Uložení výsledné topologie a souřadnic do souborů „*actin.top*“ a „*actin.crd*“

```
quit
```
Konec

Příprava struktury v programu Gromacs

V programu Gromacs se k parametrizaci používá modul „pdb2gmx,“ který se spouští následujícím způsobem [30]:

```
gmx pdb2gmx -f data.pdb -o data.gro -p data.top -i data.itp -n data.ndx -
water tip3p
```

Struktura vstupního proteinu se čte ze souboru „*data.pdb*“ a finální topologie a souřadnice se uloží do souborů „*data.top*“ a „*data.gro*.“ V souboru „*data.itp*“ jsou uložené další údaje, které se vkládají do topologie (v tomto případě silové konstanty, které mohou sloužit k omezení

pohybu těžkých atomů proteinu) a soubor „data.ndx“ (tzv. index file) obsahuje ID atomů z jednotlivých částí proteinu (např. proteinová kostra, postranní řetězce, aromatické aminokyseliny, apod.).

Následně je potřeba obklopit protein vodou. K tomu slouží modul „solvate“:

```
gmx solvate -cp data.gro -p data.top -o solution.gro -shell 1.0
```

Tento program vytvoří box okolo proteinu (do vzdálenosti 1 nm) a zaplní ho vodou. Pak je potřeba přidat ještě ionty, které budou kompenzovat náboj proteinu. Toto se provede pomocí modulu „genion.“ Tento příkaz nahradí náhodně molekuly rozpouštědla molekulami požadovanými ionty.

```
gmx genion -s data.tpr -n data.ndx -p data.top -np 11 -pname NA
```

Modul „genion“ načítá stukturní data z TPR souboru „data.tpr“. Tento soubor se vytváří modulem „grompp“ a postup, jak se to dělá, je uveden v následující části. Počet kladně nabitých iontů se specifikuje volbou "np" (v tomto případě 11, protože aktin má vodném prostředí náboj -11 *e*) a typ iontu se udává volbou „pname“ (NA je jméno pro sodík, používané v Gromacsu).

4.5 Strukturní optimalizace

Strukturní nebo též geometrická optimalizace či energetická minimalizace, je výpočetní technika používaná k nalezení stabilní struktury molekulárních i jiných systémů [28]. Je založena na poznatku, že stabilním strukturám odpovídá minimum potenciální funkce (jakékoliv narušení takovéto struktury může potenciální energii pouze zvýšit). Jak je známo z matematické analýzy, funkce mají v extrémech nulovou derivaci, což platí i pro funkci potenciální energie. Tato funkce je ale mnohodimenzionální (u systému složeného z N částic je to funkce 3N proměnných), a proto musí být v minimu nulové všechny derivace podle těchto složek

$$\vec{g} = \nabla E\left(\vec{R}_1, \vec{R}_2, \dots \vec{R}_N\right) = \left(\frac{\partial E}{\partial x_1}, \frac{\partial E}{\partial y_1}, \frac{\partial E}{\partial z_1}, \frac{\partial E}{\partial x_2}, \frac{\partial E}{\partial y_2}, \frac{\partial E}{\partial z_2}, \dots \frac{\partial E}{\partial x_N}, \frac{\partial E}{\partial y_N}, \frac{\partial E}{\partial z_N}\right) = 0 \qquad (9)$$

Vektor $\vec{g}$ se nazývá gradient a je orientován do směru největšího nárůstu funkce E v daném bodě. Naopak síly, které působí na jednotlivé částice (atomy) směřují vždy k minimu potenciální energie

$$\vec{F} = -\vec{g} = -\nabla E \tag{10}$$

Metody geometrické optimalizace hledají takové rozložení atomů, ve kterém jsou jednotlivé atomové síly nulové a ke kterému síly směřují, pokud jsou atomy v prostoru rozloženy jinak.

Hledání optimální struktury je tedy v podstatě matematický problém, k jehož řešení se používají různé metody a počítačové algoritmy. Na začátku je potřeba mít rozumný odhad toho, jak by měla tato struktura vypadat (tzv. „initial guess“). Tato počáteční geometrie se pak iterativně v jednotlivých krocích zpřesňuje. Pokud bychom označily souřadnice všech atomů v n-tém kroku jako $\vec{X}_n = (\vec{R}_1, \vec{R}_2, \ldots \vec{R}_N)$, pak následující strukturu $\vec{X}_{n+1}$, která by měla mít menší potenciální energii, dostaneme korekcí

$$\vec{X}_{n+1} = \vec{X}_n + \alpha_n \vec{d}_n \tag{11}$$

kde $\vec{d}$ je směrový vektor a $\vec{d}$ je škálovací faktor, někdy též označovaný jako délka kroku. Jednotlivé optimalizační metody se pak liší tím, jakým způsobem tento směrový vektor a škálovací faktor určují.

Nejjednodušší možností je využít poznatku, že atomové síly směřují vždy ve směru „největší srázu“ potenciální funkce (tedy tam, kde dochází nejrapidnějšímu snížení energie). Silový vektor (10) můžeme tedy přímo použít jako směrový vektor $\vec{d}$ v (11). Na tom je založená tzv. metoda největšího poklesu známá jako „steepest descent“ (SD) [29, 30]

$$\vec{X}_{n+1}^{\mathrm{SD}} = \vec{X}_n - \alpha_n \vec{g}_n \tag{12}$$

Tato metoda je robustní (v každém kroku dojde ke snížení energie), ale neefektivní, protože v blízkosti minima, kde jsou síly (gradienty) velmi malé, jsou malé i optimalizační kroky (12) a k nalezení optimální struktury je typicky potřeba velké množství těchto kroků.

Efektivnější optimalizační metodou je algoritmus nazývaný „conjugate gradient“ (CG) [29, 30], který vychází z SD, ale snaží se korigovat směrový vektor tak, aby k nalezení minima došlo v menším počtu kroků. Tato korekce je založena na znalosti směrového vektoru z předchozího optimalizačního kroku

$$\vec{X}_{n+1}^{\mathrm{CG}} = \vec{X}_n - \alpha_n \left[\vec{g}_n - \beta_n \vec{d}_{n-1}\right] \tag{13}$$

Existuje více možností, jak nastavit tzv. konjugační faktor β, všechny jsou ale založené na znalosti silového vektoru (resp. gradientu) na dané a předchozí struktuře.

Pokročilejší metody, jako např. Newton-Raphsonův algoritmus a jeho různé aproximace (např. BFGS nebo DIIS), které určují směrový vektor $\vec{d}$ z druhých derivací potenciální energie nebo jejich odhadu, jsou výpočetně náročnější, ale ještě o něco efektivnější než CG [28-30]. Škálový faktor α, který vystupuje ve všech optimalizačních algoritmech, je potřeba v každém kroku nastavit tak, aby nová struktura měla nejmenší možnou energii ve směru $\vec{d}$. Z matematického pohledu jde tedy opět o problém minimalizace, ale pouze v jednom směru. Metody, které toto řeší jsou známe jako „line search" algoritmy a jejich volba ovlivňuje efektivity i robustnost optimalizačních metod popsaných výše. V praxi se nejčastěji používá odhad hodnoty α ze znalosti energie a gradientu na dané struktuře (tzv. metoda sečen).

V simulacích proteinů a jiných velkých molekulárních systémů se strukturní optimalizace nevyužívá k nalezení minima potenciální energie, ale pouze jako přípravná fáze před molekulárně dynamickými (MD) simulacemi [29, 30]. Experimentální struktury totiž nemusí odpovídat simulovaným podmínkám, části peptidických řetězců mohou být špatně rozlišené a soubory zpravidla neobsahují informace o pozicích vodíkových atomů (to se týká především strukturních dat z rentgenové difrakce a CryoEM [7, 8]). Tyto informace se před samotnými simulacemi do struktury dodávají (chybějící aminokyseliny či celé části řetězců je možné doplnit homologním modelováním, atomy vodíků se doplňují do standardních poloh na základě chemických poznatků o struktuře aminokyselin), a pro jejich zpřesnění se právě využívá strukturní optimalizace. Po sestavení modelu se zpravidla aplikuje několik optimalizačních kroků (~1000, podle velikosti a složitosti systému) metody SD, což většinou stačí k tomu, aby se odstranily případné strukturní problémy (např. blízké kontakty některých atomů). Pokud je potřeba počáteční atomové síly dále zmenšit, aby se zamezilo případné počáteční nestabilitě následujících MD simulací, aplikuje se po prvotní SD optimalizace ještě CG.

Existují různé nástroje pro molekulární dynamické simulace a strukturní optimalizace, včetně programů jako jsou Amber, Gromacs. Tyto programy mají různé metody a algoritmy pro simulace a optimalizaci, a také různé způsoby popisu molekul, jako jsou například síly polí, ab initio metody a kvantově-mechanické simulace [29, 30].

