VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2021 Adam Vybíhal

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY
DEPARTMENT OF CONTROL AND INSTRUMENTATION

AUTOMATIZACE VÝPOČTŮ GENOTYPOVÝCH DAT
AUTOMATION OF GENOTYPE DATA CALCULATIONS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Adam Vybíhal**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Radek Štohl, Ph.D.**
SUPERVISOR

BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Automatizační a měřicí technika**

Ústav automatizace a měřicí techniky

Student: Adam Vybíhal ***ID:*** 203541

Ročník: 3 ***Akademický rok:*** 2020/21

NÁZEV TÉMATU:

Automatizace výpočtů genotypových dat

POKYNY PRO VYPRACOVÁNÍ:

1. Seznamte se a popište vlastnosti softwarů pro použití genotypových dat pro zkoumání struktur populací zvířat.
2. Zpracujte návrh automatizace výpočtů při využití programů Structure, Structure Harvester, CLUMPP a distruct.
3. Vytvořte návrh GUI aplikace pro automatizaci výpočtů.
4. Vytvořte aplikaci pro automatizované spouštění programů Structure, Structure Harvester, CLUMPP a distruct.
5. Ověřte své řešení na vzorku vstupních dat.

DOPORUČENÁ LITERATURA:

Pritchard, J. K., Stephens, M., and Donnelly, P. (2000a). Inference of population structure using

multilocus genotype data. Genetics, 155:945–959.

Termín zadání: 8.2.2021 ***Termín odevzdání:*** 24.5.2021

Vedoucí práce: Ing. Radek Štohl, Ph.D.

doc. Ing. Václav Jirsík, CSc.
předseda rady studijního programu

ABSTRAKT

Cílem této práce je automatizace práce s programy pro zkoumání struktur populací pomocí genotypových dat. Konkrétně jsou popsány vlastnosti čtyř programů, *Structure*, *Structure Harvester*, *CLUMPP* a *distruct*. V rámci práce je pak navržen proces automatizace práce s těmito programy. Na bázi tohoto návrhu je následně vytvořena aplikace na platformě .NET, která uživateli usnadňuje práci se zmíněnými programy. Navržená aplikace poskytuje uživateli grafické rozhraní pro nahrávání vstupních souborů, zadávání potřebných parametrů, nastavení automatizovaného spouštění jednotlivých programů a zobrazení jejich výsledků.

KLÍČOVÁ SLOVA

genotypová data, struktura populace, Structure, Structure Harvester, CLUMPP, distruct

ABSTRACT

The goal of this paper is to automate work with programs for the study of population structure using genotype data. Specifically, we describe properties of four programs for this use, *Structure*, *Structure Harvester*, *CLUMPP* and *distruct*. There's proposed a process of automation of work with these programs within the paper. Based on proposed process, an application is then created on the .NET platform, which allows you to work with mentioned software. This application provides a graphical user interface for creating input files, entering the necessary parameters, setting up automatic launch of individual programs and displaying their results.

KEYWORDS

genotype data, population structure, Structure, Structure Harvester, CLUMPP, distruct

VYBÍHAL, Adam. *Automatizace výpočtů genotypových dat*. Brno, 2021, 50 s. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce: Ing. Radek Štohl, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Automatizace výpočtů genotypových dat“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno 23.5.2021 .....

podpis autora

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářske práce panu Ing. Radku Štohlovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno 23.5.2021 .....

podpis autora

Obsah

Seznam obrázků

Úvod

Tato práce se zabývá softwarem pro zpracování genotypových dat za účelem zkoumání struktury populace. Konkrétně se jedná o programy *Structure*, *Structure Harvester*, *CLUMPP* a *distruct*. Vlastnosti a funkcionalita těchto programů budou v práci popsány.

Zmíněné programy a jejich výsledky na sebe navazují a tvoří tak jeden řetězec zpracování genotypových dat. Na začátku tohoto řetězce má uživatel sesbíraná surová genotypová data N jedinců z C geografických lokací. Cílem celého procesu zpracování dat je pak rozřadit tyto jedince do K skupin, na základě genetických podobností. Uživatel pak může výsledky použít pro demonstraci struktury populace, identifikaci geneticky odlišných populací, přiřazování jednotlivců k populacím, a identifikaci migrantů a smíšených jednotlivců

Práce s danými programy nemusí být z hlediska uživatele vždy optimální. Některé z nich vyžadují pro plnohodnotnou analýzu několika spuštění po sobě, pro měnící se vstupní soubory. A ne všechny z těchto softwarů poskytují uživatelské rozhraní pro určitou automatizaci práce. Cílem této práce je tedy zpracovat návrh automatizace práce programů *Structure*, *Structure Harvester*, *CLUMPP* a *distruct* a návaznosti mezi nimi.

Hlavním cílem práce je pak návrh automatizace spouštění programů *Structure*, *Structure Harvester*, *CLUMPP* a *distruct* implementovat a vytvořit aplikaci. Aplikace by také měla poskytovat určité grafické uživatelské rozhraní, ve kterém bude možné nahrávat a vytvářet potřebné vstupní soubory, zobrazovat výstupy jednotlivých programů a hlavně tedy umožnit, podle voleb uživatele, automatizovaně spouštět dané programy. Naveržená aplikace pak bude otestována na určitém vzorku vstupních dat.

1 Software pro zpracování genotypových dat

Práce se zabývá automatizací zpracování genotypových dat při využití programů *Structure*, *Structure Harvester*, *CLUMPP* a *distruct*. V této kapitole jsou popsány vlastnosti zmíněných aplikací.

1.1 Program Structure

Zdrojový kód i spustitelné soubory tohoto softwaru jsou distribuovány pro různé platformy (aktuálně Mac, Windows, Linux, Sun). K dispozici je také grafické rozhraní, které uživateli poskytuje různé užitečné funkce, včetně jednoduchého zpracování výstupu. *Structure* je také možné spustit z příkazového řádku, namísto použití grafického rozhraní.

Tato kapitola obsahuje popis základních vlastností softwaru *Structure*, další podrobnosti lze najít v manuálu pro tento program (viz [1]).

1.1.1 Popis funkce

Software *Structure* implementuje metodu shlukování pro odvození struktury populace pomocí genetických markerů (metoda a její rozšíření jsou popsány ve zdrojích [2], [3] a [4]). Metoda se dá aplikovat pro demonstraci struktury populace, identifikaci geneticky odlišných populací, přiřazování jednotlivců k populacím, a identifikaci migrantů a smíšených jednotlivců.

Při použití této medoty předpokládáme model, kde existuje K populací (K může být neznámé). Populace jsou charakterizovány sadou alelových frekvencí na každém lokusu. Jednotlivci jsou k populaci jejich původu (nebo ke dvěma a více populacím, pokud jejich genotypy naznačují, že vzešli z více populací) přiřazeni na základě pravděpodobnosti.

Model nepředpokládá proces mutace, a lze jej použít pro většinu běžně používaných genetických markerů. Model předpokládá, že markery nejsou ve vazebné nerovnováze v subpopulacích, takže nezvládne zpracovat takové markery, které jsou extrémně blízko u sebe.

1.1.2 Vstupní soubor

Formát vstupních genotypových dat ukazuje Obrázek 1.1. Celá sada dat je uspořádána jako matice do jednoho souboru. Řádky reprezentují data o jednotlivcích, a sloupce představují lokusy. Uživatel má různé možnosti ohledně formátu vstupního souboru, většina údajů (kromě genotypů) je volitelná.

Níže jsou uvedeny jednotlivé prvky vstupního souboru (jsou-li přítomny, musí být v daném pořadí). Uživatel předem upřesňuje, která data jsou přítomna, buď v grafickém rozhraní, nebo (při spuštění z příkazového řádku) v samostatném souboru, *mainparams*. Uživatel taktéž předem sdělí informace o počtu jednotlivců a lokusů.

Label	*Pop*	*Flag*	*Location*	*Phen*	*ExtraCols*	*Loc 1*	*Loc 2*	*Loc 3*	....	*Loc L*
					M_1	M_2	M_3	....	M_L	
					r_1	r_2	r_3	....	r_L	
					-1	$D_{1,2}$	$D_{2,3}$	....	$D_{L-1,L}$	
$ID^{(1)}$	$g^{(1)}$	$f^{(1)}$	$l^{(1)}$	$\phi^{(1)}$	$y_1^{(1)},...,y_n^{(1)}$	$x_1^{(1,1)}$	$x_2^{(1,1)}$	$x_3^{(1,1)}$	....	$x_L^{(1,1)}$
$ID^{(1)}$	$g^{(1)}$	$f^{(1)}$	$l^{(1)}$	$\phi^{(1)}$	$y_1^{(1)},...,y_n^{(1)}$	$x_1^{(1,2)}$	$x_2^{(1,2)}$	$x_3^{(1,2)}$	....	$x_L^{(1,2)}$
					$p_1^{(1)}$	$p_2^{(1)}$	$p_3^{(1)}$	....	$p_L^{(1)}$	
$ID^{(2)}$	$g^{(2)}$	$f^{(2)}$	$l^{(2)}$	$\phi^{(2)}$	$y_1^{(2)},...,y_n^{(2)}$	$x_1^{(2,1)}$	$x_2^{(2,1)}$	$x_3^{(2,1)}$	....	$x_L^{(2,1)}$
$ID^{(2)}$	$g^{(2)}$	$f^{(2)}$	$l^{(2)}$	$\phi^{(2)}$	$y_1^{(2)},...,y_n^{(2)}$	$x_1^{(2,2)}$	$x_2^{(2,2)}$	$x_3^{(2,2)}$	....	$x_L^{(2,2)}$
					$p_1^{(2)}$	$p_2^{(2)}$	$p_3^{(2)}$	....	$p_L^{(2)}$	
....										
$ID^{(i)}$	$g^{(i)}$	$f^{(i)}$	$l^{(i)}$	$\phi^{(i)}$	$y_1^{(i)},...,y_n^{(i)}$	$x_1^{(i,1)}$	$x_2^{(i,1)}$	$x_3^{(i,1)}$	....	$x_L^{(i,1)}$
$ID^{(i)}$	$g^{(i)}$	$f^{(i)}$	$l^{(i)}$	$\phi^{(i)}$	$y_1^{(i)},...,y_n^{(i)}$	$x_1^{(i,2)}$	$x_2^{(i,2)}$	$x_3^{(i,2)}$	....	$x_L^{(i,2)}$
					$p_1^{(3)}$	$p_2^{(3)}$	$p_3^{(3)}$	....	$p_L^{(3)}$	
....										
$ID^{(N)}$	$g^{(N)}$	$f^{(N)}$	$l^{(N)}$	$\phi^{(N)}$	$y_1^{(N)},...,y_n^{(N)}$	$x_1^{(N,1)}$	$x_2^{(N,1)}$	$x_3^{(N,1)}$	....	$x_L^{(N,1)}$
$ID^{(N)}$	$g^{(N)}$	$f^{(N)}$	$l^{(N)}$	$\phi^{(N)}$	$y_1^{(N)},...,y_n^{(N)}$	$x_1^{(N,2)}$	$x_2^{(N,2)}$	$x_3^{(N,2)}$	....	$x_L^{(N,2)}$
					$p_1^{(L)}$	$p_2^{(L)}$	$p_3^{(L)}$	....	$p_L^{(1)}$	

Obr. 1.1: Formát vstupního data souboru do programu *Structure* [1].

Řádky vstupního souboru mohou obsahovat následující prvky:

1. **Názvy markerů** M (volitelné; textový řetězec): První řádek v souboru může obsahovat seznam identifikátorů pro všechny markery v datové sadě. Tento řádek obsahuje L textových řetězců, kde L je počet lokusů.
2. **Recesivní alely** r (Pouze u dat s dominantními markery; celé číslo): Tento řádek označuje recesivní alelu (pokud existuje) na každém markeru.
3. **Vzdálenosti mezi markery** D (volitelné; reálné číslo): Tento řádek v souboru specifikuje vzdálenosti mezi jednotlivými markery. Markery musí být v pořadí, v rámci skupin propojení. Když jsou po sobě jdoucí markery z různých vazebných skupin (např. různých chromozomů), měla by být hodnota tohoto pole rovna -1. Prvnímu markeru je taktéž přiřazena hodnota -1. Všechny ostatní vzdálenosti jsou nezáporné. Tento řádek obsahuje L reálných čísel.

4. **Informace o fázi vazby** p (volitelné; pouze u diploidních dat; reálné číslo v rozsahu [0,1]): Jedná se o jeden řádek L pravděpodobností, který následuje po údajích o genotypu každého jednotlivce.
5. **Genotypová data** (požadováno): Genotypové údaje o každém jednotlivci v daném vzorku dat.

Každý řádek jednotlivých dat obsahuje následující prvky:

1. **Label** (volitelné; textový řetězec): Textový řetězec, použitý pro označení daného jednotlivce.
2. **PopData** (volitelné; celé číslo): Celé číslo označující uživatelem definovanou populaci, ze které byl získán jednotlivec (sloupec může například označovat geografickou lokaci vzorkování jednotlivců).
3. **PopFlag** (volitelné; 0 nebo 1) Logický příznak, který označuje, zda se má při výpočtech použít položka PopData.
4. **LocData** (volitelné; celé číslo) Celé číslo, označující uživatelem definované místo vzorkování daného jedince.
5. **Phen** (volitelné; celé číslo) Číslo označující hodnotu fenotypu jednotlivce (informace o fenotypu *Structure* ve skutečnosti nepoužívá, položka je zde k umožnění bezproblémového spuštění programu *STRAT*, který se může používat souběžně se *Structure*).
6. **ExtraCols** (volitelné; textový řetězec): Pro uživatele může být výhodné přidat další data do vstupního souboru, ty budou však programem ignorována.
7. **Genotypová data** (požadováno; celé číslo): Každá alela na určitém lokusu, by měla být dána jedinečným celým číslem.

Informace o vstupním souboru jsou převzaty z manuálu programu *Structure* [1].

1.1.3 Spouštění z příkazové řádky

Řada parametrů programu je nastavovaná uživatelem před spuštěním. Parametry jsou ve dvou souborech (*mainparams* a *extraparams*), a čtou se pokaždé, když se program spustí. Soubor *mainparams* specifikuje vstupní formát datového souboru a základní parametry pro činnost programu. Soubor *extraparams* určuje širší možnosti programu. Všechny hodnoty v *mainparams* se musí nastavit pro určitý vstupní soubor, zatímco výchozí hodnoty v souboru *extraparams* jsou většinou v pořádku pro libovolný vstup, a přenastavovat se nemusí. Níže je popsáno několik základních parametrů, které specifikuje uživatel v souboru *mainparams*.

Základní parametry programu:

- **MAXPOPS** (celé číslo): Počet předpokládaných populací K pro konkrétní běh programu.

- **BURNIN** (celé číslo): Počet iterací simulace, než začne sběr dat, aby se minimalizoval účinek počáteční konfigurace.
- **NUMREPS** (celé číslo): počet iterací simulace, ze kterých se sbírají data, pro získání přesných odhadů parametrů.

Formát vstupního data souboru: Řada parametrů, týkajících se formátu vstupního souboru indikují, zda jsou ve vstupním souboru přítomny určité typy dat (data jsou popsány v kapitole 1.1.2). Takovými parametry jsou: LABEL, POPDATA, POPFLAG, LOCDATA, PHENOTYPE, EXTRACOLS, MARKERNAMES (M), RECESSIVEALLELES (r), MAPDISTANCES (D). Tyto parametry nabývají hodnot 0 či 1. Dalšími parametry jsou například:

- **NUMINDS** (celé číslo) Počet jedinců v datovém souboru.
- **NUMLOCI** (celé číslo) Počet lokusů v datovém souboru.
- **PLOIDY** (celé číslo) Ploidie organismu. Výchozí hodnota je 2 (diploidní).
- **MISSING** (celé číslo) Hodnota přidělená chybějícím genotypovým datům. Hodnota musí být celé číslo a nesmí se objevit jinde v souboru dat. Výchozí hodnota je -9.
- **ONEROWPERIND** (Boolean) Nabývá hodnot 0 či 1. Indikuje, zda-li jsou data pro každého jednotlivce uspořádána do jednoho řádku.

1.1.4 Výstup programu

	Label	(%Miss)	Pop:	Inferred clusters	(and 90% probability intervals)	
1	17	(0)	2 :	0.977 0.023	(0.829,1.000)	(0.000,0.171)
2	1219	(7)	2 :	0.997 0.003	(0.988,1.000)	(0.000,0.012)
3	1223	(0)	2 :	0.833 0.167	(0.003,1.000)	(0.000,0.997)
4	1329	(7)	1 :	0.005 0.995	(0.000,0.020)	(0.980,1.000)
5	15	(0)	3 :	0.006 0.994	(0.000,0.016)	(0.984,1.000)

Obr. 1.2: Formát výstupního souboru *Structure* [1].

Na Obrázku 1.2 je znázorněn formát výstupní matice Q, pro takový případ, kdy nejsou známé předchozí informace o populacích ve vstupních datech. Parametr K zde byl nastaven na hodnotu 2. Výstupní data mají výpovědní charakter následující (pro řádek 1): Popis jedince (LABEL; ze vstupního souboru) = 17; procento chybějících údajů o tomto jedinci = 0%; populace přiřazená uživatelem = 2; odhadované členství v klastrech 1 a 2 = $(0.977; 0.023)$; intervaly odhadovaného členství v klastrech na hladině významnosti 90% jsou (0,829,1 000), respektive (0,000,0,171).

Když je program spuštěn s předchozími informacemi o populaci, výstup je trochu jiný (Obrázek 1.3). První sloupec výsledků („Pop:“) ukazuje pravděpodobnost, že

dotyčný jedinec je správně přiřazen k dané populaci. Následné sloupce zobrazují pravděpodobnosti, že pochází z jiných populací nebo v nich má předka.

Pro řádek 1 tedy platí: jedinec 17 (má 0% chybějících dat) pochází z předpokládané populace (označnou číslem 2) s pravděpodobností 0.998. Existuje (přibližně) nulová pravděpodobnost, že tento jedinec má nedávný původ v populaci 1 (pravděpodobnost 0.000, že jedinec je z populace 1, a 0.001, že má jednoho rodiče z populace 1). Totéž pro něho obdobně platí s populací 3.

```
 Label (%Miss) Pop :
1  17    (0) 2 :  0.998 | Pop 1:  0.000 0.001 | Pop 3:  0.000 0.000
2 1219   (7) 2 :  1.000 | Pop 1:  0.000 0.000 | Pop 3:  0.000 0.000
3 1223   (0) 2 :  0.612 | Pop 1:  0.000 0.000 | Pop 3:  0.004 0.383
4 1329   (7) 1 :  1.000 | Pop 2:  0.000 0.000 | Pop 3:  0.000 0.000
5  15    (0) 3 :  0.999 | Pop 1:  0.000 0.001 | Pop 2:  0.000 0.000
```

Obr. 1.3: Formát výstupního souboru *Structure* s předem známými populacemi [1].

1.2 Program Structure Harvester

Structure Harvester zpracovává výstup programu *Structure*, generuje soubory (s příponami *.indfile* a *.popfile*) pro další použití s aplikací CLUMPP (1.3) a je-li to možné, provede Evannovu metodu (více o Evannově metodě v [5]).

Dostupné jsou dvě verze tohoto programu - webová aplikace a skript pro Python. Webová verze poskytuje jednoduché grafické uživatelské rozhraní pro správu vstupních a výstupních souborů.

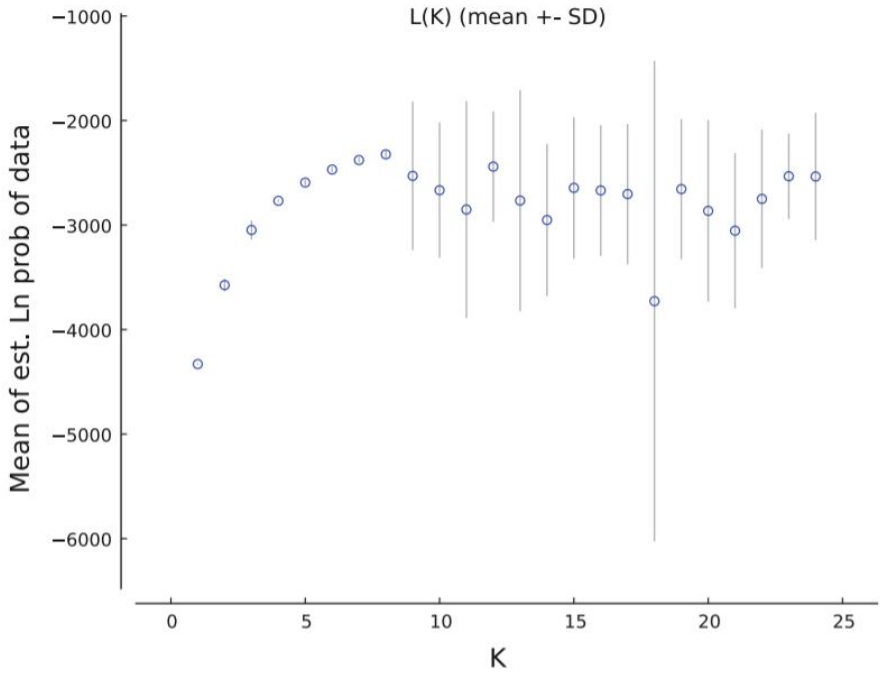
1.2.1 Popis funkce programu

Pokud uživatel pracuje s webovou aplikací, pak na domovské stránce nahrává komprimovaný archiv výstupních souborů z programu *Structure*. *Structure Harvester* provede analýzy a vrátí výsledky do webového prohlížeče. Pro každou hodnotu K (pro kterou byla provedena *Structure* analýza), jsou vytvořeny soubory pro další použití programem *CLUMPP*. Webová verze též generuje grafy pro průměrné hodnoty pravděpodobností K klastrů, ve zpracovaných datech, a pro výsledky Evannovy metody (příklad grafu pravděpodobností K viz Obrázek 1.4). Veškerý vygenerovaný obsah je komprimován do jednoho archivu pro stažení. [6]

Jestliže uživatel pracuje s Python verzí, přímo na svém počítači, pak spouští program z příkazové řádky, kde ke skriptu zadává argumenty, upřesňující práci, která má být provedena. Argumenty jsou následující:

- **--dir=PATH/TO/DIR** Tento argument upřesňuje cestu ke složce, kde se nachází výstupní soubory *Structure*.
- **--out=PATH/TO/DIR** Tento argument upřesňuje cestu ke složce, kam se mají vygenerovat výsledky programu *Structure Harvester*.
- **--evanno** Pokud uživatel přidá tento argument, pak (je-li to možné) je provedena Evannova metoda, jejíž výsledky jsou uloženy do textového souboru.
- **--clumpp** Tento argument dává programu najevo, že má vygenerovat pro každé K, odpovídající soubory do navazujícího programu *CLUMPP*.

Algoritmus, který určuje, zda lze provést Evannovu metodu, vyžaduje, aby byly analyzovány alespoň tři po sobě jdoucí hodnoty K, každá alespoň o 3 opakováných běžích programu *Structure*, se stejnými vstupními daty, a aby standardní směrodatná odchylka hodnot logaritmické pravděpodobnosti napříč všemi hodnotami K byla nenulová. K tomu, že je daná směrodatná odchylka nulová, může někdy dojít při malém počtu opakovaných běhů, nebo když náhodou všechny běhy vygenerují stejné odhadované hodnoty pravděpodobností. Jsou-li podmínky splněny, pak program provede Evannovu metodu pro detekci počtu klastrů K, které nejlépe odpovídají datové sadě. [6]



Obr. 1.4: Graf průměrných hodnot pravděpodobností $L(K)$ a rozptylu hodnot K [6].

1.3 Program CLUMPP

V populačně-genetických shlukových algoritmech, jako používá například výše zmiňovaný *Structure* (1.1), jsou jednotlivé multilokusové genotypy často rozdělovány do sady klastrů pomocí takových postupů, které zahrnují stochastické simulace. Výsledkem pak je, že opakovaný běh klastrové analýzy, o stejných vstupních údajích, může přinést odlišné odhady koeficientů, jenž přiřazují jednotlivce k určitému klastru, i když byly použity stejné počáteční podmínky. Neshody výsledných řešení napříč opakovanými běhy mají dva hlavní zdroje:

1. „přepínání štítků“, jenž je způsobeno libovolným označováním jednotlivých klastrů
2. „skutečná multimodalita“, skutečně odlišná řešení napříč běhy [7]

1.3.1 Popis funkce programu

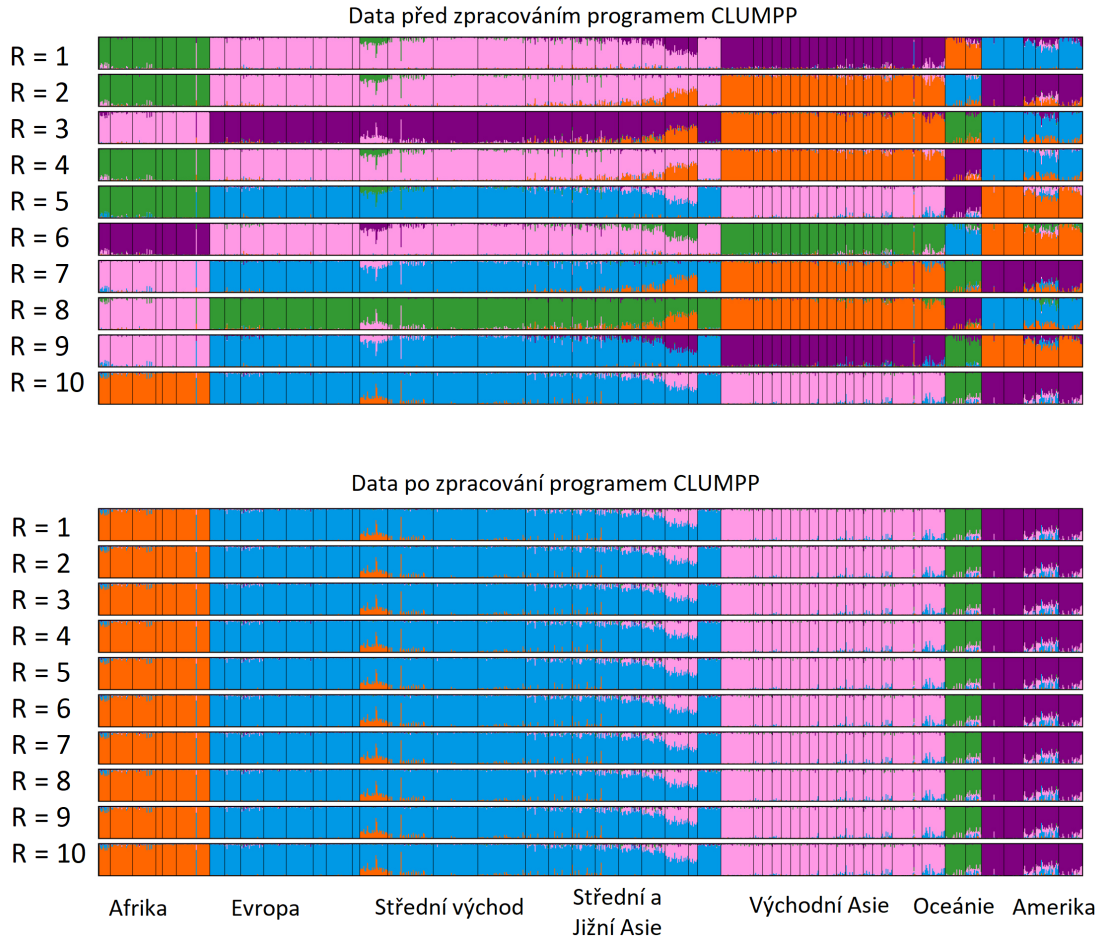
Bez ohledu na zdroj rozdílů ve výsledcích shlukové analýzy, je nutná nějaká metoda pro zpracování výsledků ze všech kopií analýzy. Proto byl vytvořen software *CLUMPP*, který implementuje tři různé algoritmy - FullSearch, Greedy a LargeKGreedy (konkrétní podrobnosti o principech algoritmů viz [7]) pro vyhledávání optimálního seřazení dat z R kopií klastrových analýz, o stejných vstupních datech. *CLUMPP* bere jako vstup odhadované matice koeficientů členství jednotlivců v klastru z R běhů programu pro shlukovou analýzu. Jako výstup vrací matice stejného typu, uspořádané tak, aby všechny kopie měly co největší shodu [7].

Vstupní soubor je podobný tomu výstupnímu z programu *Structure* (viz 1.1.4). Výstupní soubor aplikace *CLUMPP* pak může být dále použit jako vstup do vizualizačního programu *distruct* (1.4).