Optimalizace v programu Amber

Aby program věděl, jaký typ simulace má provést s jakými parametry, je potřeba mu poskytnout vstupní soubor (input) s popisem požadovaného nastavení. Zde je ukázka vstupního souboru (dále označovaného jako „*optimalizace.inp*“ s popisem SD optimalizace:

```
&cntrl
```
Začátek tzv. kontrolní sekce (tento příkazový blok je ukončen lomítkem na začátku řádku)

```
 imin=1,
```
Geometrická optimalizace (energetická minimalizace)

```
 maxcyc=1000,
```
Maximální počet cyklů minimalizace

```
 ntx=1,
```
Specifikace dat ve vstupním souboru (očekávají se pouze souřadnice atomů a informace o velikosti simulačního boxu)

```
 irest=0,
```
Nová simulace (nejde o restart z předchozího běhu)

```
 cut=10,
```
Tzv. cutoff pro elektrostatickou interakci (v Angströmech)

```
 ntxo=1,
```
Formát ukládaných dat (v tomto případě ASCII, tedy nekomprimovaný textový formát)

```
 ntpr=100,
```
Frekvence zápisu energií v průběhu minimalizace (každých 100 kroků)

```
 ntb=1,
```
Okrajové podmínky (1 indikuje periodické okrajové podmínky s neměnným boxem)

```
/
```
Konec kontrolní sekce

Tento soubor slouží jako vstup pro modul Sander (součást volné části balíku Amber) i pro modul PMEMD (optimalizovaná paralelizace a GPU akcelerace) [29]. K samotnému spuštění simulace je nutné ještě napsat skript pro frontový systém, kde se specifikuje na kolika procesorech (popř. grafických kartách) má výpočet běžet, jaká bude maximální doba běhu a kolik k tomu bude program potřebovat operační a diskové paměti. Na MetaCentru VO v současné době stále běží frontový systém PBS (v blízké budoucnosti je plánován přechod na systém SLURM) a skript pro spuštění vypadá takto:

#!/bin/bash
Začátek skriptu (specifikace shellu pro interpretaci linuxových příkazů)

#PBS -l select=1:ncpus=4:mem=500MB:scratch_local=500MB
Specifikace pro PBS: specifikace počtu výpočetních strojů (1), procesorů (4), operační a diskové paměti (obě 500 MB)

#PBS -l walltime=01:00:00
Specifikace pro PBS: zde určujeme, kolik času si chceme rezervovat na daném stroji (po vypršení tohoto času bude úloha ukončena)

#PBS -N actin
Specifikace pro PBS: jméno úlohy (bude používáno při zobrazování údajů o běhu)

module load amber-14
Zpřístupnění programu Amber14

cd $PBS_O_WORKDIR/
Přechod do naší domovské složky

cp optimalizace.inp $SCRATCHDIR/
Zkopírování souboru optimalizace.inp do pracovního adresáře na výpočetním stroji

cp actin.crd $SCRATCHDIR/
Zkopírování souboru se souřadnicemi daného proteinu

cp actin.top $SCRATCHDIR/
Zkopírování souboru s topologií daného proteinu

cd $SCRATCHDIR/
Přechod do pracovního adresáře, kde bude probíhat výpočet (tzv. scratch)

mpirun sander.MPI -O -i optimalizace.inp -o optimalizace.out -r optimalizace.rst -c actin.crd -p actin.top
Samotný příkaz pro výpočet, volba -i specifikuje vstupní soubor, -o výstupní soubor, -r restart soubor, -c souřadnice, -p topologii

cp optimalizace.out $PBS_O_WORKDIR/
Zkopírování výstupního souboru s daty z výpočtu do naší domovské složky

cp optimalizace.rst $PBS_O_WORKDIR/
Zkopírování restart souboru s daty z výpočtu do naší domovské složky

clean_scratch
Vymazání pracovního adresáře na výpočetním stroji

Tento skript se pak spustí příkazem „qsub.“ Výsledné souřadnice optimalizované struktury jsou uložené v tzv. restart souboru („*optimalizace.rst*"), zatímco informace o energii jsou ve výstupním souboru („*optimalizace.out*“).

Optimalizace v programu GROMACS

Vstupní soubor v programu musím mít koncovku MDP a má následující formát [30]. GROMACS, v porovnání s Amberem, používá jiné jednotky (nm vs. Å, kJ/mol vs. kcal/mol), klíčová slova ve vstupním souboru nejsou rozdělena do sekcí a používá se tzv. volný formát (nadbytečné volné řádky a mezery se ignorují, slova mohou být na libovolné pozici v řádku).

integrator = steep
Minimalizace metodou SD

emstep = 0.01
Maximální délka kroku v nanometrech

emtol = 1.0
Konvergenční kritérium v kJ/mol (pokud je změna v energii menší než tato hodnota, tak se optimazace ukončí)

nsteps = 1000
Maximální počet kroků

nstxout = 100
Frekvence zápisu strukturních dat během optimalizace (každých 100 kroků)

nstlog = 100
Frekvence zápisu energií

cutoff-scheme = Verlet
Způsob rozdělení částic do skupin pro efektivní výpočet elektrostatických interakcí

nstlist = 10
Frekvence kontroly a update rozdělení částic (každých 10 kroků)

pbc = xyz
Periodické okrajové podmínky ve všech směrech

rlist = 1.0
Cutoff rozdělování částic (v nanometrech)

coulombtype = pme
Metoda pro výpočet elektrostatiky (smooth partical mesh Ewald)
r_coulomb = 1.0

Cutoff pro elektrostatickou interakci (v nanometrech)

vdwtype = cut-off
Způsob vyčíslení van der Waalsový interakcí (pouze do specifikované vzdálenosti)

vdw-modifier = force-switch
Vyhlazení vdW potenciálu tak, aby ve specifikované vzdálenosti měl nulovou derivaci

```
rvdw-switch = 0.9
```
Vzdálenost, od které se potenciál začne utlumovat

```
rvdw = 1.0
```
Maximální vzdálenost (cutoff)

Před spuštěním samotné simulace je nutné nejprve zkompilovat vstupní MDP soubor společně se vstupními souřadnicemi a topologií do tzv. TPR souboru. Ten má binární formát a obsahuje všechny data potřebná k provedení simulace. Ke kompilaci se používá modul „grompp“ [30]

```
gmx grompp -f optimalizace.mdp -c optimalzace.gro -p optimalizace.top
-o optimalizace.tpr
```

Simulace se provádí pomocí modulu „mdrun“, který je na MetaCentru VO nutné spustit pomocí skriptu pro frontový systém PBS:

```
#!/bin/bash
#PBS -l select=1:ncpus=4:mem=500MB:scratch_local=500MB
#PBS -l walltime=01:00:00
#PBS -N actin

module load gromacs

cd $PBS_O_WORKDIR/
cp optimalizace.tpr $SCRATCHDIR/
cd $SCRATCHDIR

mpirun gmx_mpi mdrun -s optimalizace.tpr -deffnm optimalizace

cp optimalizace.* $PBS_O_WORKDIR/

clean_scratch
```

Výsledné souřadnice jsou v tomto případě uloženy v textovém souboru „*optimalizace.gro*“, zatímco celá trajektorie obsahující jednotlivé struktury ukládané během optimalizace je binární soubor „*optimalizace.trr*“. Energie jednotlivých struktur a jejich příspěvky jsou uloženy v binárním souboru „*optimalizace.edr*“. Informace o průběhu simulace lze nalézt v textovém souboru „*optimalizace.log*“.

4.6 Molekulární dynamika

Simulace metodami molekulární dynamiky (MD) jsou důležitou součástí studia struktur makromolekul v biologii. Původně se nahlíželo na proteiny jako na rigidní struktury, ale dnes se používá dynamický model. V tomto modelu je nejdůležitější pro jejich funkci vnitřní pohyb a konformační změny, které z něho vyplívají.

Molekulární dynamika je numerická metoda, která umožňuje zkoumat interakce a pohyb molekul v odlišných podmínkách (těmi mohou být tlak, teplota a koncentrace). Metoda vychází ze znalosti potenciálové funkce, z které lze pro každou strukturu určit síly působící na jednotlivé atomy. Ze znalosti sil lze pak predikovat další strukturu, která odpovídá dynamickému vývoji takového molekulárního systému v čase [28-30].

Díky molekulární dynamice je možné simulovat různé vlastnosti molekulárních systémů. Například můžeme simulovat jejich termodynamické vlastnosti, kinetiku reakcí, interakci molekul s okolím a konformaci molekul. MD se využívá například v chemii, biologii, fyzice, inženýrství a materiálových vědách. Dá se využít k navrhování léků, výzkumu a vývoji nových látek, ke zkoumání proteinů a enzymů, k tomu abychom zjistili, jak se materiály budou chovat v extrémních podmínkách.

Integrace Newtonových rovnic

Molekulární dynamika je tedy založená na řešení pohybových rovnic. V molekulárně mechanickém popisu se k tomuto účelu používají klasické Newtonovy rovnice pro systém částic [28]

$$\vec{F}_i = m_i \vec{a}_i = m_i \frac{x^2 \vec{R}_i}{dt^2}, \tag{14}$$

kde vektor $\vec{a}_i$ je zrychlení i-té částice o hmotnosti m_i, které je způsobené působící silou $\vec{F}_i = -\nabla E$. Zrychlení ale přímo souvisí s polohou částice (je to druhá derivace trajektorie), takže integrováním těchto pohybových rovnic můžeme z působící síly měnící se polohu spočítat. Existuje několik způsobů, jak tuto integraci numericky provést, které se liší přesností a výpočetní náročností. V biomolekulárních simulacích se nejčastěji používá algoritmus známý jako „velocity Verlet“, který v každém časovém kroku Δt predikuje nové polohy $\vec{R}_i$ a rychlosti $\vec{v}_i$ jednotlivých atomů podle vztahů [29, 30]

$$\vec{R}_i(t + \Delta t) = \vec{R}_i(t) + \vec{v}_i(t)\Delta t + \frac{1}{2}\vec{a}_i(t)[\Delta t]^2 \tag{15}$$

$$\vec{v}_i(t + \Delta t) = \vec{v}_i(t) + \frac{1}{2}[\vec{a}_i(t + \Delta t) + \vec{a}_i(t)]\Delta t \quad (16)$$

K molekulární dynamice tedy potřebujeme znát počáteční polohy všech atomů (tedy strukturu dané molekuly), síly působící na tyto atomů (které se dají přímo spočítat z potenciální funkce) a jejich počáteční rychlosti. Ty se přiřadí jednotlivým atomům náhodně tak, aby měl celý systém požadovanou teplotu, protože ta je přímo spojená s průměrnou kinetickou energií částic. Náhodně přiřazené rychlosti samozřejmě nemusí odpovídat skutečným rychlostem, které by atomy měly mít, a proto je počáteční část MD simulací (tzv. ekvilibraci) dělat se zvýšenou opatrností, protože může docházet k nečekanému chování systému (např. prudké zvýšení teploty a náhlým strukturním změnám, u proteinů např. změny v cis-trans orientaci peptidových vazeb anebo poškození sekundární struktury v důsledku narušení pravidelného uspořádání vodíkových vazeb). Aby se takovému nestabilnímu chování předešlo, provádí se před samotnou MD simulací zpravidla geometrická optimalizace.

Dalším faktorem, který zásadně ovlivňuje stabilitu a přesnost simulací, je délka časového kroku Δt. Aby byla numerická integrace přesná, měl by být tento krok řádově menší než perioda nejrychlejšího vibračního pohybu v systému. V proteinech nejrychleji vibrují atomy vodíku, protože jsou lehké a zároveň relativně pevně vázané na ostatní atomy (nejčastěji uhlík). Tyto vibrace mají frekvence v rozmezí 2800 až 3200 cm^{-1}, což odpovídá dobám kmitu zhruba 10 fs [28]. Proto by časový krok v MD proteinů neměl být větší než 1 fs. Na druhou stranu chceme ale, aby byl tento krok co největší, protože pohyby proteinů mohou být pomalé (řádově až v oblasti μs) a malý časový krok zvětšuje náročnost těchto simulací (1 μs odpovídá 10^9 kroků o délce 1 fs). Proto se často v biomolekulárních MD simulacích fixují vzdáleností vazeb s vodíkovými atomy, aby se tyto rychlé vibraci odstranily, a používá se dvojnásobný časový krok, tedy Δt = 2 fs.

4.6.1 Ekvilibrace teploty

S dostatečně malým krokem bude integrace Newtonových rovnic relativně přesná. To znamená, že se nebude měnit celková mechanická energie (tedy součet potenciální a kinetické energie) systému. Bude se ale měnit teplota, což je u simulací proteinů nežádoucí. Proteiny se zpravidla nacházejí v prostředí o pokojové teplotě (25 °C = 298 K), a MD výpočty by měly simulovat tyto podmínky. Proto se k integračním rovnicím přidává ještě tzv. termostat [29, 30], který koriguje velikost rychlostí částic tak, aby byla se celková teplota měnila co nejméně. MD

simulace biomolekul tedy začínají ekvilibrací teploty. Počáteční rychlosti sice odpovídají požadované teplotě, ta se ale během prvních simulačních kroků změní s tím, jak se celý model relaxuje. Je potřeba tedy zapojit termostat a běžet dostatečně dlouhou dynamiku, dokud se teplota neustálí na nastavené hodnotě.

Ekvilibrace teploty v programu Amber

Vstupní soubor Amberu, který je zde pojmenován jako „*dynamika.inp*“, je potřeba modifikovat následujícím způsobem:

```
&cntrl
 imin=0,
```
MD simulace

```
 nstlim=100000
```
Celkový počet kroků

```
 dt=0.001,
```
Časový krok udávaný v pikosenkundách (1 ps = 1000 fs)

```
 ntx=1,
 irest=0,
 cut=10,
 temp0=300.0,
```
Referenčí teplota, na které má být systém udržován (v Kelvinech)

```
 tempi=300.0,
```
Počáteční teplota systému (v Kelvinech)

```
 ntt=3,
```
Typ termostatu (v tomto případě Langevinův stochastický termostat)

```
 gamma_ln=1,
```
Srážková frekvence termostatu (v převrácených pikosekundách, tedy ps^{-1})

```
 ntxo=1,
 ntpr=100,
 ntwx=100,
```
Frekvence zápisu strukturní souřadnic do trajektorie (každých 100 kroků)

```
 ntwr=100,
```
Frekvence zápisu restart souboru

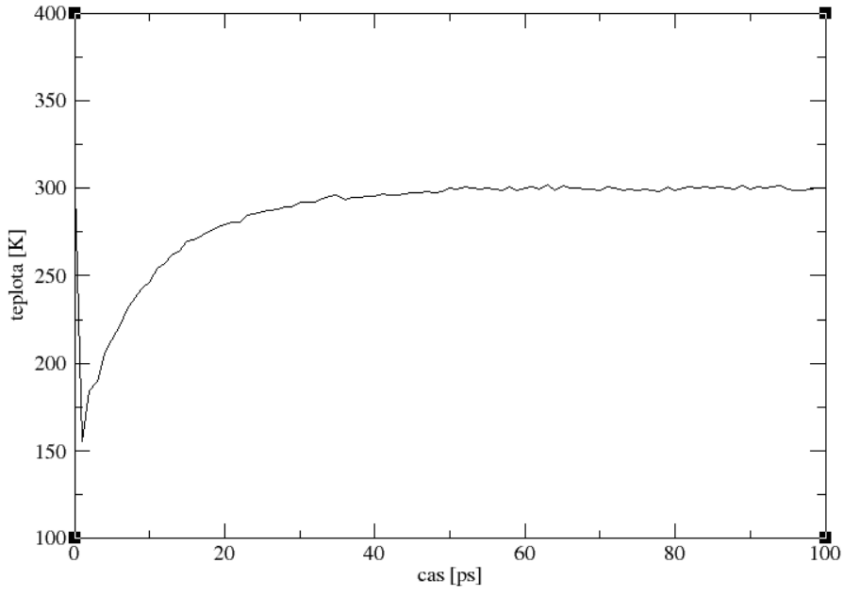
```
 ntb=1
/
```

MD simulace se pustí následujícím příkazem, který je třeba opět napsat do skriptu pro frontový systém a odeslat příkazem „qsub“

```
mpirun sander.MPI -O -i dynamika.inp -o dynamika.out -r dynamika.rst
-x dynamika.trj -c optimalizace.rst -p actin.top
```

V této simulaci jsme jako výchozí strukturu použily částečně optimalizovanou geometrii („*optimalizace.rst*“) actinu. Výsledkem simulace je MD trajektorie „*dynamika.trj*“, kterou je možné načíst např. do VMD a pohyb proteinu si prohlédnout. Energie a teploty jednotlivých struktur lze nalézt v souboru „*dynamika.out*.“

Časový vývoj teploty během této 100 ps simulace je zobrazen na obrázku 21. Jak je vidět, během prvních několik kroků teplota rapidně poklesne z počátečních 300 K téměř na polovinu, protože struktura je částečně optimalizována a efektivně „zamražena“ v minimu potenciální funkce (síly působící na jednotlivé atomy jsou zde malé, a proto je malé i zrychlení, a to pak snižuje rychlosti těchto atomů). Jak ale simulace pokračuje, jsou rychlosti korigovány termostatem a systém (aktin ve vodném roztoku) se ohřívá. Požadovaná teplota 300 K je dosažena zhruba po 50 ps a po zbytek simulace se již nemění.



Obrázek 21: *Časový vývoj teploty v průběhu MD ekvilibrace (program Amber, Langevinův termostat, srážková frekvence 1 ps^{-1}, požadovaná teplota 300 K).*

Ekvilibrace teploty v programu Gromacs

Postup je stejný jako s programem Amber. Nejdříve byla provedena simulace dynamika, abychom dali strukturu do pohybu. Dále byla provedena simulace, abychom dali struktuře konstantní tlak. Poslední byla provedena finální dlouho běžící simulace, ze které byli provedeny analýzy popsané v části 4.7.

Zde je vstupní MDP soubor:

```
integrator = md
Molekulární dynamika

nsteps = 10000

dt = 0.001
Časový krok v pikosekundách

nstxout = 100
nstlog = 100

cutoff-scheme = Verlet
nstlist = 10
pbc = xyz
rlist = 1.0
coulombtype = pme
r_coulomb =  1.0
vdwtype = cut-off
vdw-modifier = force-switch
rvdw-switch = 0.9
rvdw = 1.0

Následující příkazy jsou určeny pro nastavení termostatu
tcoupl = berendsen
tc-grps = system
tau-t = 1.0
ref-t = 300
```

Simulace se pak spustí následujícím příkazem:

```
mpirun gmx_mpi mdrun -s dynamika.tpr -deffnm dynamika
```

4.6.2 Ekvilibrace tlaku

Po počáteční ekvilibraci teploty je nutné ekvilibrovat také hustotu modelového systému. Voda přidaná při stavbě modelu okolo proteinu nemá na povrchu tohoto požadovanou strukturu a zpravidla nevyplňuje všechny dutiny. Tyto strukturní nedostatky se většinou samovolně opraví

již během první MD simulace, hustota modelu ale zpravidla neodpovídá reálné hustotě, která by v případě solvatovaných proteinů měla být jen o něco větší než 1 g/cm^3. Při optimální hustotě by průměrný tlak v modelu měl být stálý (u simulací proteinů blízký atmosférickému tlaku). Aby se toho dosáhlo, je potřeba měnit velikost simulovaného boxu (zmenšení velikosti boxu vede ke zmenšení objemu a zvýšení tlaku a naopak). Toto v MD simulacích zařizuje tzv. barostat [29, 30].

Ekvilibrace tlaku v programu Amber

Vstupní soubor pojmenovaný jako „*hustota.inp*“ je modifikovaný následujícím způsobem:

```
&cntrl
 imin=0,
 nstlim=500000,
 dt=0.001,
 ntx=5,
```
Vstupní soubor obsahuje nejen souřadnice, ale také rychlosti atomů

```
 irest=1,
```
Počáteční rychlosti se znovu negenerují, ale použijí se ty ze vstupního souboru

```
 cut=10,
 temp0=300.0,
 ntt=3,
 gamma_ln=1,
 ntp=1,
```
Isotropní barostat (velikost boxu se mění ve všech směrech stejně)

```
 system0=1,
```
Referenční tlak (v barech, kde 1 bar=0,987 atm)

```
 taup=2.0,
```
Doba tlakové relaxace (v pikosekundách)

```
 ntxo=1,
 ntpr=100,
 ntwx=500,
 ntwr=500,
 ntb=2,
```
Periodické okrajové podmínky pro systém, kde se mění velikost boxu

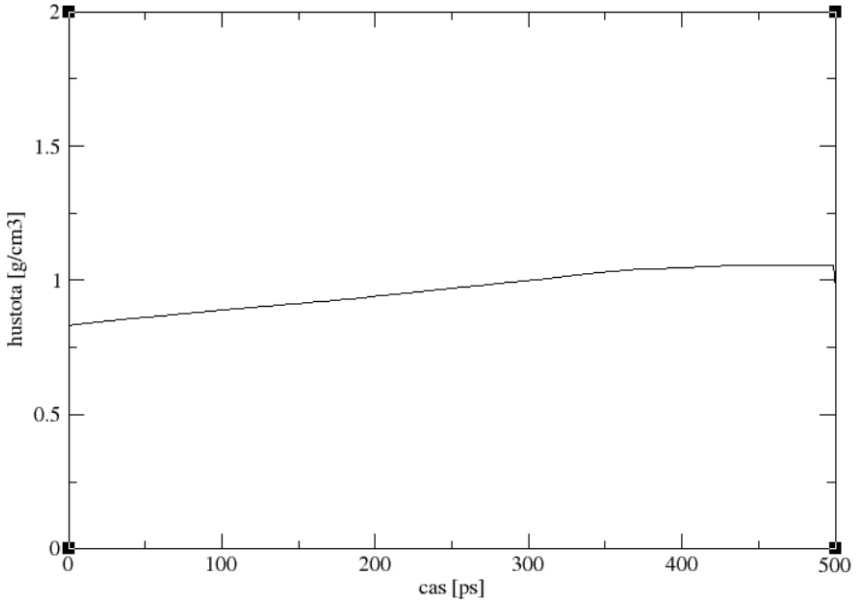
```
 iwrap=1,
```
Tento příkaz zabalí souřadnice zapsané do souboru restart a trajektorie do primárního boxu.