Obrázek 1.5 ukazuje příklad, jakým způsobem *CLUMPP* zpracuje a uspořádá data. Grafy dat před zpracováním, přímo znázorňují výstup ze *Structure*. Konkrétně ukazují populační strukturu odhadovanou pomocí 10-ti běhů programu *Structure*, se vstupními daty o 1056 jedincích z 52 lidských populací [9]. V grafu je každý jednotlivec reprezentován svislou čarou, rozdělenou do 5 barev, které představují jeho odhadované členské podíly v K= 5 klastrech. Černé svislé čáry oddělují jednotlivce z různých populací. Data byla vizualizována pomocí programu *distruct*.

1.3.2 Vstupní soubory

Aplikace *CLUMPP* při spouštění čte potřebné parametry ze souboru (*paramfile*). Program také načítá soubor obsahující matice Q (generované pomocí *Structure Harvester*). V některých případech, je též vyžadován soubor, obsahující permutace běhů (*permutationfile*).



Obr. 1.5: Vizualizace výstupních dat programu *Structure* před a po zpracování pomocí aplikace *CLUMPP* [9]

Jak bylo zmíněno, jedním ze vstupních souborů je ***paramfile***, který obsahuje parametry pro správný chod programu *CLUMPP*. Příklady hlavních parametrů:

- **DATATYPE** (celé číslo): Typ dat, které mají být použity. Je-li hodnota DATATYPE = 0, očekává CLUMPP matice Q jednotlivců, ze souboru *indfile*, a pokud DATATYPE = 1, očekává, že bude číst Q matice populací, z *popfile*.
- **INDFILE** (textový řetězec): název (cesta k) souboru *indfile*.
- **POPFILE** (textový řetězec): název (cesta k) souboru *popfile*.
- **OUTFILE** (textový řetězec): název (cesta k) výstupnímu souboru *outfile*. Obsahuje průměrnou matici Q ze všech běhů v optimálním rozložení.
- **MISCFILE** (textový řetězec): název (cesta k) výstupnímu souboru *miscfile*. Obsahuje nastavení parametrů pro aktualní běh programu.
- **K** (celé číslo): Počet klastrů.
- **C** (celé číslo): Počet jednotlivců nebo populací.

- **R** (celé číslo): Počet matic Q, resp. běhů R.
- **M** (1, 2 nebo 3): Metoda, která má být použita. (1 = FullSearch, 2 = Greedy, 3 = LargeKGreedy)

Více podrobností o vstupních souborech se nachází v manuálu programu [8].

1.3.3 Spouštění programu

CLUMPP se spouští z příkazového řádku pomocí příkazu *CLUMPP paramfile* (v systémech Unix a MacOS X pak "./CLUMPP paramfile"), kde parametr *paramfile* odkazuje na název, případně cestu k souboru s potřebnými parametry (viz 1.3.2). Pokud po zadání příkazu "CLUMPP"není zadán žádný parametr, *CLUMPP* zkusí vyhledat soubor s názvem "paramfile". Jestliže tento soubor není nalezen, *CLUMPP* ukončí činnost a hlásí chybovou zprávu [8].

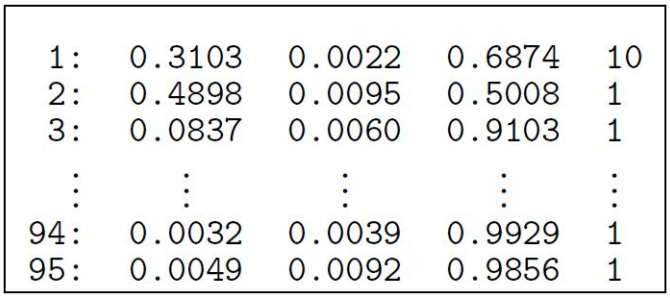
1.3.4 Výstupní soubory

CLUMPP má řadu výstupních souborů [8]. Pro účely této práce bude popsán pouze jeden výstupní soubor *outfile*, který obsahuje vypočtenou a seřazenou matici Q.

Pokud byl ve vstupním souboru *paramfile* nastaven parametr DATATYPE = 0, bude *outfile* obsahovat matici s $(K + 5)$ sloupci a C řádky. Prvních 5 sloupců je stejných jako v souboru *indfile*. Druhý sloupec tvoří identifikátory jedinců. Sloupce 6 až $(K + 5)$ tvoří průměrnou matici Q.Tato matice se počítá jako průměr ze všech matic Q ve vstupním souboru poté, co byly sloupce srovnány algoritmem zvoleným pro běh programu.

Jestliže byl parametr DATATYPE nastaven na hodnotu 1, výstup bude obsahovat taktéž jednu matici, ale s $(K+2)$ sloupci a C řádky. První sloupec je identifikátor populace, následujících K sloupců tvoří matici Q (tato matice se počítá stejně, jako v předešlém případě), a poslední sloupec označuje počet jednotlivců v každé populaci.

Na Obrázku 1.6 je zobrazen výstup programu pro vstupní data: DATATYPE = 1, $C = 95$, $K = 3$ a $R = 9$ [8].



1:	0.3103	0.0022	0.6874	10
2:	0.4898	0.0095	0.5008	1
3:	0.0837	0.0060	0.9103	1
⋮	⋮	⋮	⋮	⋮
94:	0.0032	0.0039	0.9929	1
95:	0.0049	0.0092	0.9856	1

Obr. 1.6: Příklad výstupního souboru programu *CLUMPP* [8]

1.4 Program *distruct*

1.4.1 Popis funkce programu

Program *distruct* je určen pro použití s programem *Structure* (1.1) (výstup ze *Structure*, je vstupem pro *distruct*). Pro každého jedince *distruct* vyžaduje pouze odhady jeho členských koeficientů v K klastrech a jeho předdefinovaný identifikátor skupiny, například místo sběru vzorků. V zásadě mohou být potencionálním vstupem do programu členské koeficienty, získané jakýmkoli přístupem.

Program je spouštěn z příkazové řádky v Unixu nebo Windows a vytváří výstupní soubor ve formátu PostScript (ukázka vizualizace dat pomocí programu *distruct* je na obrázku 1.5). Při používání je možné ovládat také podrobnosti, jako je např. systém barev, či pořadí tisku skupin a jednotlivců [10].

1.4.2 Vstupní soubory

Data, která mají být vykreslena, jsou specifikována v textových souborech, odvozených z výstupu programu *Structure* (celý výstupní soubor se nepoužívá). Program vezme soubor s Q maticí populací (vyžadováno), a samostatný soubor s Q maticí jednotlivců (též vyžadováno). Tyto soubory můžeme získat buď manuálním kopírováním přímo z výstupu *Structure*, nebo můžeme použít přímo výstupní soubory programu *CLUMPP*.

Nastavení programu jsou specifikována v souboru *drawparams*, i když některá mohou být uvedena s argumenty příkazového řádku. Hlavní nastavení se týká vstupních souborů, množství dat a položek k tisku. Výběr parametrů:

- **INFILE_POPQ** (textový řetězec): Název vstupního souboru pro Q matici populací. Tento soubor je povinný.

- **INFILE_INDIVQ** (textový řetězec): Název vstupního souboru pro matici Q jednotlivců.
- **OUTFILE** (textový řetězec): Název výstupního souboru ve formátu PostScript. Existující soubor s tímto názvem bude přepsán.
- **K** (celé číslo): Počet klastrů.
- **NUMPOPS** (celé číslo): Počet populací v Q matici populací.
- **NUMINDS** (celé číslo): Počet jednotlivců v Q matici jednotlivců.
- **PRINT_INDIVS** (Boolean): 1 pro vykreslení Q matici jednotlivců, 0 pro vykreslení Q matici populací.

Více podrobností ke vstupním souborům a parametrům je uvedeno v manuálu k programu *distruct* [11].

1.4.3 Spouštění programu

Jak bylo zmíněno, program je spouštěn pomocí příkazové řádky. Pomocí argumentů je umožněno uživateli zadat určité parametry přímo z příkazového řádku. Hodnoty zadané pomocí argumentů (kromě „-d“), přepíší hodnotu v souboru *drawparams*. Argument „-d“ umožňuje použít jiný soubor parametrů namísto *drawparams*, který se nachází ve stejné složce jako program. Hodnoty zadané v souboru parametrů můžeme přepisovat následujícími argumenty:

- **-d** (drawparams): Přečte zadaný soubor s parametry.
- **-K**: Změní počet klastrů K.
- **-M**: Změní počet předdefinovaných populací.
- **-N**: Změní počet jednotlivců.
- **-p**: (vstupní soubor) Změní soubor, ze kterého má program číst matici populací Q.
- **-i**: (vstupní soubor) Změní soubor, ze kterého má program číst matici jednotlivců Q [11].

2 Návrh automatizace zpracování dat

V kapitole 1.1 byl popsán program *Structure*. Zopakujme, že tento software bere jako vstup matici genotypových dat jedinců z různých populací. *Structure* daná genotypová data zpracuje, a jedince rozdělí, na základě pravděpodobnosti, do K klastrů (kde K volí uživatel před během programu). Matici genotypových dat, ve vstupním souboru, a možnosti práce programu, specifikuje uživatel ve dvou parametrických souborech, *mainparams* a *extraparams*. Před spuštěním aplikace *Structure* tedy potřebujeme, aby uživatel vybral vstupní soubor a nastavil parametry pro běh programu. Podle těchto informací, můžeme následně vytvořit zmíněné dva parametrické soubory.

Je-li vybrán vstupní soubor a vytvořené parametrické soubory, pak uživatel ještě musí nastavit rozsah hodnot K, pro které má aplikace *Structure* zpracovat data. Pro plnohodnotnou analýzu, je též potřeba provést několik iterací R, přes všechna zvolená K. Jak bylo zmíněno v kapitole 1.2.1, algoritmus v navazujícím programu *Structure Harvester*, potřebuje alespoň tři, po sobě jdoucí hodnoty K, o nejméně 3 iteracích. Po zvolení rozsahu K a počtu iterací R, můžeme začít automatizovaně spouštět program *Structure*.

Jakmile jsou dostupné všechny výstupní soubory z aplikace *Structure*, můžeme spustit navazující *Structure Harvester* (v této práci budeme používat verzi Python skriptu, aby pro běh navrhované aplikace nebylo nutné připojení k internetu). Pro běh této aplikace není třeba, aby uživatel specifikoval nějaké parametry, *Structure Harvester* stačí odkázat na složku s výsledky předešlého programu *Structure*, a ty budou zpracovány. Výstupem aplikace *Structure Harvester* (Python skript), je pak textový soubor se shrnutím provedené práce, vytvořené vstupní soubory do navazujícího softwaru *CLUMPP*. Je-li možné provést Evannovu metodu, pak skript generuje ještě textový soubor se shrnutím této metody. Jestliže uživatel potřebuje tento výstupní soubor, ale metodu nebylo možno provést, musí pak opakovat práci programu *Structure*, o větším rozsahu K, respektive více iteracích R.

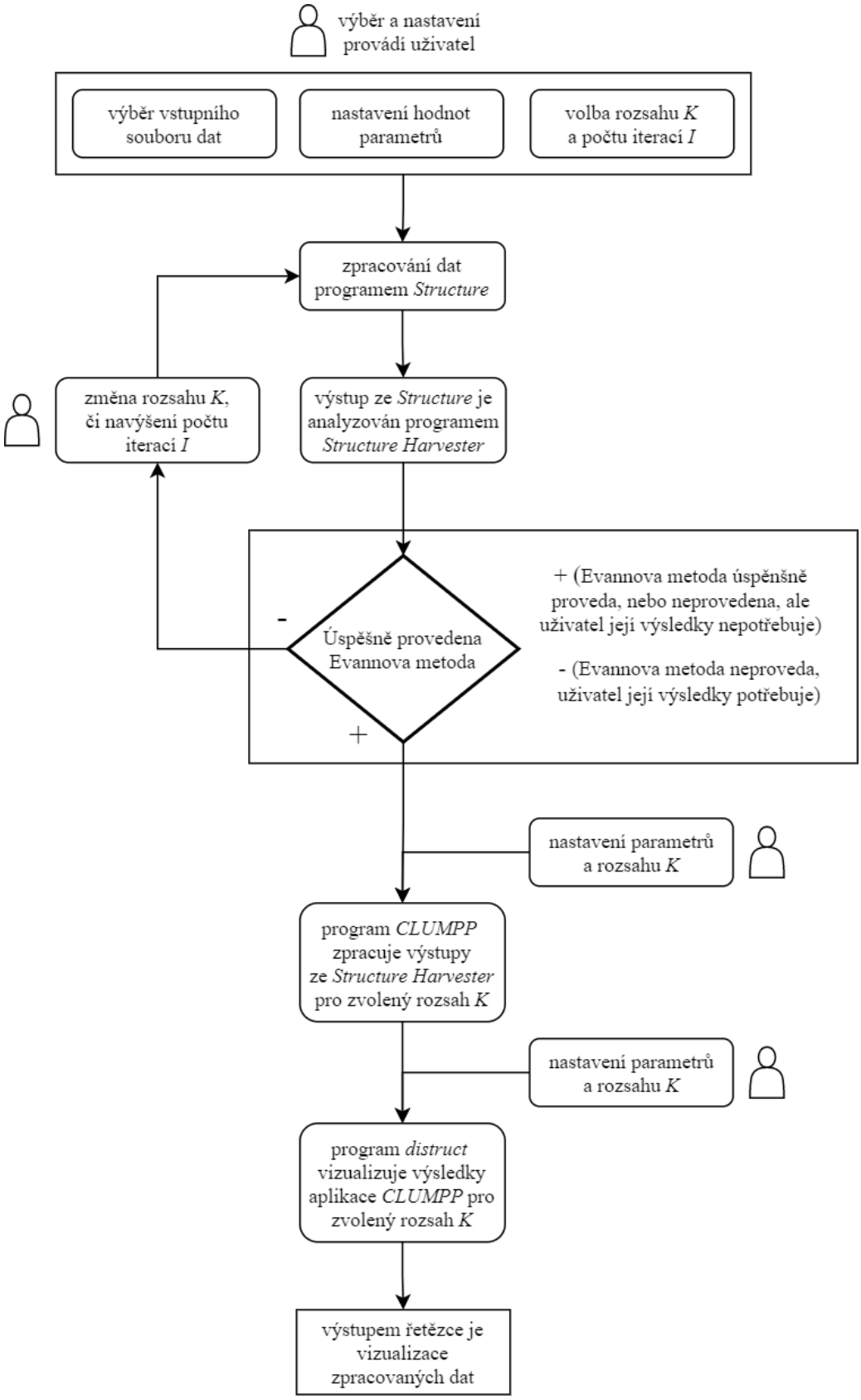
Po programu *Structure Harvester*, následuje aplikace *CLUMPP*. Před během této aplikace, je opět nutná interakce uživatele, ten musí specifikovat parametry nutné pro běh programu. Po zadání parametrů a odvození známých parametrů (konkrétně počet jednotlivců a počet iterací R), z prací předešlých programů, můžeme vytvoři parametrické soubory pro *CLUMPP*. Pak uživateli zbývá pouze zadat rozsah K (zde je nutné uživatele omezit na určitý rozsah K, který je daný volbami uživatele, před spuštěním programu *Structure*), a můžeme automatizovaně spouštět aplikaci *CLUMPP*. *CLUMPP* postupně zpracuje výstupy programu *Structure Harvester*, které jsou pojmenovány podle hodnoty K jako $K1$.*indfile*, $K2$.*indfile*, ... (resp. $K1$.*popfile*, $K2$.*popfile*, ...). Výstupem práce programu *CLUMPP* jsou pak soubory

$K1.indq$, $K2.indq$, ... (resp. $K1.popq$, $K2.popq$, ...).

Výsledky aplikace *CLUMPP* pak v poslední řadě můžeme vizualizovat pomocí programu *distruct*. Jak bylo zmíněno v kapitole 1.4.2, *distruct* vyžaduje pro svou práci parametry, znovu ve formě parametrického souboru. Parametry počet populací, počet jednotlivců a počet klastrů K můžeme, jako v případě programu *CLUMPP*, odvodit z běhů předešlých programů. Uživatel pak specifikuje rozsah K, pro který chce získat vizualizace, a parametry, týkající se vzhledu vizualizace. Poté můžeme vytvořit parametrické soubory a spouštět automatizovaně program *distruct* pro rozsah K. Výstupem jsou vizualizace ve formátu PostScript.

Diagram návrhu automatizace zpracování genotypových dat pomocí zmíněných programů je na obrázku 2.1. Podle tohoto diagramu můžeme zjednodušeně popsat návrh v následujících bodech:

1. Uživatel nahraje vstupní soubor dat pro program *Structure*. Nastaví hodnoty potřebných parametrů pro tento program, zvolí rozsah hodnot K a počet iterací R přes všechna K.
2. Program *Structure* je automatizovaně spouštěn, generuje výstupní soubory.
3. Výstpní soubory programu *Structure* jsou zpracovány programem *Structure Harvester*. Ten generuje vsutpní soubor pro aplikaci *CLUMPP*.
4. V případě, že nebyla provedena Evannova metoda a uživatel potřebuje znát její výsledky, je třeba opakovat analýzu programu *Structure*, s navýšeným počtem iterací R, popřípadě jiným rozsahem K.
5. Uživatel nastaví potřebné parametry pro program *CLUMPP*, a vybere rozsah hodnot K. *CLUMPP* je pak automatizovaně spouštěn, se vstupními soubory pro zvolený rozsah K. *CLUMPP* generuje vstupní soubory do programu *distruct*.
6. Uživatel nastaví potřebné parametry pro program *distruct*, a vybere rozsah hodnot K. *distruct* je pak automatizovaně spouštěn, se vstupními soubory pro zvolený rozsah K. Výsledkem jsou vizualizace ve formátu PostScript.



Obr. 2.1: Diagram návrhu automatizace zpracování genotypových dat.

3 Aplikace pro automatizované zpracování genotypových dat

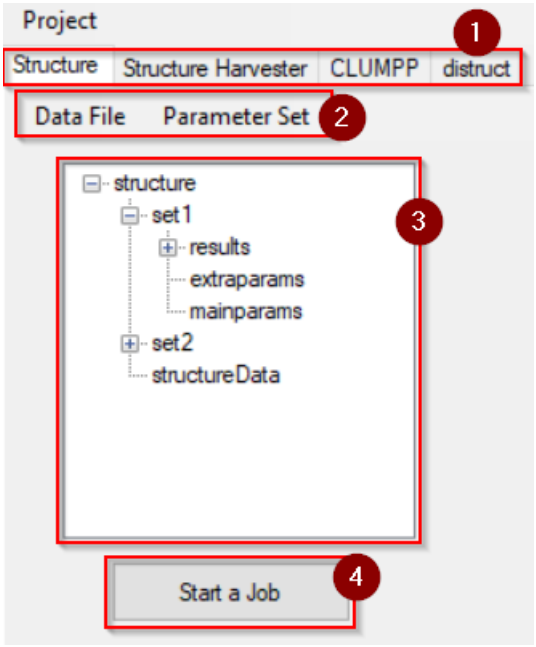
Cílem této práce je vytvoření aplikace pro automatizované spouštění programů *Structure*, *Structure Harvester*, *CLUMPP* a *distruct*. Aplikace má také uživateli poskytnout grafické rozhraní, které umožní nahrávání vstupních souborů, vytváření potřebných parametrických souborů pro korektní práci jednotlivých programů a zobrazení jejich výstupů. Aplikace bude implementována v rozhraní *C# .NET* na modelu *WindowsForms*. Tato kapitola se věnuje navržené aplikaci, popisuje grafické uživatelské rozhraní a její funkcionalitu.

3.1 Grafické uživatelské rozhraní

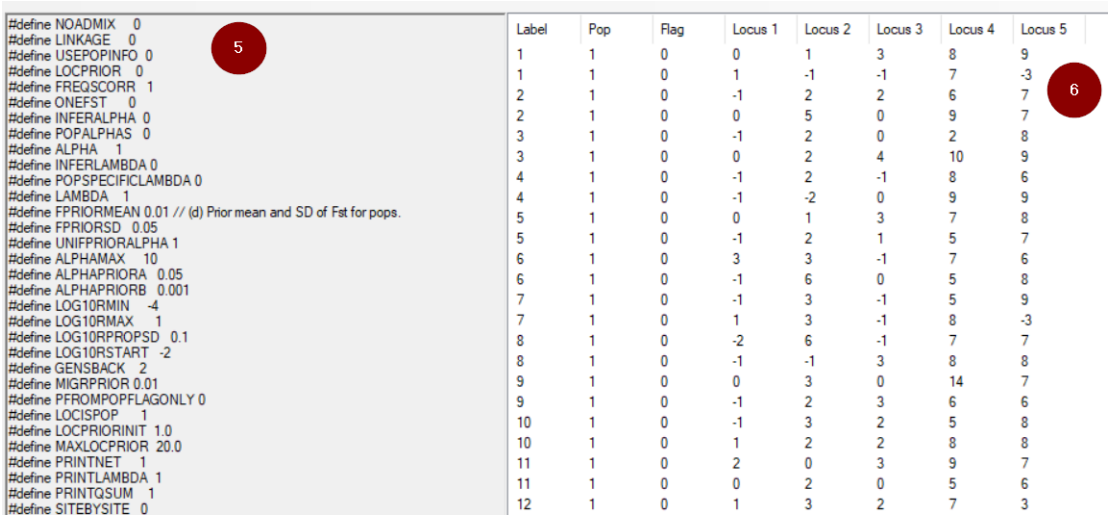
Na obrázcích 3.1 a 3.2 je vyzrobrazeno grafické uživatelské rozhraní navržené aplikace. Komponenty mají následující funkci:

1. Záložky, pro volbu programu, který chce uživatel obsluhovat.
2. Menu dané záložky. Na obrázku 3.1 je vybrána záložka pro program *Structure*. V případě *Structure* je možné přes možnost *Data File* v menu nahrát, a popsat, vstupní soubor (u ostatních programů aplikace přiřazuje automaticky vstupní soubory z předešlých programů). Ve všech záložkách pro programy, je možné přes položku v menu, vytvářet novou sadu parametrů, případně ji později upravit či smazat (v případě obrázku je to přes položku *Parameter Set*).
3. Komponenta *TreeView*. Slouží k zobrazení složky, určené pro ukládání souborů spjatých s daným programem. Uživatel tak může přímo z aplikace kontrolovat například, jestli daný program správně vygeneroval všechny výstupní soubory do složky, a když klikne na dané soubory, může si zobrazit jejich obsah. Tato komponenta je tedy na všech záložkách pro dané programy.
4. Tlačítko pro spouštění automatizované práce daného programu.
5. Komponenta programu, zobrazující obsah, uživatelem zvoleného souboru v komponentě *TreeView*.
6. Tato komponenta se nachází pouze na záložce *Structure*. Slouží k zobrazení naformátovaného vstupního souboru genotypových dat.

Při tvorbě sady parametrů pro programy, se uživateli otevírají dialogová okna. V okně je pak možné nastavit hodnoty veškerých parametrů pro daný program. Po stisknutí tlačítka *OK* se pak vytvoří parametrické soubory s nastavenými hodnotami. Ukázka takového okna, konkrétně pro nastavování parametrů pro program *Structure*, je na obrázku 3.3.

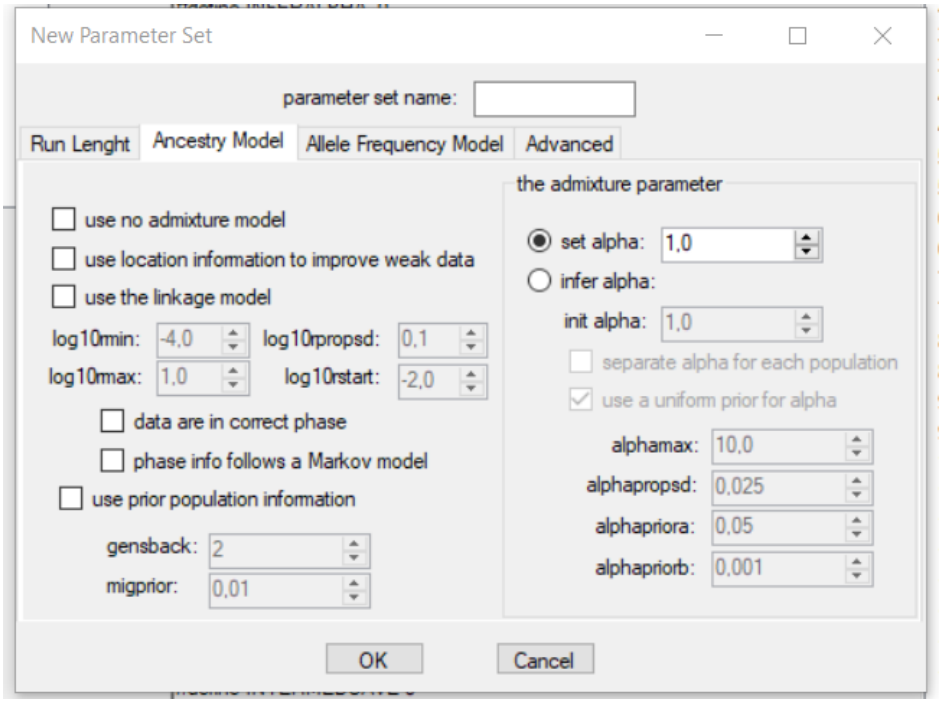


Obr. 3.1: Grafické uživatelské rozhraní navržené aplikace (ovládací prvky).



Label	Pop	Flag	Locus 1	Locus 2	Locus 3	Locus 4	Locus 5
1	1	0	0	1	3	8	9
1	1	0	1	-1	-1	7	-3
2	1	0	-1	2	2	6	7
2	1	0	0	5	0	9	7
3	1	0	-1	2	0	2	8
3	1	0	0	2	4	10	9
4	1	0	-1	2	-1	8	6
4	1	0	-1	-2	0	9	9
5	1	0	0	1	3	7	8
5	1	0	-1	2	1	5	7
6	1	0	3	3	-1	7	6
6	1	0	-1	6	0	5	8
7	1	0	-1	3	-1	5	9
7	1	0	1	3	-1	8	-3
8	1	0	-2	6	-1	7	7
8	1	0	-1	-1	3	8	8
9	1	0	0	3	0	14	7
9	1	0	-1	2	3	6	6
10	1	0	-1	3	2	5	8
10	1	0	1	2	2	8	8
11	1	0	2	0	3	9	7
11	1	0	0	2	0	5	6
12	1	0	1	3	2	7	3

Obr. 3.2: Grafické uživatelské rozhraní navržené aplikace (prvky pro zobrazení dat).



Obr. 3.3: Dialog pro nastavení parametrů pro program *Structure*.

3.2 Struktura projektu

Pro daný projekt navržená aplikace uživateli vytvoří složku, a v té automaticky generuje další složky, pro uchování výsledků jednotlivých aplikací. Složka projektu také obsahuje soubory, ze kterých se zpětně načítají data, když chce uživatel projekt znovu otevřít. Při vytvoření sady parametrů, pro daný program, je generována složka s názvem sady, v adresáři, který je tomuto programu určen. Do této složky se poté generují parametrické soubory a výstupy daného programu. Adresář pro výsledky programu *Structure* navíc obsahuje kopii uživatelem nahraného vstupního data souboru (*structureData*).

3.3 Funkcionalita aplikace

Funkcionalita aplikace vychází z návrhu automatizace zpracování genotypových dat, popsaného v kapitole 2.

3.3.1 Nahrávání vstupního souboru pro program Structure

V první řadě, musí uživatel zvolit vstupní soubor s genotypovými daty. Průběh tohoto procesu, v naší aplikaci, ukazuje diagram na obrázku 3.4. Uživateli se otevře

dialog, kde volí cestu ke vstupnímu souboru a specifikuje jeho formát, tedy určuje, jaká data se v souboru mají vyskytovat. Aby nedošlo k chybě, jsou z popisu vstupních dat vypočteny rozměry matice, kterou má vstupní soubor obsahovat, a jsou porovnány s rozměry matice, kterou vstupní soubor skutečně obsahuje. Jestliže spolu tyto rozměry nesouhlasí, je uživatel upozorněn, aby zkontroloval a opravil zadané hodnoty, případně změnil vstupní soubor.