Příkaz pro spuštění MD simulace v skriptu pro frontový systém:

```
mpirun sander.MPI -O -i hustota.inp -o hustota.out -r hustota.rst -x
hustota.trj -c dynamika.rst -p actin.top
```

Jako vstupní souřadnicový soubor pro ekvilibraci tlaku použijeme restart soubor „*dynamika.rst*“ z předchozí ekvilibrace teploty, kde jsou uloženy polohy i rychlosti jednotlivých atomů. Struktury se během MD zapisují do trajektorie „*hustota.trj*“ a poslední geometrie, rychlosti a velikost boxu jsou uloženy v „*hustota.rst*.“ Informace o energii, teplotě, tlaku a hustotě se zapisuje do souboru „*hustota.log*.“

Průběh hustoty během ekvilibrace tlaku je zobrazen na obrázku 22. Relaxace je tentokrát pomalejší než v případě teploty. Počáteční hustota je relativně nízká (~0,8 g/cm^3) a v průběhu simulace pozvolna stoupá až se po čase 400 ps ustálí na hodnotě ~1,1 g/cm^3.



Obrázek 22: *Časový vývoj hustoty v průběhu MD ekvilibrace (program Amber, isotropický barostat s relaxační dobou 2 ps, požadovaná teplota 300 K a tlak 1 bar).*

Ekvilibrace tlaku v programu Gromacs

Kontrola tlaku pomocí barostatu se nastaví v MDP souboru následovně:

```
integrator = md
nsteps = 100000
dt = 0.001

nstxout = 500
nstlog = 500
nstenergy = 500
```

```
cutoff-scheme = Verlet
nstlist = 10
pbc = xyz
rlist = 1.0
coulombtype = pme
r_coulomb =  1.0
vdwtype = cut-off
vdw-modifier = force-switch
rvdw-switch = 0.9
rvdw = 1.0

tcoupl = berendsen
systéms = system
tau-t = 1.0
ref-t = 300.0
```

Zde je nastavení barostatu:

```
pcoupl = berendsen
pcoupltype = isotropic
compressibility = 4.5e-5
tau-p = 10.0
ref-p = 1.0
```

Zde je pak příkaz pro spuštění simulace:

```
mpirun gmx_mpi mdrun -s hustota.tpr -deffnm hustota
```

4.6.3 Produktivní simulace

Po dosažení správné teploty a tlaku je možné konečně provést finální MD simulaci, která bude sloužit k samotnému studiu a analýzám daného proteinu. Tato simulace se označuje jako produktivní (angl. *production run*) a na rozdíl od ekvilibrací bývá velmi dlouhá (v praxi až několik μs), což je velmi výpočetně náročné. Proto se tyto simulace často dělají pouze za použití termostatu (barostat, který zpomaluje výpočet, se většinou neaplikuje) a fixují se vazby s vodíkovými atomy, aby bylo možné použít delší časový krok (typicky 2 fs). Současně se využívá paralelizace na více procesorech a dnes i běžně GPU akcelerace [29, 30]. Velikost výsledné trajektorie se minimalizuje méně častým zápisem strukturních dat a jejich ukládáním v binárních datových formátech, které mohou být i komprimované.

Produktivní simulace v programu Amber

Vstupní soubor nazvaný „*simulace.inp*“ je podobný tomu, který jsme použili pro ekvilibraci teploty. Liší se delším časovým krokem, fixací vazeb s vodíkovými atomy a binárním formátem trajektorie.

```
&cntrl
 imin=0,
 nstlim=10000000,
 dt=0.002,
 ntf=2
```
Modifikace výpočtu síly, kdy možnost 2 znamená, že vazebné interakce zahrnující H-atomy jsou vynechány

```
 ntc=2
```
Fixace vazeb s vodíkovými atomy pomocí algoritmu SHAKE (možnost 2 je používána pro TIP3P model vody)

```
 ntx=5,
 irest=1,
 cut=10,
 temp0=300.0,
 ntt=3,
 gamma_ln=1,
 ntxo=1,
 ioutfm=1,
```
Binární NetCDF formát trajektorie

```
 ntpr=1000,
 ntwx=1000,
 ntwr=1000,
 ntb=1,
 iwrap=1
/
```

Skript pro frontový systém je potřeba modifikovat tak, abychom využili GPU akcelerace. Proto jej zde uvádíme celý:

```
#!/bin/bash
#PBS -l select=1:ncpus=1:ngpus=1:mem=5GB:gpu_mem=5GB:scratch_local=10 B
```
Požadavek na výpočetní stroj s GPU s pamětí 5 GB, paralelizace na CPU nebude použita.

```
#PBS -l walltime=04:00:00
#PBS -q gpu
```
Při GPU akceleraci je nutné odeslat výpočet do specifické fronty

```
#PBS -N actin
module load amber-14-gpu8
```
Verze Amber14 s GPU akcelerací

```
cd $PBS_O_WORKDIR/
cp simulace.inp $SCRATCHDIR/
cp hustota.rst $SCRATCHDIR/
cp actin.top $SCRATCHDIR/
cd $SCRATCHDIR/
```

```
pmemd.cuda -O -i simulace.inp -o simulace.out -r simulace.rst -x simulace.trj
-c hustota.rst -p actin.top
```
Optimalizovaný PMEMD modul s GPU akcelerací

```
cp simulace.out $PBS_O_WORKDIR/
cp simulace.rst $PBS_O_WORKDIR/
cp simulace.trj $PBS_O_WORKDIR/
clean_scratch
```

Produktivní simulace v programu Gromacs

Finální simulace je provedena za pomoci následujícího nastavení v MDP souboru:

```
integrator = md
nsteps = 10000000
dt = 0.002
```
Delší časový krok

```
nstxout = 0
nstxout-compressed = 1000
```
Komprimovaný formát trajektorie

```
nstlog = 1000
nstenergy = 1000
cutoff-scheme = Verlet
nstlist = 10
pbc = xyz
rlist = 1.0
coulombtype = pme
r_coulomb =  1.
vdwtype = cut-off
vdw-modifier = force-switch
rvdw-switch = 0.9
rvdw = 1.0

constraints = h-bonds
constraint-algorithm = lincs
```
Zafixování vazeb s vodíkovými atomy

```
tcoupl = nose-hoover
tc-grps = system
tau-t = 10.0
ref-t = 300.0
```

Gromacs na rozdíl od Amberu se spouští stejným příkazem, rozdíl je pouze v nastavení frontového systému (viz výše):

```
mpirun gmx_mpi mdrun -s simulace.tpr -deffnm simulace
```

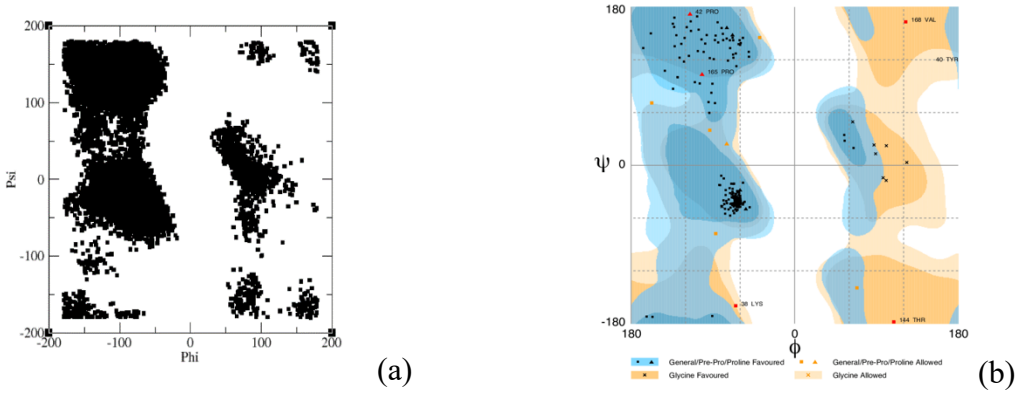
4.7 Zpracování a analýza dat

Výsledkem molekulárně-dynamických simulací jsou tzv. trajektorie, tj. soubory dat se souřadnicemi atomů, které byly uloženy v průběhu simulace. Jsou to tedy strukturní data, které jsou závislé na čase. Tyto data je potřeba zanalyzovat a správně interpretova, abychom mohli vyvodit určité závěry o chování simulovaného systému. Záleží také na tom, jestli chceme analyzovat samotnou strukturu a její stabilitu, nebo její dynamické vlastnosti, popř. nějaké energetické charakteristiky. Mezi základní strukturní analýzy proteinů patří RMSD, Ramachandranův diagram, nebo histogramy specifických geometrických parametrů jako jsou vzdálenosti a úhly [28-30]. Dynamiku proteinů lze charakterizovat např. pomocí RMSF nebo vibračních analýz. Mezi pokročilejší analýzy pak patří např. charakteristiky dynamického chování vodíkových vazeb v solvatační vrstvě okolo proteinů, difuse specifických iontů či skrz proteinové struktury, změny volných energií během enzymatických reakcí, vazebné energie alosterických ligandů, atd. [28-30]. V této části se představujeme nejběžnější ze základních metod.

4.7.1 Ramachandranův diagram

Ramachandranův diagram je grafický nástroj, který se využívá ve strukturní biologii a biochemii. Slouží ke zobrazení možných prostorových uspořádání peptidového řetězce v závislosti na rotaci kolem vazeb mezi atomy C_α a N v aminokyselinách. Jde o kvalitativní analýzu, kdy se zobrazuje rozložení peptidických úhlů ψ a ϕ v rovině okolo těchto dvou vazeb pro jednotlivé páry aminokyselin. Protože různé prvky sekundární struktury (α-šroubovice, β-skládané listy) mají dobře definovanou strukturu, dá se jejich zastoupení v daném proteinu z Ramachandranova diagramu vyčíst. Např. β-skládané listy se v diagramu nachází v horním levém rohu, pravotočivé α-šroubovice jsou vlevo (mírně v záporných ψ hodnotách), zatímco levotočivé α-šroubovice jsou v pravé části (mírně v kladných ψ hodnotách).

Ukázka Ramachandrova diagramu aktinu je na obrázku 20. V levé části je diagram, který jsme napočítali z posledních 100 ps produktivní MD trajektorie (aktin je značně flexibilní protein, a při použití celé trajektorie by „skvrny“ na diagramu byly příliš velké, splývaly by, a byly by špatně rozpoznatelné). V pravé části je pak referenční diagram převzatý z publikace [37]. Jak je vidět, oba diagramy jsou kvalitativně shodné. Aktin evidentně obsahuje jak β-skládané listy, tak běžné pravotočivé α-šroubovice. Díky flexibilitě proteinů jsou zastoupeny i levotočivé α-šroubovice, které u rigidnějších bílkovin nejsou typické a někdy mohou indikovat i poškození proteinové struktury.



(a) (b)

Obrázek 20: *Ramchandranův plot. Obrázek (a) je námi vytvořený Ramachandranův diagram. Obrázek (b) je převzatý ze zdroje [37] a slouží zde jako reference.*

Ramachandranův diagram je možné nejsnadněji získat pomocí programu VMD [19]. Po načtení trajektorie je možné ho zobrazit pomocí menu „Extensions“ → „Analyses“ → „Ramachandran Plot“. To je praktické, protože po provedení MD simulací je vhodné nejprve výslednou trajektorii vizuálně zkontrolovat, abychom se ujistili, že nedošlo k nějakému nečekanému chování (např. poškození struktury). Zobrazení Ramachandranova diagramu a jeho kvalitativní zhodnocení je vhodné doplnění této počáteční inspekce.

4.7.2 RMSD

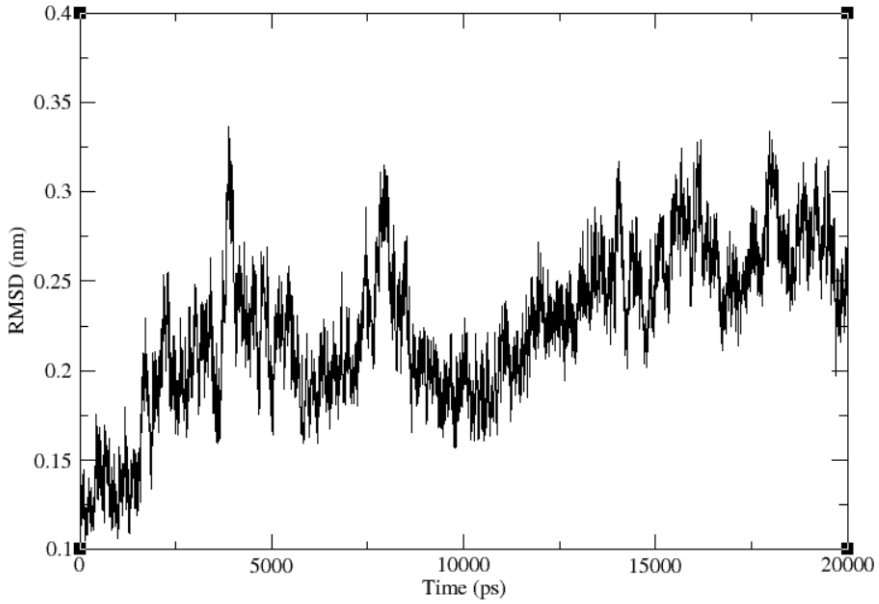
Root-Mean-Square Deviation (RMSD, střední kvadratická odchylka atomových pozic) je typ analýzy, která se používá zhodnocení stability studované struktury [29,30]. RMSD se vypočítá jako druhá odmocnina ze středních čtverců vzdáleností mezi odpovídajícími atomy dvou struktur, jedné z MD trajektorie ($\vec{R}(t)$) a druhé referenční ($\vec{R}_0$):

$$RMSD(t) = \sqrt{\frac{1}{T}\sum_{t\prime}^{T}\left[\vec{R}(t) - \vec{R}_0\right]^2} \tag{17}$$

V případě simulací proteinů se jako referenční bere většinou krystalická struktura (popř. jiná experimentálně získaná geometrie). RMSD je pak časově závislá funkce, která nám říká, jak moc se v daném čase během MD simulace struktura studovaného proteinu lišila od té referenční. Čím nižší je hodnota RMSD, tím jsou si porovnávané struktury podobnější.

Ukázka RMSD, které bylo napočítáno z 20 ns trajektorie aktinu při konstantní teplotě 300 K, je na obrázku 21. V tomto případě jsme počítaly RMSD jen pro tzv. kostru proteinu

(backbone), což jsou v podstatě pouze atomy, které jsou součástí peptidických vazeb mezi jednotlivými aminokyselinami (tedy C_α uhlík, C a O atomy z karboxylické skupiny, N a H atomy z amino skupiny). Jak je vidět, na počátku trajektorie byl aktin v konformaci relativně blízké krystalické struktuře (RMSD ~0,1 nm), ale postupem času se tato hodnota zvýšila a ustálila se na ~0,25 nm. To indikuje, že struktura aktinu je relativně flexibilní (rigidní proteiny jako např. cytochromy, kde jsou peptidické řetězce kovalentně provázané s těžko deformovatelnými organickými ligandy – hemy – mají RMSD pro backbone blízké ~0,1 nm).



Obrázek 21: *RMSD pro backbone aktinu během 20 ns MD simulace při konstantní teplotě 300 K.*

Výpočet RMSD v Amberu

RMSD z trajektorií napočítaných Amberem je možné získat pomocí programu CPPtraj, který je součástí tohoto programového balíku a slouží k provádění různých analýz. To, jakou analýzu chceme provést, je nutné specifikovat ve vstupním skriptu, který pro RMSD vypadá následovně:

```
parm simulace.top
```
Načtení topologie systému ze souboru „*simulace.top*“

```
trajin simulace.trj
```
Načtení strukturních dat z MD trajektorie uložené v souboru „*simulace.trj*“

reference aktin.crd
Načtení referenční geometrie (model krystalické struktury uložený v souboru „*aktin.crd*“)

rmsd r1 :1-368&!@H= reference
Výpočet RMSD pro všechny atomy z residuí 1 až 368, které nejsou vodík (tzv. těžké atomy, pole hodnot RMSD je označené jako „r1“)

rmsd r2 :1-368&@CA,C,O,N,H
Výpočet RMSD pro backbone atomy z residuí 1 až 368, pro výběr jsou požitá jména atomů, tak jak jsou definovaná v topologii (pole hodnot RMSD je označné jako „r2“)

run
Spuštění výše definovaných výpočtů

write rmsd.dat r1 r2
Hodnoty RMSD uložené v polích „r1“ a „r2“ se uloží do souboru „rmsd.dat“

quit
Konec skriptu

Výsledný soubor „rmsd.dat“ je textový a v tomto případě obsahuje 3 sloupce dat: čas, RMSD pro všechny těžké atomy a RMSD pro kostrové (backbone) atomy. K zobrazení těchto dat je možné použít běžný software pro tvorbu grafů (např. Gnuplot nebo Xmgrace).

Výpočet RMSD v Gromacsu

V Gromacsu se výpočet RMSD provádí zpravidla ve dvou krocích. Protože protein se může během MD simulací volně pohybovat, dochází běžně i k situacím, kdy překračuje stěny simulačního boxu. Když ale používáme periodické okrajové podmínky, tak se struktura proteinů zdánlivě naruší, protože atomy, které stěnu boxu překročí, se automaticky vloží na druhou stranu boxu (tzv. wrapping). Modul „rms“, který se v Gromacsu používá pro výpočet RMSD, ale neumí strukturu proteinu automaticky zcelit. To je nutné udělat zvlášť, pomocí jiného modulu, který se jmenuje „trjconv,“ což je první krok.

```
gmx trjconv -f simulace.xtc -s aktin.gro -n aktin.ndx -fit rot+trans
-o rmsd.xtc
```

V tomto případě zcelujeme protein tak, že jej nafitujeme na původní krystalickou strukturu, která je počátečním modelu („*aktin.gro*“) uprostřed boxu. V souboru „*aktin.ndx*“ jsou uložené indexy backbone atomů, pomocí kterých se fitování (překládání) struktur provádí. Zcelení struktury by šlo udělat také volbou „-pbc whole“, ale fitování na referenční strukturu je většinou lepší, protože takto předělaná trajektorie se snadněji zobrazuje. Takto předělaná

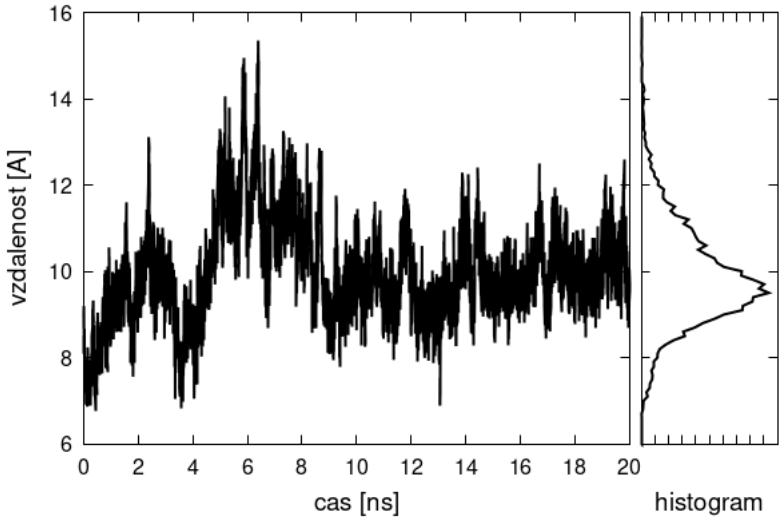
trajektorie, uložená do souboru „*rmsd.xtc*“ (do kterého není nutné ukládat souřadnice solventu, stačí jen protein), se pak použije k výpočtu RMSD

```
gmx rms -f rmsd.xtc -s aktin.gro -n aktin.ndx -o rmsd.xvg
```

Výsledný soubor „*rmsd.xvg*“ obsahuje 2 datové sloupce – simulační čas a RMSD hodnoty. Soubor je vhodný k zobrazení pomocí program Xmgrace.