Rozměry matice IxJ, kterou má obsahovat vstupní soubor, jsou vypočteny následovně:

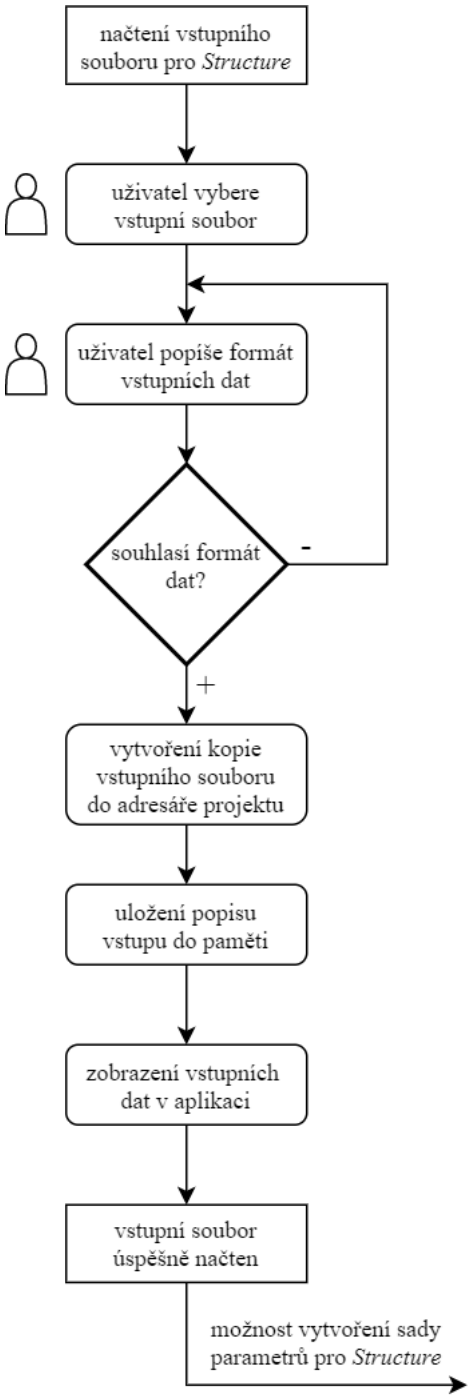
$$I = (NUMINDS \cdot PLOIDY) + M + r + D + (NUMINDS \cdot p) \tag{3.1}$$

kde $NUMINDS$ představuje počet jednotlivců, $PLOIDY$ je ploidie dat a M, r, D a p (0 nebo 1) jsou logické příznaky, které označují výskyt řádků s názvy markerů, s označením recesivních alel, se vzdálenosti mezi markery a nebo informacemi o fázi vazby (všechny tyto údaje zadává uživatel).

$$J = Loci + Label + PopData + PopFlag + LocData + Phen + ExtraCols \tag{3.2}$$

kde *Loci* je počet lokusů, *Label*, *PopData*, *PopFlag*, *LocData* a *Phen* (0 nebo 1) jsou logické příznaky, které označují výskyt těchto sloupců v datech a *ExtrCols* je počet extra sloupců, které jsou programem ignorovány (všechny tyto údaje zadává uživatel). Výpočet rozměrů matice je odvozen z informací o tom, jak má vypadat vstupní soubor (viz kapitola 1.1.2).

Jestliže jsou uživatelem zadaná data v pořádku, popis formátu vstupního souboru je uložen do paměti, pro pozdější generování parametrických souborů. Také jsou zobrazena vstupní data a do adresáře projektu, je vytvořena kopie souboru, jenž tyto data obsahuje. Dialog, obsluhující nahrávání vstupních dat je zavřen, a uživatel může pokračovat vytvářením nové sady parametrů pro běh programu *Structure*.



Obr. 3.4: Diagram postupu načítání vstupního souboru pro program *Structure*.

3.3.2 Tvorba sady parametrů pro program Structure

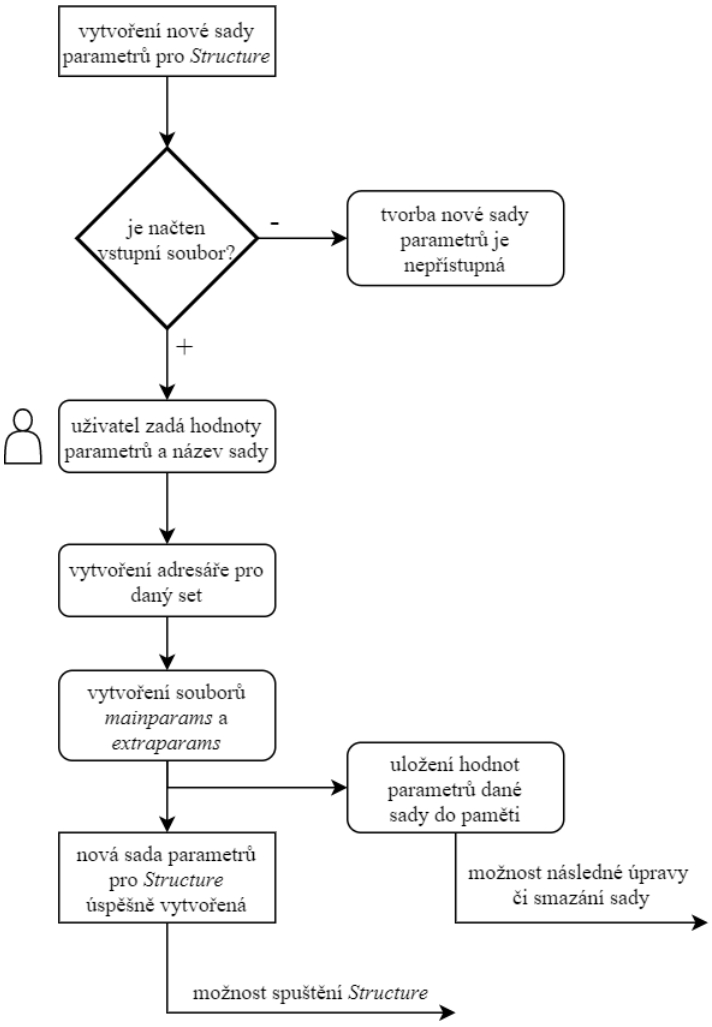
Novou sadu parametrů je možné vytvořit pouze v případě, že je již načten vstupní soubor. Z popisu vstupního souboru se totiž automaticky odvodí určité parametry, potřebné pro vytvoření souboru *mainparams*. Průběh tohoto procesu v aplikaci zobrazuje diagram na obrázku 3.5.

Jestliže je tedy načten vstupní soubor, uživateli je umožněn přístup do části programu pro zadávání a tvoření nové sady parametrů. Po zadání zvolených parametrů, uživatel musí tuto sadu pojmenovat. S tímto jménem je poté v adresáři projektu vytvořena složka, do které jsou uloženy dva parametrické soubory. Po spuštění práce aplikace *Structure* pro danou sadu parametrů, jsou zde také ukládány výstupní soubory.

Hodnoty parametrů, spjaté s názvem sady, jsou uložené do paměti. Uživatel má následně možnost tyto parametry upravovat, či celou sadu smazat. V případě změny parametrů, jsou taktéž přepsány dané parametrické soubory. Pokud si uživatel přeje sadu parametrů smazat, je smazána celá složka, i s případnými výsledky.

Poté, co jsou úspěšně vytvořeny parametrické soubory a hodnoty parametrů uloženy do paměti, je možné spustit *Structure* s potřebnými parametry a datovým souborem.

Sad parametrů si může uživatel vytvořit více, jestliže chce nebo potřebuje srovnat výsledky programu pro různé hodnoty parametrů.



Obr. 3.5: Diagram postupu pro vytvoření sady parametrů pro program *Structure*.

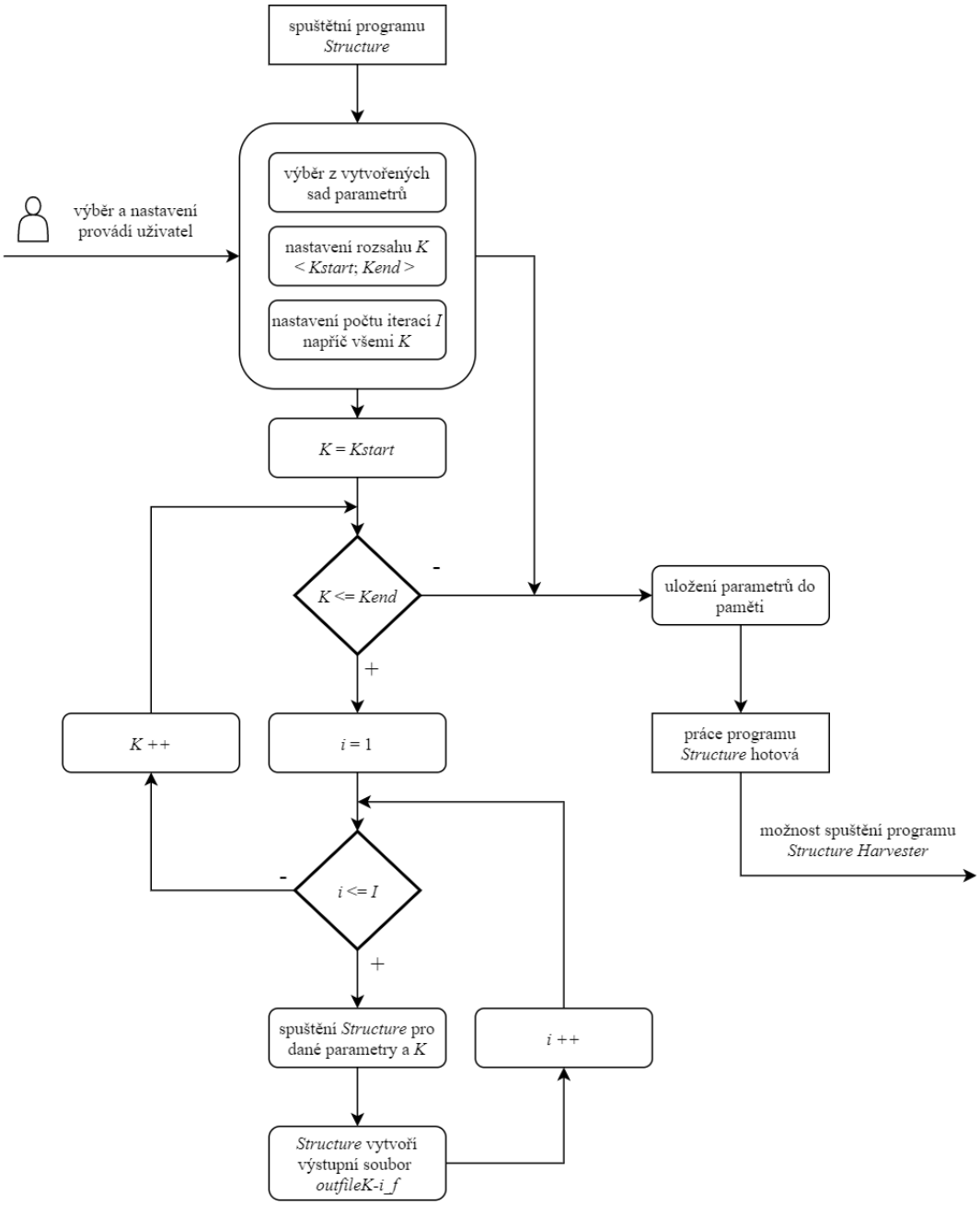
3.3.3 Spouštění programu Structure

Program *Structure* je možno spustit, jestliže má uživatel nahraný vstupní soubor a vytvořenou alespoň jednu sadu parametrů. Před spuštěním je otevřen dialog, kde se po uživateli vyžaduje zvolení sady parametrů, zadání rozsahu hodnot K a zvolení hodnoty počtu iteraci R, napříč všemi K. Po zadání platných hodnot, se začíná automatizovaně spouštět aplikace *Structure*. Průběh automatizovaného spouštění je zobrazen v diagramu na obrázku 3.6.

Aplikace spouští jeden běh *Structure* po druhém, R-krát pro každé K (ze zvoleného rozsahu). *Structure* generuje výstupní soubory nazvané *outfileK-r_f*, kde $K \in < Kstart; Kend >$ a $r \in < 1, 2, \ldots R >$. Při běhu programu lze navrženou aplikaci dále ovládat, například uživatel může vytvářet nové sady parametrů, nelze ovšem spustit další práci *Structure*. Jakmile je práce dokončena, uživateli je to ozná-

meno, může zobrazit vygenerované výstupní soubory, a může je dále zpracovávat spuštěním programu *Structure Harvester*.

Parametry zadané před spuštěním *Structure*, tedy rozsah K a počet iterací R, jsou uloženy do paměti pro automatické odvození těchto hodnot při práci s navazujícími programy.



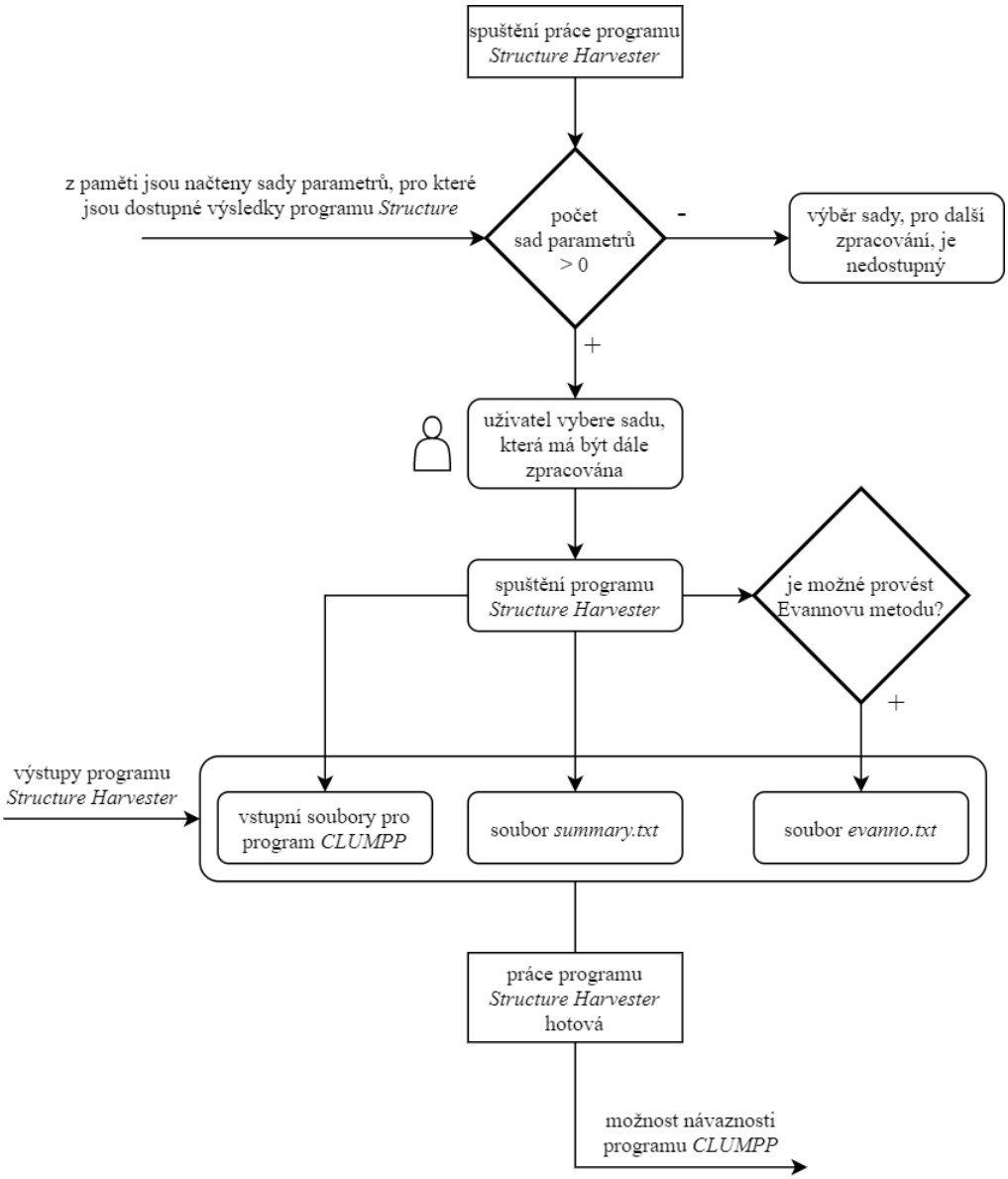
Obr. 3.6: Diagram automatizovaného spouštění programu *Structure*.