4.7.3 Histogramy geometrických parametrů

Další kvantitativní analýzou, která se často používá k vyšetřování proteinových struktur a jejich stability, je detekce různých geometrických parametrů (vzdálenosti, úhly) a jejich distribucí (histogramů) [29, 30]. Například peptidický řetězec proteinu aktin je složen do struktury, která tvarem připomíná písmenu U, jehož konce se od sebe mohou přibližovat a vzdalovat. Abychom věděli, do jaké nejmenší a největší vzdálenosti se mohou tyto části proteinu dostat, můžeme detekovat vzdálenost mezi vybranými atomy v těchto částech. Vizualizací lze zjistit, že na jednom konci se nalézá alanin v pozici 59 a na druhém threonin v pozici 199. Vzdálenost mezi uhlíky C_α těchto aminokyselin je zobrazena na obrázku 22 jako funkce simulačního času. Z těchto vzdáleností je pak sestrojen histogram, který je v pravé části obrázku. Z něho vidět, že typickou (nejčastější) vzdáleností mezi těmito částmi proteinu je 0,95 nm. Vzdálenost se může zmenšit až na 0,7 nm, což prakticky odpovídá kontaktu obou konců (okolo C_α uhlíků jsou ještě postranní řetězce). Dochází ale i k jejich oddalování, a to až do vzdálenosti 1,4 nm.



Obrázek 22: *Vzdálenost mezi Cα uhlíky aminokyselin Ala59 a Thr199 během 20 ns MD simulace. V pravé části obrázku je histogram této vzdálenosti.*

Výpočet vzdáleností v Amberu

Pro výpočet vzdáleností opět použijeme CPPtraj modul. Zde je ukázka vstupního souboru:

```
parm simulace.top
trajin simulace.trj

distance d :59@CA :199@CA out distance.dat
```
Spočítá vzdálenost mezi Cα uhlíky na residuích 59 a 199 (pole vzdáleností je označeno „d“ a zároveň se uloží do souboru „*distance.dat*“)

```
hist d,6.0,16.0,*,100 out distance.hst normint
```
Sestrojí histogram v rozmezí 6 až 16 A ze vzdáleností (histogram je rozdělen na 100 dílků, je přeškálován tak, aby plocha pod ním byla rovna jedné, a uloží se do souboru „*distance.hst*“)

```
run
quit
```

Výsledné soubory „*histogram.dat*“ a „*histogram.hst*“ jsou textové a můžeme je zobrazit pomocí programů Gnuplot nebo Xmgrace.

Výpočet vzdáleností v Gromacsu

V programovém balíku Gromacs lze vzdálenosti a jejich histogram spočítat z MD trajektorie pomocí modulu „distance“:

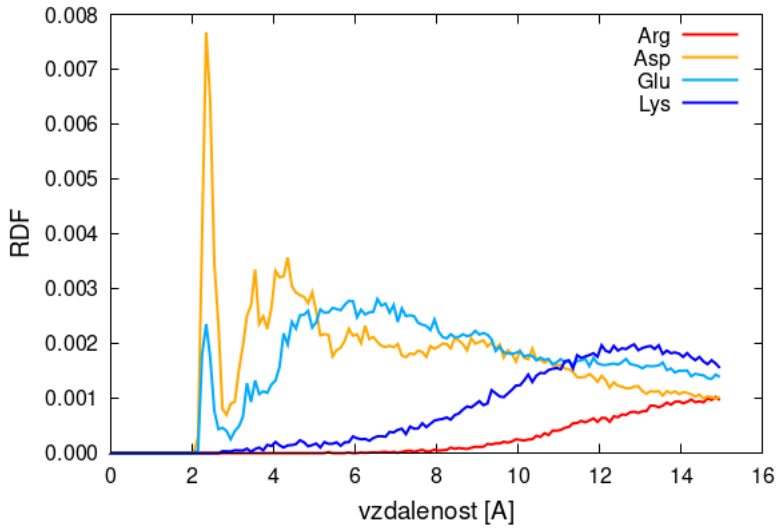
```
gmx distance -f simulace.xtc -s simulace.tpr -n distance.ndx -oall
distance.xvg -oh distance-hist.xvg -select 0 -len 1.0 -tol 0.5 -binw
0.01
```

Indexy atomů, mezi kterými se vzdálenost počítá, jsou uloženy v souboru “*distance.ndx*” jako první datový záznam (jeho výběr provádíme volbou “select”). Vzdálenosti v jednotlivých strukturách trajektorie „*simulace.xtc*“ se uloží do souboru „*distance.xvg*“. Histogram, v rozmezí 0,5 at 1,5 nm se šířkou jednoho dílku 0,01 nm se uloží do souboru „*distance-hist.xvg*“. Oba tyto XVG soubory jsou vhodné k přímému zobrazení pomocí program Xmgrace.

4.7.4 Radiální distribuční funkce

Další strukturní analýzou, která se nejčastěji používá k vyšetření rozložení solventu (molekul vody) nebo iontů okolo specifické části proteinu, je tzv. radiální distribuční funkce (RDF) [29, 30]. Tato funkce udává, kolik částic (molekul vodu, iontů, apod.) se nachází v určité vzdálenosti od nějakého atomu, aminokyseliny, nebo jiné části proteinu. Např. na obrázku 23 jsou porovnány RDF pro ionty sodíku (Na^+) okolo kladně (Lys, Arg) a záporně (Asp, Glu)

nabytých aminokyselin na povrchu aktinu. Protože ionty sodíku jsou záporně nabité, přitahují se elektrostatickou silou se záporně nabitými aminokyselinami a lze je tedy často najít v jejich okolí. Tomu odpovídá velký peak v RDF pro Asp a Glu ve vzdálenosti 2,2 nm, který nám říká že ionty jsou často v kontaktní vzdálenosti s karboxylovými (COO^-) skupinami těchto aminokyselin. Široký peak okolo 3,5 nm a dále pak ukazuje na to, že tyto ionty se často pohybují v těchto větších vzdálenostech, tam ale už nejsou tak těsně vázány (interakce je slabší). Naopak v okolí kladně nabitých aminokyselin ionty sodíku prakticky nenajdeme (RDF pro Arg a Lys mají pro krátké vzdálenosti velmi nízké hodnoty a rostou až okolo 1 nm a dále). To je samozřejmě způsobené elektrostatickým odpuzováním (repulzí) těchto kladně nabitých reziduí s kladnými ionty sodíku.



Obrázek 23: *Radiální distribuční funkce (RDF) detekující vzdálenost sodíkových iontů od nabitých aminokyselin na povrchu aktinu (funkce nejsou normalizované).*

Výpočet RDF v Amberu

Radiální distribuční funkce spočteme opět pomocí CPPtraj modulu s následujícím skriptem:

```
parm simulace.top
trajin simulace.trj

radial rdf-glu.dat 0.1 15.0 :GLU :Na+
```

Spočítá RDF pro vzdálenost sodíkových iontů (residua Na+) od všech glutamových kyselin (residua GLU). Funkce bude počítána pro vzdálenosti od 0,1 do 15 A a uložena do souboru „*rdf-glu.dat*.“

```
run
quit
```

Výsledný soubor „*rdf-glu.dat*“ je opět textový a vhodný k zobrazení v Gnuplotu nebo Xmgrace. RDF pro ostatní aminokyseliny se počítá obdobně.

Výpočet RDF v Gromacsu

V Gromacsu se RDF počítá pomocí modulu „rdf“:

```
gmx rdf -f simulace.xtc -s simulace.tpr -n rdf.ndx -o rdf.xvg -rmax
1.0 -cut 0.01
```

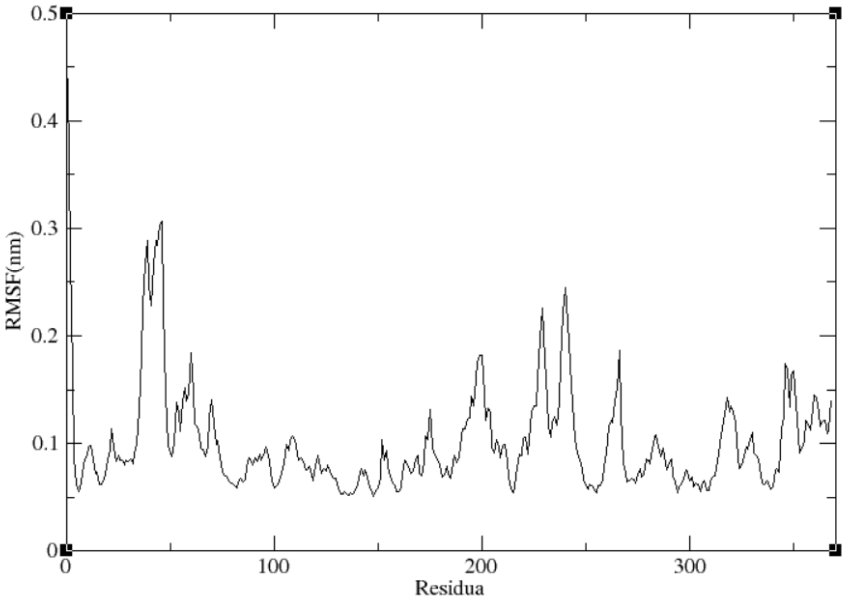
Skupiny atomů, mezi kterými se RDF počítá, jsou uvedeny v indexovém souboru „*rdf.ndx*“ a jejich výběr probíhá interaktivně po spuštění programu Volby „cut“ a „rmax“ určují rozsah vzdáleností. Výsledná funkce se uloží do souboru „*rdf.xvg*“, který je možné zobrazit pomocí Xmgrace.

4.7.5 RMSF

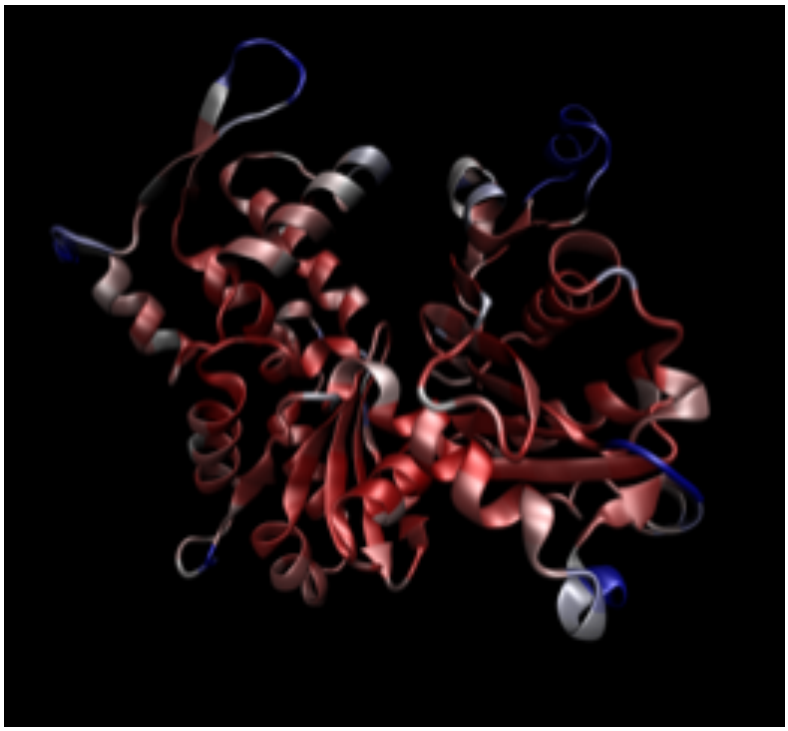
Root-Mean-Square Fluctuation (RMSF, střední kvadratická odchylka fluktuace) je analýza, která nám řekne, jak moc polohy jednotlivých atomů nebo residuí ve struktuře proteinů fluktuují okolo rovnovážné polohy [29, 30]. Je vypočítána jako druhá odmocnina střední hodnoty čtvercové vzdálenosti mezi polohou každého atomu $\vec{R}_i$ a jeho průměrnou polohou $\langle \vec{R}_i \rangle$ napříč simulační trajektorií.

$$RMSF_i(t) = \sqrt{\frac{1}{T}\sum_{t\prime}^{T}\left[\vec{R}_i(t) - \langle \vec{R}_i \rangle\right]^2} \tag{18}$$

Touto analýzou můžeme tedy zjistit flexibilitu jednotlivých atomů v celé proteinové struktuře. Pokud dostaneme vysoké hodnoty RMSF ukazuje to na vysokou flexibilitu atomu a jeho náchylnost ke fluktuacím, zatímco nízká hodnota RSMF ukazuje na rigidnost a menší náchylnost ke fluktuacím u atomů. To je ukázána na obrázku 24, kde je vyneseno RMSF pro jednotlivá residua (aminokyseliny) v aktinu. Vysoké peaku ukazují na místa, která se hodně pohybují. Tyto hodnoty je možné vložit do PDB souboru a zobrazit ve VMD, jak je ukázáno na obrázku 25, kde jsou rigidní části obarvené červeně, zatímco ty flexibilní mají modrou barvu.



Obrázek 24: *Hodnoty RMSF pro jednotlivá rezidua v aktinu získané z 20 ns MD simulace při konstantní templotě 300 K.*



Obrázek 25: *Struktura aktinu s grafickým zobrazením RMSF (červená označuje rigidní místa, modrá flexiblní). Zobrazeno ve VMD.*

Výpočet RMSF v Amberu

Pro výpočet RMSF pomocí modulu CPPtraj je potřeba následující skript:

```
parm simulace.top [Orig]
```
Načtená topologie ze souboru „simulace.top“ se označí jako „Orig“

```
Reference aktin.crd [Crystal]
```
Referenční struktura pro výpočet středních poloh (označena jako „Crystal“)

```
trajin simulace.trj
```
Trajektorie se strukturními daty

```
strip !(:1-368) outprefix rmsf
```
Odstraníme z trajektorie všechnu vodu a ionty (zůstanou jen proteinová residua 1 až 368, nová topologie se uloží do souboru „*rmsf.simulace.top*“)

```
rmsd :1-368@CA,C,O,N,H ref [Crystal]
```
Nafitujeme (přeložíme) kostru (backbone) jednotlivých struktur z trajektorie na referenční krystalovou geometrii

```
trajout rmsf.simulace.trj netcdf
```
Nafitované struktury se uloží nové trajektorie „*rmsf.simulace.trj*“ (binární NetCDF formát)

```
average rmsf.simulace.pdb pdb
```
Výpočet středních poloh jednotlivých atomů (uloží se do souboru „*rmsf.simulace.pdb*“)

```
run
parm rmsf.simulace.top [ProteinOnly]
```
Načtená topologie se přepíše změnšenou topologií pro protein bez solventu

```
reference rmsf.simulace.pdb parm [ProteinOnly] [Avrg]
```
Zmenšená topologie se použije pro načtení středních hodnot atomových pozic jako referenční struktury (ta se označí jako „Avrg“)

```
rmsd :1-368@CA,C,O,N,H ref [Avrg]
```
Backone atomy jednotlivých struktur z trajektorie se nafitují (přeloží) na střední pozice těchto atomů

```
atomicfluct out rmsf-b.dat :1-368@CA,C,O,N,H byres bfactor
```
Výpočet RMSF pro backbone atomy (hodnoty se průměrují přes jednotlivá residua a RMSF se převede na tzv. B-faktor používaný v krystalografii, data se uloží do souboru „*rmsf-b.dat*“)

```
atomicfluct out rmsf-r.dat :1-368 byres bfactor
```
Výpočet RMSF (hodnoty pro všechny atomy se průměrují přes jednotlivá residua, data se uloží do souboru „*rmsf-r.dat*“)

```
atomicfluct out rmsf-a.dat :1-368 byatom bfactor
```
Výpočet RMSF (hodnoty pro jednotlivé atomy, data se uloží do souboru „*rmsf-a.dat*“)

```
run
quit
```

Výpočet RMSF v Gromacsu

V Gromacsu je výpočet RMSF o něco přímočařejší než v Amberu. Stejně jako při výpočtu RMSD se musí nejprve struktura proteinu nafitovat na referenční strukturu pomocí modulu „trjconv“. Výsledná trajektorie „rmsd.xtc“ lze pak přímo použít k výpočtu RMSF pomocí „rmsf“ modulu“:

```
gmx rmsf -f rmsd.xtc -s aktin.gro -n rmsd.ndx -o rmsf.xvg -res -oq
rmsf.pdb
```

Soubor „*rmsf.xvg*“ obsahuje RMSF hodnoty pro jednotlivá residua a může být použit k vytvoření grafu jako na obrázku 24. Soubor „*rmsf.pdb*“ obsahuje B-faktory a lze jej použít k zobrazení struktury jak ona obrázku 25.

5 Závěr

Hlavním cílem mé práce bylo naučit se a popsat základní postupy, které se využívají při počítačových simulacích proteinů metodami molekulární dynamiky. Text práce by měl sloužit jako jakýsi úvodní text či příručka pro studenty z nefyzikálních, především biologických oborů, kteří se běžně neorientují ve výpočetních technikách a dostupné učebnice a programové manuály jsou pro ně těžko čitelné, protože jsou psané technickým stylem a často i s pomocí většího či menšího množství matematických formulí. Navíc jsou tyto manuály často velmi detailní a dlouhé (mají stovky až tisíce stran). Text práce se proto věnuje úplným základům, od získání struktury proteinu v online databázích, přes ukázky různých grafických softwarů, s kterými je možné struktury zobrazovat, až po stručné představení základních simulací a jejich analýz. Práce si tedy neklade za cíl představit pokročilé simulační techniky či aplikovat molekulární dynamiku na nějaký specifický problém. Snahou bylo vytvořit spíše jednoduchý postup, který by byl pochopitelný pro další studenty biologických oborů a pomohl by jim při následném čtení podrobnějších textů a programových manuálů.