3.3.4 Spouštění programu Structure Harvester

Před tím, než je uživateli umožněno spuštění *Structure Harvester*, musí být dostupné výsledky, alespoň pro jednu sadu parametrů, z programu *Structure*. Pro *Structure Harvester* není potřeba, aby uživatel zadával nějaké parametry. Pokud chce aplikaci spustit, stačí vybrat sadu parametrů, pro kterou jsou z předešlých kroků dostupné výstupní soubory z programu *Structure*. Naše aplikace pak automaticky spouští *Structure Harvester* (Python skript), ten postupně generuje výstupní soubory. Výstupní soubory zahrnují textový soubor se shrnutím práce *summary.txt* a soubory s maticemi populací a jednotlivců, které se použijí jako vstup pro navazující program *CLUMPP*. Též je generován textový soubor *evanno.txt* se shrnutím provedé Evannovy metody, za předpokladu, že bylo možné ji provést. Celý proces je znázorněn v diagramu na obrázku 3.7.

Jestliže nebyla provedena Evannova metoda, ale uživatel potřebuje, aby provedena byla, musí opakovat zpracování dat s předešlým programem *Structure*, a zvolit větší počet iterací R, či zvětšit rozsah analyzovaných K.

Po ukončení práce programu *Structure Harvester*, je do paměti uloženo, pro jakou sadu parametrů byly výstupy vygenerovány. Poté je umožněna práce s navazující aplikací *CLUMPP*.



Obr. 3.7: Diagram automatizovaného spouštění programu *Structure Harvester*.

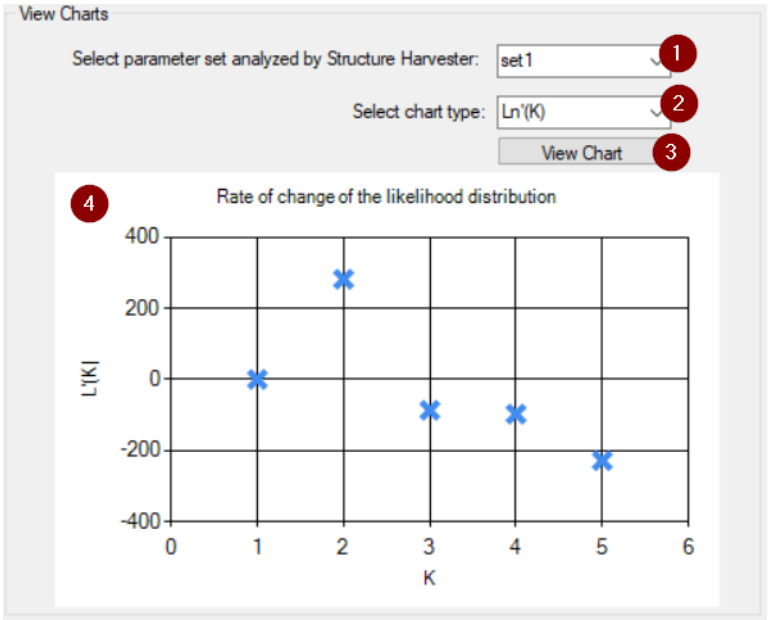
3.3.5 Tvorba grafů z výstupu programu Structure Harvester

V kapitole 1.2.1 bylo zmíněno, že webová verze programu *Structure Harvester* poskytuje zpracované výsledky též ve formě grafů. V navržené aplikaci je ale používána verze Python skriptu, která grafy negeneruje. Proto byl navržen algoritmus, který extrahuje data z výstupních souborů *summary.txt* a *evanno.txt*. Uživatel si pak v aplikaci může zobrazit stejné grafy, jako ve webové verzi programu *Structure Harvester*. Konkrétně se jedná o čtyři grafy. První z nich zobrazuje průměrné pravděpodobnosti jednotlivých hodnot K. Další tři zobrazují výsledky Evannovy metody.

Část aplikace, kde si uživatel může vygenerovat grafy je zobrazena na obrázku 3.8. Jednotlivé prvky této části mají následující funkci:

1. Slouží k výběru sady parametrů, pro kterou je hotová *Structure Harvester* analýza.
2. Výběr zobrazených hodnot v grafu.
3. Po stisknutí tohoto tlačítka se vygeneruje zvolený graf.
4. Graf zvolených hodnot v závislosti na hodnotě K.

Graf který je na obrázku 3.8, je generován z hodnot výstupního souboru, jehož část, s danými hodnotami, je na obrázku 3.9 (konkrétně se jedná o soubor *evanno.txt*). Na obrázku je vyznačeno, které hodnoty graf zobrazuje. Jedná se o graf první derivace pravděpodobností jednotlivých hodnot K.



Obr. 3.8: Příklad vygenerovaného grafu z výstupu programu *Structure Harvester*.

```
##########
# K     Reps    Mean LnP(K)     Stdev LnP(K)    Ln'(K)          |Ln''(K)|       Delta K
1       3       -4357.4000      0.0000          NA              NA              NA
2       3       -4075.9667      0.8083          281.433333      368.166667      455.488123
3       3       -4162.7000      5.0120          -86.733333      10.600000       2.114930
4       3       -4260.0333      44.3136         -97.333333      130.933333      2.954700
5       3       -4488.3000      101.6769        -228.266667     NA              NA
```

Obr. 3.9: Příklad výstupního souboru programu *Structure Harvester*, použitého ke tvorbě grafů.

3.3.6 Tvorba sady parametrů pro program CLUMPP

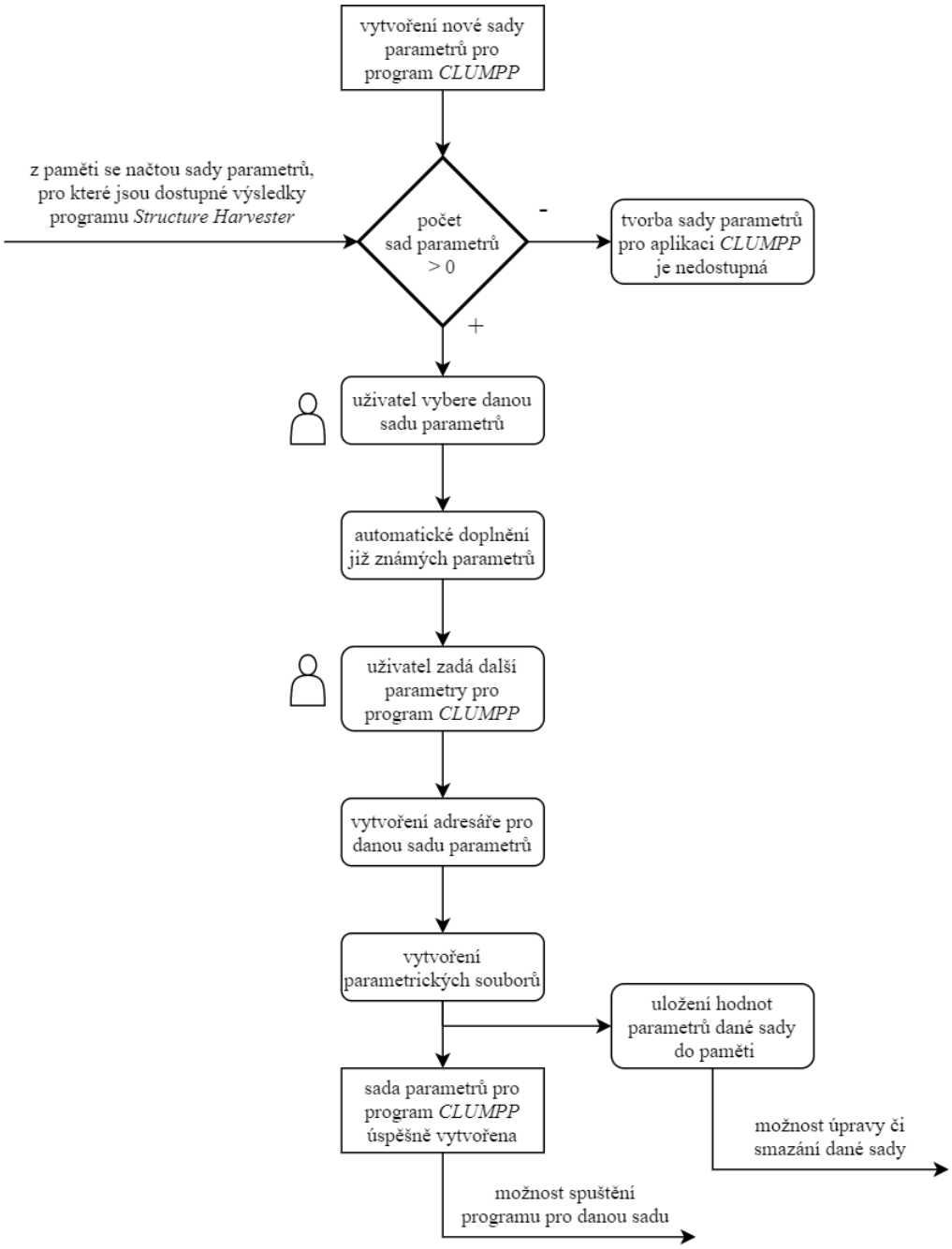
Jestliže byl původní vstupní soubor o určité sadě parametrů zpracován programy *Structure* a *Structure Harvester*, pak je uživateli umožněno dále zpracovávat data pomocí aplikace *CLUMPP*. Před spuštěním samotné aplikace, je po uživateli vyžadováno zadání potřebných parametrů. Známé parametry, zvolené sady parametrů, konkrétně počet jedinců a počet iterací R analýzy programu *Structure*, jsou automaticky doplněny (jsou uloženy v paměti).

Po zadání parametrů jsou vytvořeny dva parametrické soubory (*paramfile0* a *paramfile1*) pro vstup do programu *CLUMPP*. Jak bylo zmíněno v kapitole 1.3.2, v parametrickém souboru je nastavován parametr **DATATYPE** (na hodnotu 0 nebo 1), který určuje, zda-li program *CLUMPP* zpracovává soubor s maticí jednotlivců, či soubor s maticí populací. Tento parametr není možné měnit z příkazové řádky, proto jsou vytvořeny dva parametrické soubory. V případě, že *CLUMPP* zpracovává soubor s jednotlivci (výstupní soubor programu *Structure Harvester* s příponou *.indfile*), pak čte parametrický soubor *paramfile0*. Pokud zpracovává matici populací (ve výstupních souborech programu *Structure Harvester* s příponou *.popfile*), pak čte *paramfile1*. O toto se stará navržená aplikace.

Hodnoty parametrů pro danou sadu jsou uloženy do paměti a lze je, jako v případě u programu *Structure*, upravovat.

Po úspěšném vytvoření parametrických souborů, a uložení hodnot parametrů do paměti, pro danou sadu je možné pro tyto parametry spustit aplikaci *CLUMPP*.

Proces tvorby sady parametrů pro program *CLUMPP* je zobrazen na obrázku 3.10.



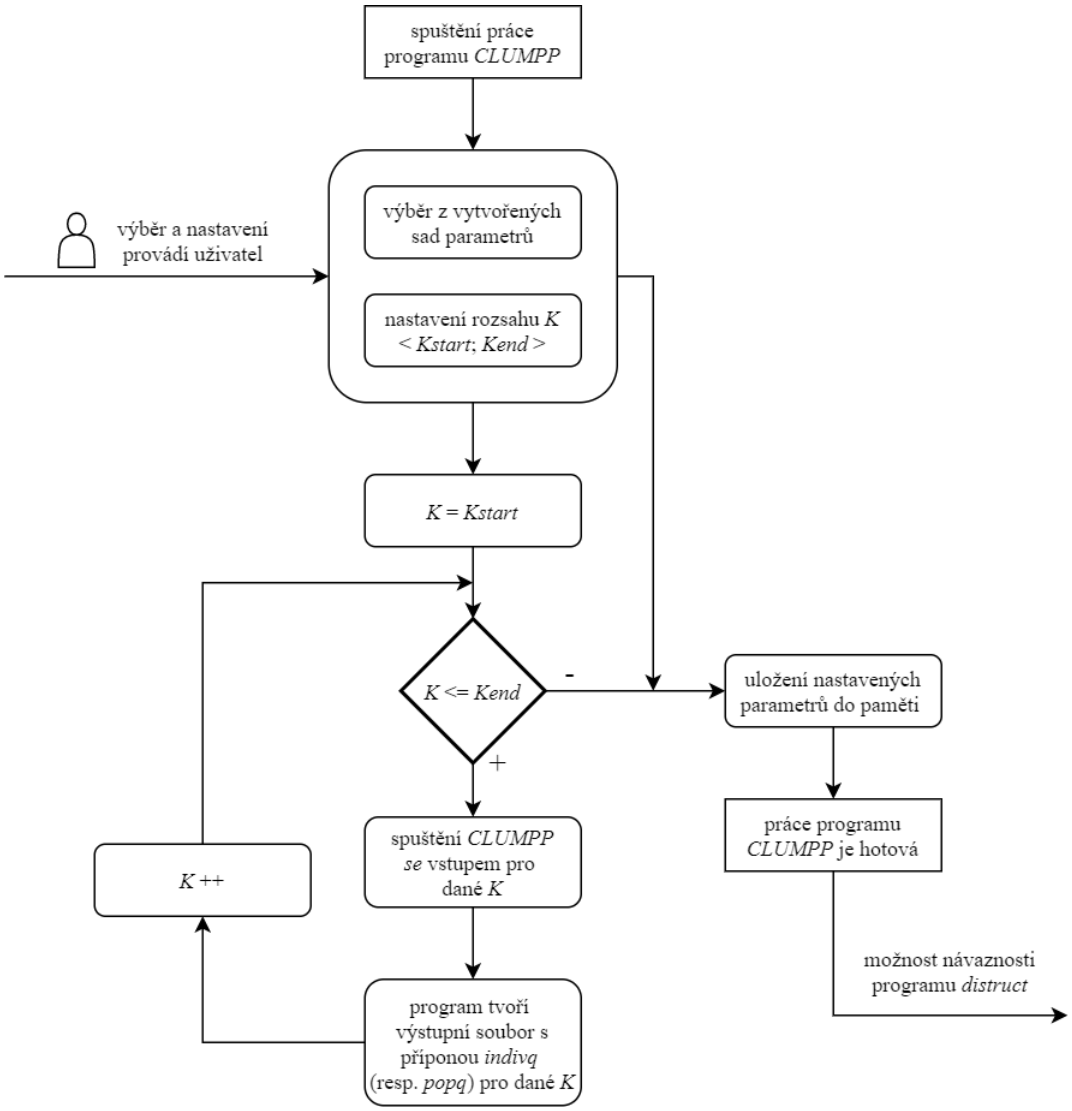
Obr. 3.10: Diagram postupu pro vytvoření sady parametrů pro program *CLUMPP*.