V práci byla popsána PDB databáze, ve které je možné najít velké množství proteinových struktur získaných experimentálně i pomocí počítačových predikcí. Databáze je velmi přehledná a dobře se s ní pracuje. Jsou tam stažitelná data, se kterými se dá dále pracovat v simulacích, nebo si je jen zobrazit v některém z grafických programů a detailně si prohlédnout jejich strukturu. To se může studentům hodit hlavně při studiu molekulární biologie, ale i když potřebují vytvořit obrázek proteinu do prezentaci nebo nějaké jiné práce. V této práci je představen jednoduchý internetový vizualizační nástroj Mol*Viewer a programy Chiméra, PyMol, a VMD. Program Mol*Viewer je dobře ovladatelný, jednoduchý a volně dostupný program pro zobrazení a úpravu proteinových struktur. Program Chiméra je jednoduchý a vcelku prostý, zdarma dostupný program, jeho slabinou je, že se na něm již moc nepracuje a nevyvíjí se. Program VMD je nejužitečnější, kvůli široké škále možností analýz, zobrazení, animací, renderování. S tímto programem jsem ve své práci pracoval nejvíc. Práce s ním je dobrá, i když na počátku není úplně lehké se s ním naučit a ovládat ho. Program PyMol je uživatelsky přívětivý a není těžké s ním pracovat, má spousty funkcí. Jedna z nevýhod je ale potřeba znát příkazy na jeho ovládání přes příkazový řádek, i když jej lze částečně ovládat i přes grafické rozhraní.

V další části práce jsou popsány samotné simulace, kdy nejprve stručně představujeme molekulárně-mechanický popis a pak se věnujeme praktickým otázkám, jako je stavba modelu,

jeho relaxace, ekvilibrace a následné molekulárně-dynamické simulace. Z mnoha dostupného softwaru se programům Amber a Gromacs, které jsou se k simulacím proteinů nejčastěji používají. Programový balík Amber je částečně placený, ale má i volně dostupnou verzi (dříve známou jako AmberTools), která na základní výpočty často stačí. Software Gromacs je open-source a z mého pohledu je srozumitelnější a lepší na ovládání. Simulace ukázané v této práci se mi o něco lépe dělali v Gromacsu než v Amberu, i když Amber byl v MetaCentru VO, kde jsem výpočty prováděl, značně rychlejší než Gromacs.

V poslední části jsou uvedeny některé základní analýzy, které se při simulacích proteinů často provádí. Věnujeme se hlavně strukturním analýzám jako je Ramachandranův diagram, histogramy různých geometrických parametrů, RMSD a RDF. Z dynamických analýz představujeme pouze RMSF, které lze dobře vizualizovat a dá člověku vhled to toho, které části proteinů jsou rigidní a které naopak flexibilní. Ve vědecko-výzkumné praxi se samozřejmě provádějí mnohem pokročilejší analýzy a simulace, např. při studiu reakčních mechanismů enzymů nebo výpočty volných energii charakterizující nejrůznější děje, těm se ale v této práci nevěnujeme. Doufáme ale, že text této práce může posloužit dalším studentům jako prvotní návod i motivace k dalšímu studiu a pokročilejším simulacím.

Seznam použité literatury

[1] R. K. Murray, D. K. Granner, P. A. Mayes, V. W. Rodwell: *Harperova Biochemie* . 2. vyd. Connecticut: Appleton & Lange, 1993, ISBN 80-85787-38-5.

[2] J. Vacík a kol. *Přehled středoškolské chemie*. 3. vyd. Praha: SPN, pedagogické nakladatelství, 1999, ISBN 80-72535-108-7.

[3] O. Nečas a kol. *Obecná biologie*. 4. vyd. Praha: H & H, 2003, ISBN 80-86022-46-3.

[4] R. Uribe, D. Jay: *A Review of Actin Binding Proteins: New Perspectives.* Mol. Biol. Rep. **36**, 121-125 (2009), DOI: 10.1007/s11033-007-9159-2.

[5] R. Dominguez, K. C. Holmes: *Actin Structure and Function*. Annu. Rev. Biophys. **40**, 169-186 (2011), DOI: 10.1146/annurev-biophys-042910-155359.

[6] PDB Protein Data Bank: https://www.rcsb.org/ (03 2023).

[7] V. Prosser a kol. *Experimentální metody biofyziky.* Praha, Academia, 1989, ISBN 80-200-0059-3.

[8] Chemistry World: https://www.chemistryworld.com/news/explainer-what-is-cryo-electron-microscopy/3008091.article (03 2023).

[9] A. Waterhouse, M. Bertolni, S. Bienert, G. Studer, G. Tauriello, R. Gumienny, F. T. Heer, T. A. P. de Beer, Ch. Rempfer, L. Bordoli, R. Lepore, T. Schwede: *SWISS-MODEL: Homology Modelling of Protein Structures and Complexes.* Nucl. Acids Res. **46**, W296-W303 (2018), DOI: 10.1093/narlgky427.

[10] M. T. Muhammed, E. Akin-Yalcin: *Homology Modeling in Drug Discovery: Overview, Current Applications, and Future Perspectives.* Chem. Biol. Drug. Des. **93**, 12-20 (2019), DOI: 10.1111/cdbb.13388.

[11] P. S. F. C. Gomes, D. E. B. Gomes, R. C. Bernardi: *Protein Structure Prediction in the Era of AI: Challenges and Limitations When Applying to In Silico Force Spectroscopy.* Front. Bioinform. **2**, 983306 (2022), DOI: 10.3389/fbinf.2022.983306.

[12] Nostrum Biodicovery: https://www.nostrumbiodiscovery.com/highlits/alphafold-code/ (03 2023).

[13] D. Shreiner, M. Woo, J. Neider, T. Davis. *OpenGL Průvodce programátora.* Brno, Computer Press, a.s., 2006, ISBN 80-251-1275-6.

[14] W. Humprey, A. Dalke, K. Schulten: *VMD: Visual Molecular Dynamics.* J. Mol. Graphics **14**, 33-38 (1996), DOI: 10.1016/0263-7855(96)00018-5.

[15] D. Sehnal, S. Bittrich, M. Deshpande, R. Svobodova, K. Berka, V. Bazgier, S. Velankar, S. K. Burley, J. Koca, A. S. Rose: *Mol*Viewer: Modern Web App for 3D Visualization and Analysis of Large Biomolecular Structures.* Nucl. Acids. Res. **49**, W431-W437 (2021), DOI: 10.1093/narlgkab314.

[16] Mol*Star Viewer: https://molstar.org/viewer/ (03 2023).

[17] E. F. Pettersen, T. D. Goddard, C. C. Huang, G. S. Couch, D. M. Greenblatt, E. C. Meng, T. E. Ferrin: *UCSF Chimera – A Visulization System for Exploratory Research and Analysis.* J. Comput. Chem. **25**, 1605-1612 (2004), DOI: 10.1002/jcc.20084.

[18] UCSF Chimera: https://www.cgl.ucsf.edu/chimera/ (03 2023).

[19] VMD – Visual Molecular Dynamics: https://www.ks.uiuc.edu/Research/vmd/ (03 2023).

[20] PyMol by Schrodinger: https://pymol.org/ (03 2023).

[21] D. A. Case, T. E. Cheatham, III, T. Darden, H. Gohlke, R. Luo, K. M. Merz, Jr., A. Onufriev, C. Simmerling, B. Wang, R. J. Woods: *The Amber Biomolecular Simulation Programs.* J. Comput. Chem. **26**, 1668-1688 (2005), DOI: 10.1002/jcc.20290.

[22] D. A. Case, et al.: Amber 2022, University of California, San Francisco (2022).

[23] H. J. C. Berendsen, D. van der Spoel, R. van Drunen: *GROMACS: A Message-Passing Parallel Molecular Dynamics Implementation.* Comp. Phys. Commun. **91**, 43-56 (1995), DOI: 10.1016/0010-4655(95)00042-e.

[24] M. J. Abraham, T. Murtola, R. Schulz, S. Pall, J. C. Smith, B. Hess, E. Lindahl: *GROMACS: High Performance Molecular Simulations through Multi-Level Porallelism from Laptops to Supercomputers.* Software X **1-2**, 19-25 (2015), DOI: 10.1016/j.softx.2015.06.001.

[25] R. Stones, N. Matthew: *Linux – začínáme programovat.* 1. vyd., Praha, Computer Press, 2000, ISBN: 80-7226-307-2.

[26] MetaCentrum VO: https://metavo.metacentrum.cz/ (04 2023).

[27] Putty: https://www.putty.org/ (04 2023).

[28] E. Lewars: *Computational Chemistry – Introduction to the Theory and Applications of Molecular and Quantum Mechanics.* Kluwer Academic Publishers, 2004, ISBN: 0-306-48391-2.

[29] Amber manual: https://www.ambermd.org (03 2023).

[30] Gromacs manual: https://www.gromacs.org (03 2023).

[31] R. B. Brooks, et al.: *CHARMM: The Biomolecular Simulation Program*. J. Comput. Chem. **30**, 1545-1614 (2009), DOI: 10.1002/jcc.21287.

[32] W. L. Jorgensen, D. S. Maxwell, J. Tirado-Rives: *Development and Testing of the OPLS All-Atom Force Field on Conformational Energetics and Properties of Organic Liquids.* J. Am. Chem. Soc. **118**, 11225-11236 (1996), DOI: 10.1021/ja9621760.

[33] Ch. Oostenbrink, A. Villa, A. E. Mark, W. F. van Gunsteren: *A Biomolecular Force Field Based on the Free Enthanlpy of Hydration and Solvation: The GROMOS Force-Field Parameter Sets 53A5 and 53A6.* J. Comput. Chem. **25**, 1656-1676 (2004), DOI: 10.1002/jcc.20090.

[34] W. R. P. Scott, P. H. Hunenberger, I. G. Tironi, A. E. Mark, S. R. Billeter, J. Fennen, A. E. Torda, T. Huber, P. Kruger, W. F. van Gunsteren: *The GROMOS Biomolecular Simulation Program Package.* J. Phys. Chem. A **103**, 3596-3607 (1999), DOI: 10.1021/jp984217f.

[35] J. C. Phillips, et al.: *Scalable Molecular Dynamics on CPU and GPU Architectures with NAMD.* J. Chem. Phys. **153**, 044130 (2020), DOI: 10.1063/5.0014475.

[36] E. Krieger, G. Vriend: *YASARA View – Molecular Graphics for All Devices – From Smartphones to Workstations.* Bioinformatics **30**, 2981-2982 (2014), DOI: 10.1093/bioinformatics/btu426.

[37] L. Backman: *Alpha-Actinin of the Chlorarchiniophyte Bigelowiella natans.* PeerJ **6**, e4288 (2018), DOI: 10.7717/peerj.4288.