3.3.7 Spouštění programu CLUMPP

Spuštění programu *CLUMPP* je možné, pokud jsou dostupné výstupní soubory z aplikace *Structure Harvester* a byly vytvořeny parametrické soubory. Uživatel pak může vybrat danou sadu parametrů a rozsah K (ten je omezen rozsahem, zvoleným u předešlé práce programu *Structure*).

Po nastavení parametrů, jsou současně spouštěny dva běhy programu *CLUMPP*. V jednom případě *CLUMPP* zpracovává data o jednotlivcích, v druhé případě zpracovává data o populacích. Ke každému zpracovanému K tak vytváří dva soubory, které se použijí jako vstup do navazujícího programu pro vizualizaci dat, tedy *distruct*. Diagram průběhu práce je na obrázku 3.11.

Po dokončení práce programu *CLUMPP*, je do paměti uložen zvolený rozsah K, a ke zvolené sadě parametrů, je možné dále přistoupit v práci s programem *distruct*.



Obr. 3.11: Diagram automatizovaného spouštění programu *CLUMPP*.

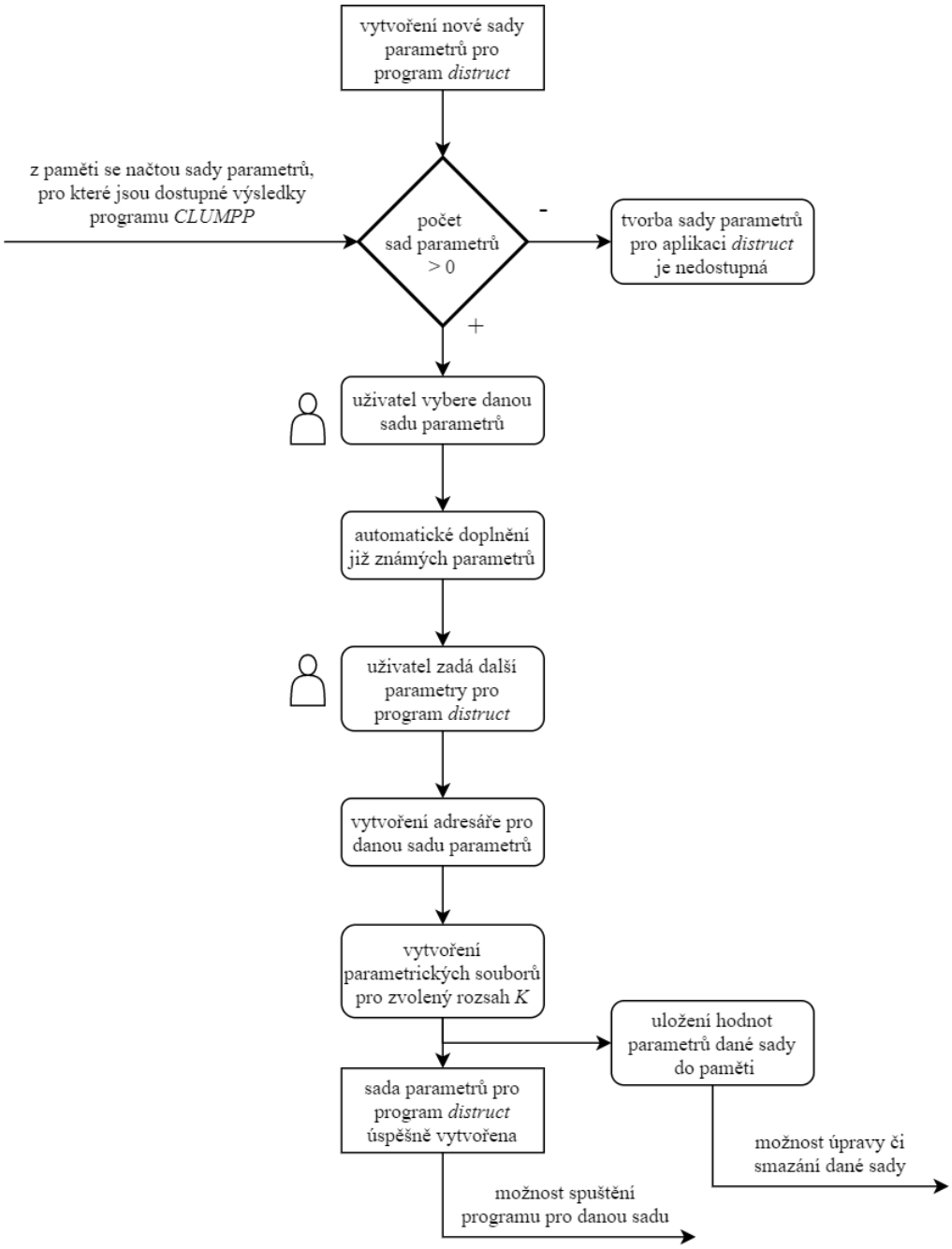
3.3.8 Tvorba sady parametrů pro program *distruct*

V případě, že jsou alespoň pro jednu sadu parametrů dostupné výstupní soubory z práce programu *CLUMPP*, pak je možné navázat aplikací *distruct*. Stejně jako v předchozích případech, vyžaduje *distruct* pro svou práci určité parametry. Některé parametry jsou znovu odvozeny z předchozích programů (konkrétně se jedná o počet jednotlivců, počet populací, a největší možný rozsah hodnot K, pro které byla provedena práce přechozího programu *CLUMPP*). Uživatel v tomto případě nastavuje parametry týkající se zejména vzhledu vizualizace, a poté rozsah K.

Po tom, co si uživatel vybere, se kterou sadou parametrů chce dále pracovat v případě programu *distruct*, a doplní další parametry pro tento program, je vytvořena složka pro sadu parametrů a samotné parametrické soubory pro zvolený rozsah K. Pokud si tedy uživatel vybere například rozsah $K = 3, 4, 5$, jsou v dané složce vytvořeny 3 parametrické soubory, v každém je jiný parametr K. Parametrické soubory jsou takto vytvořeny po jednom pro správnou funkčnost aplikace *distruct*.

Vytvořená sada parametrů a její hodnoty jsou znovu uloženy do paměti, a uživatel se k nim může kdykoliv vrátit a upravovat.

Diagram postupu vytvoření nové sady parametrů pro aplikaci *distruct* je na obrázku 3.12.



Obr. 3.12: Diagram postupu pro vytvoření sady parametrů pro program *distruct*.

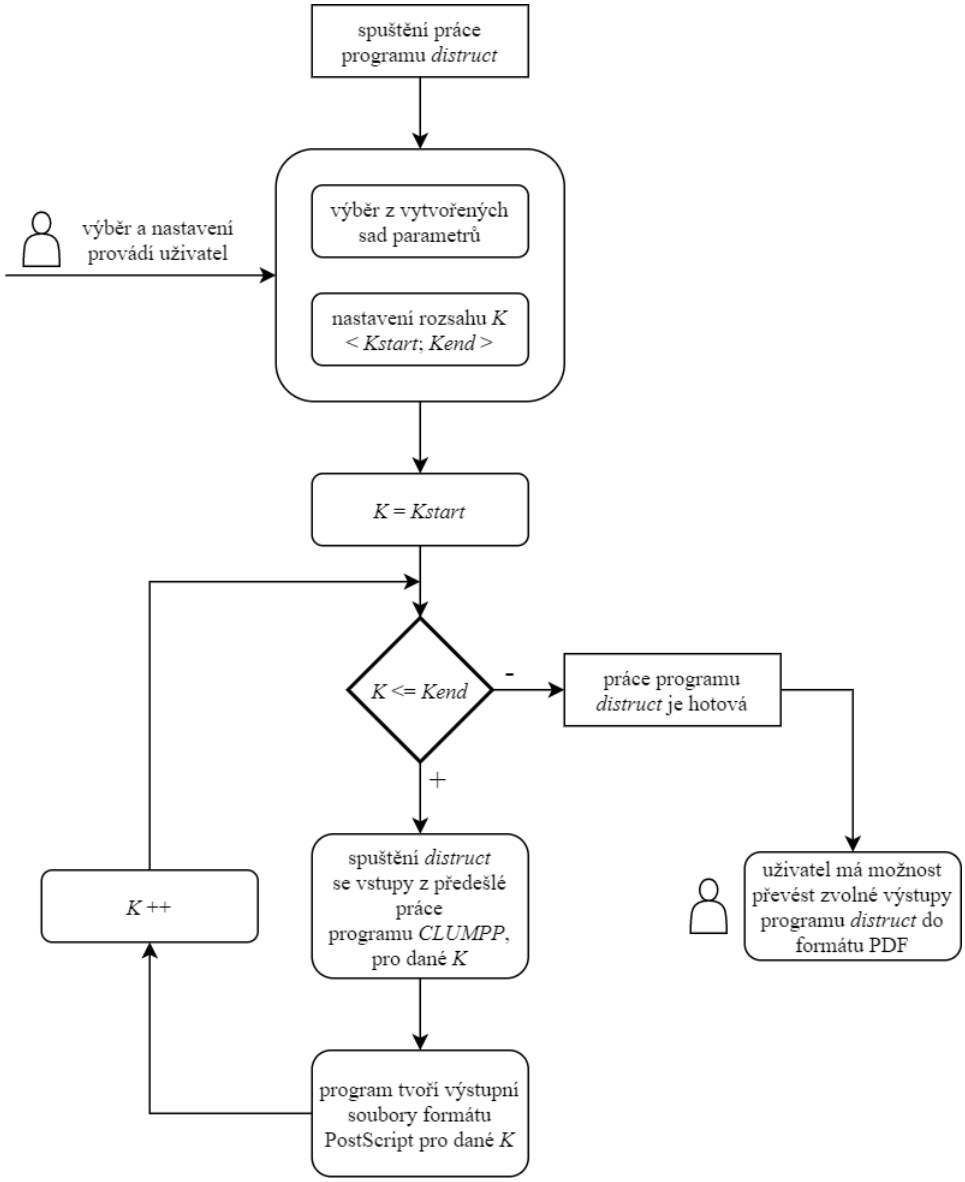
3.3.9 Spouštění programu *distruct*

Má-li uživatel vytvořené parametrické soubory pro *distruct*, může pro zvolenou sadu program spustit. Naše aplikace začne automatizovaně spouštět *distruct*, jeden běh po druhém, pro zvolený rozsah K. Pro každou takovou hodnotu K jsou dostupné dva výstupní soubory programu *CLUMPP*, jeden obsahuje matici populací, druhý

matici jednotlivců (například pro $K = 3$ jsou to soubory *K3.indivq* a *K3.popq*). Oba tyto soubory, pro dané K, slouží jako vstup pro jeden běh programu *distruct*.

Výstupem aplikace *distruct*, pro každé K, je jeden soubor s vizualizací dat ve formátu PostScript. Naše aplikace umožňuje převádět tyto souboru do formátu PDF, dle uvážení uživatele.

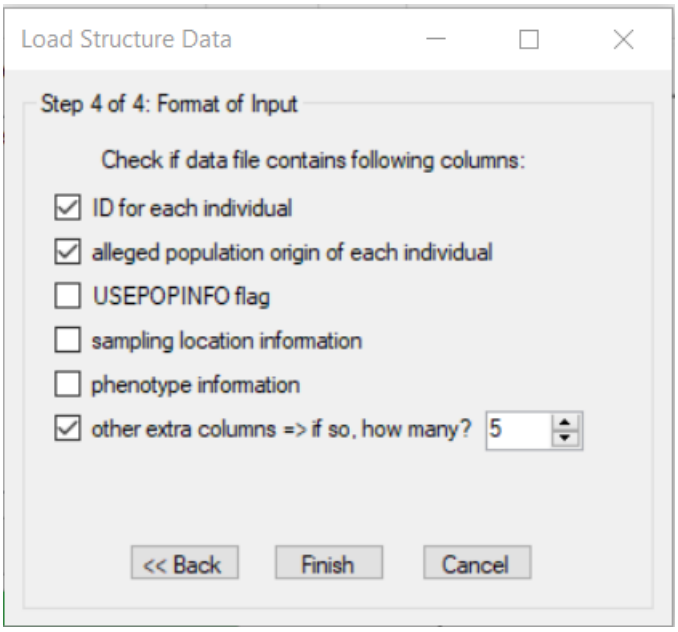
Diagram postupu automatizovaného spouštění aplikace *distruct* je na obrázku 3.13.



Obr. 3.13: Diagram automatizovaného spouštění programu *distruct*.

3.4 Ověření chodu aplikace na vzorku vstupních dat

Správná funkcionalita programu byla pro účely této práce testována na vzorku vstupních dat dostupné z [12]. Jedná se o vzorek dat s genotypovými daty o 100 jedincích. Formát dat je následující: LABEL = 1, POPDATA = 1, EXTRACOLS = 5, NUMLOCI = 10, MISSING = -9, PLOIDY = 2, MARKERNAMES = 1. Část dialogu pro popis vstupních dat je na obázku 3.14. Popsaný vzorek dat je pak zobrazen uživateli (obrázek 3.15).



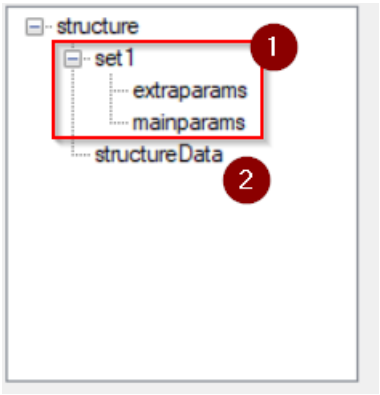
Obr. 3.14: Dialog popisu vstupních dat pro program *Structure*.

Label	Pop	Extra 1	Extra 2	Extra 3	Extra 4	Extra 5	Locus 1	Locus 2	Locus 3	Locus 4	Locus 5	Locus 6	Locus
							1	2	3	4	5	6	7
1	19	0.000000	0.000000	1.000000	0.000000	0.000000	2	3	3	1	2	2	1
1	19	0.000000	0.000000	1.000000	0.000000	0.000000	2	1	1	1	1	2	1
2	1	1.000000	0.000000	0.000000	0.000000	0.000000	2	3	3	1	1	3	1
2	1	1.000000	0.000000	0.000000	0.000000	0.000000	0	0	0	1	1	3	1
3	20	0.000000	0.991974	0.008026	0.000000	0.000000	0	0	0	1	3	3	1
3	20	0.000000	0.991974	0.008026	0.000000	0.000000	3	3	3	1	3	3	1
4	4	0.000000	0.000000	0.000000	1.000000	0.000000	2	3	3	1	3	2	1
4	4	0.000000	0.000000	0.000000	1.000000	0.000000	2	3	0	1	3	2	1
5	9	0.000000	0.000000	0.000000	0.000850	0.999150	2	0	1	1	1	2	3
5	9	0.000000	0.000000	0.000000	0.000850	0.999150	2	0	3	1	1	2	2
6	24	1.000000	0.000000	0.000000	0.000000	0.000000	2	3	3	1	3	2	1

Obr. 3.15: Zobrazení části načtených vstupních dat pro program *Structure*.

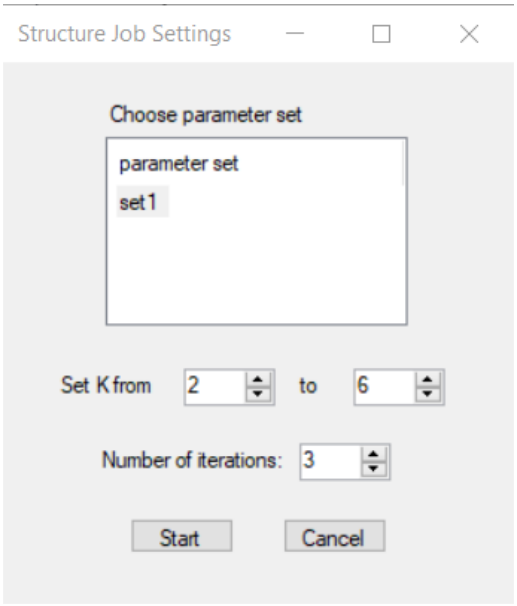
Po načtení vstupního souboru vytvoříme sadu parametrů, pojmenovanou *set1*. Hlavní parametry jsou odvozeny z popisu formátu vstupního soubor, zbylé hodnoty parametrů, pro testovací účely, ponecháme na výchozích hodnotách. Na obrázku 3.16

vidíme zobrazenou složku, kam se úspěšně vytvořily parametrické soubory (1) a zkopíroval vstupní datový soubor (2).

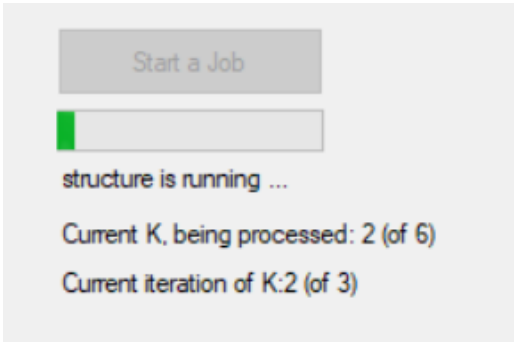


Obr. 3.16: Zobrazení složky s daty pro program *Structure*.

Po vytvoření sady parametrů můžeme nastavit automatizované spouštění programu *Structure*. Dialog pro nastavení je na obrázku 3.17a. Po spuštění, aplikace postupně informuje, pro které K (a o jaké iteraci) zrovna *Structure* zpracovává data (obrázek 3.17b).



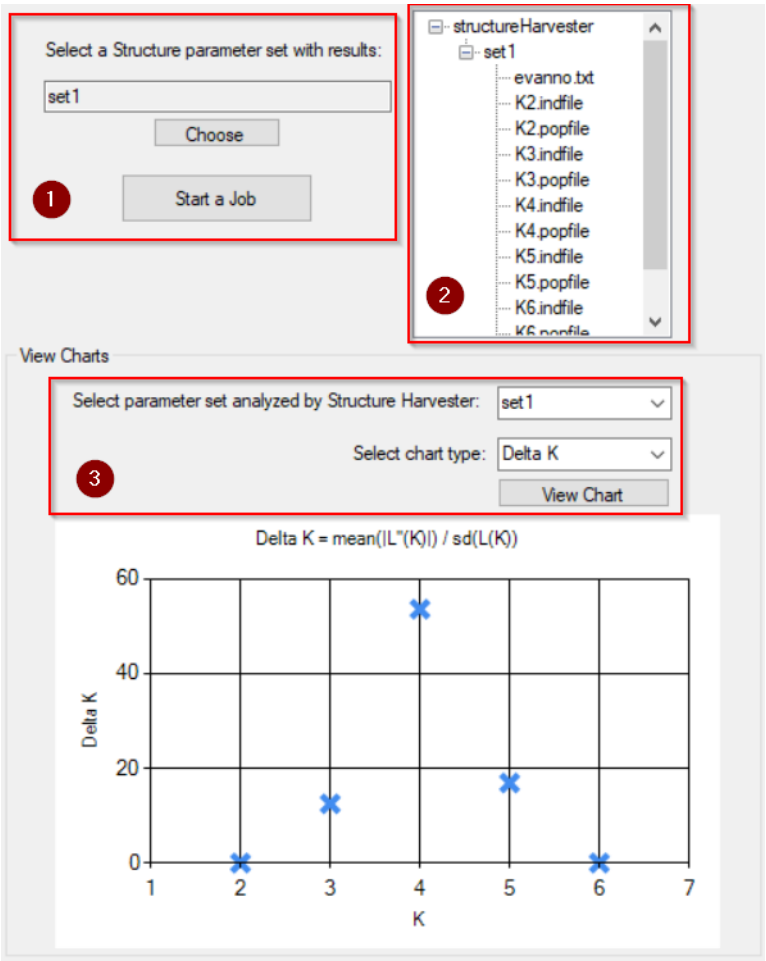
(a) Nastavení automatizovaného spouštění programu *Structure*



(b) Průběh automatizovaného spouštění programu *Structure*

Obr. 3.17: Nastavení a průběh spouštění programu *Structure* v aplikaci.

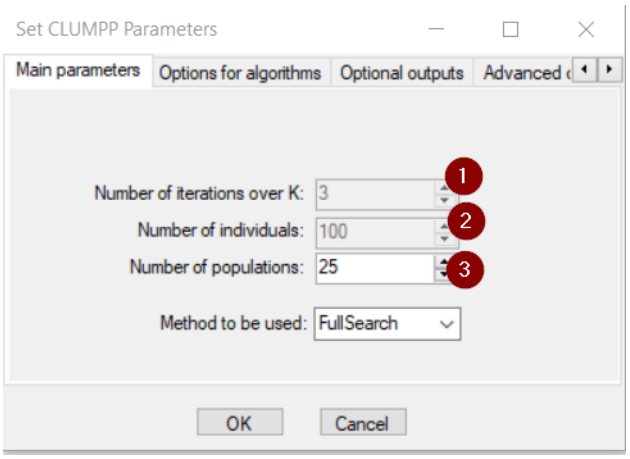
Po dokončení analýzy programem *Structure*, můžeme pro zpracovanou sadu parametrů spustit *Structure Harvester*. Záložka pro *Structure Harvester* je na obrázku 3.18. Nejprve vybereme sadu, pro kterou jsou dostupné výsledku programu *Structure* a spustíme *Structure Harvester* (1). Výstupní soubory jsou generovány do adresáře pro program *Structure Harvester*, do složky pro danou sadu parametrů (2).Jestliže jsou dostupné soubory *summary.txt* a *evanno.txt*, můžeme zobrazovat grafy (3).



Obr. 3.18: Nastavení spouštění programu *Structure Harvester* a vygenerované výsledky.

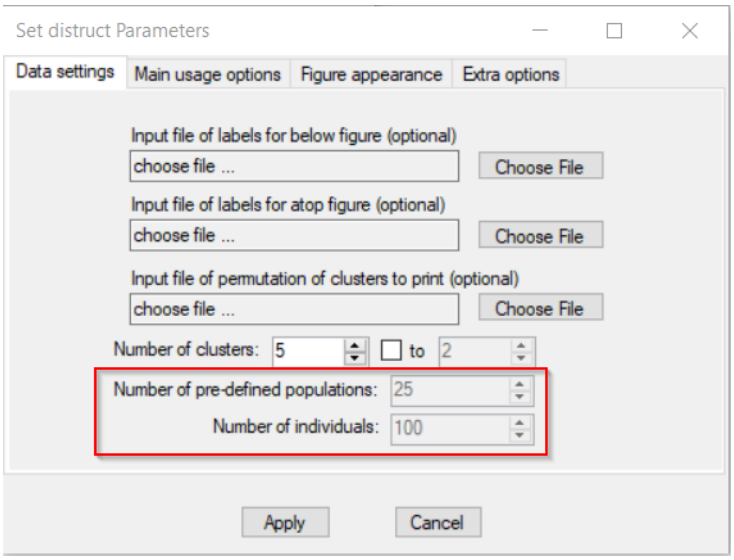
Pro vytvořenou sadu parametrů *set1* máme výsledky programu *Structure Harvester*, můžeme tedy pokračovat nastavením prametrů pro program *CLUMPP*. Dialog pro nastavení hodnot parametrů je na obrázku 3.19. Hodnoty pro označené komponenty (1) a (2), jsou odvozeny automaticky. Hodnota pro komponentu (3), tedy počet předdefinovaných populací, je dána vzorkem dat (doplňuje uživatel). Po zadání hodnot parametrů, se vytvoří parametrické soubory (popsáno v kapitole 3.3.6),

a můžeme spustit program *CLUMPP*. Dialog pro nastavení spouštění programu je obdobný jako v případě *Structure* (obrázek 3.17a). U programu *CLUMPP* ale nenastavujeme iterace.



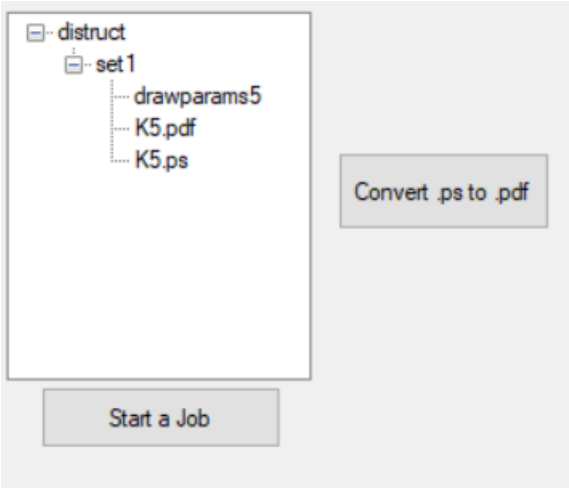
Obr. 3.19: Dialog pro nastavení hodnot parametrů pro program *CLUMPP*.

Máme dostupné výsledky programu *CLUMPP* a můžeme vytvořit sadu parametrů pro *distruct*. Dialog pro zadávání hodnot parametrů je na obrázku 3.20. Zvýrazněné komponenty na obrázku jsou doplněny automaticky. Po zadání hodnot parametrů jsou vytvořeny parametrické soubory pro zvolený rozsah K a můžeme spustit *distruct*. Dialog pro nastavení spouštění programu je znovu obdoba toho pro *Structure* (obrázek 3.17a) bez nastavení počtu iterací.

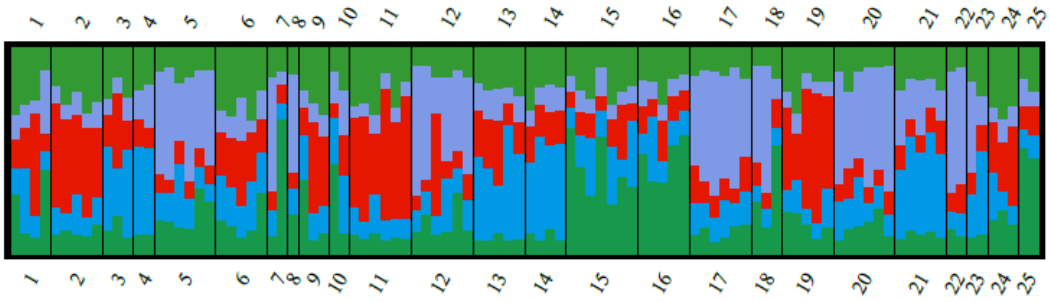


Obr. 3.20: Dialog pro nastavení hodnot parametrů pro program *distruct*.

Adresář s parametrickými a výstupními soubory programu *distruct* je zobrazen v komponentě *TreeView* (obrázek 3.21. Jestliže chceme převést výstupní soubor formátu PostScript do formátu PDF, stačí v komponentě označit daný soubor a stisknout tlačítko pro konverzi. Soubor PDF je pak vygenerován do stejné složky. Dvjoklikem na daný PDF soubor se otevře výchozí program pro zobrazení PDF souborů. Ten obsahuje vizualizaci dat, generovanou programem *distruct*. Na obrázku 3.22 je obsah souboru *K5.pdf*



Obr. 3.21: Zobrazení adresáře s výsledky programu *distruct*.



Obr. 3.22: Vizualizace dat vytvořená programem *distruct*.

Závěr

V rámci této práce proběhlo seznámení se s čtyřmi aplikacemi pro zpracování genotypových dat, konkrétně tedy *Structure*, *Structure Harvester*, *CLUMPP* a *distruct*. Byla vysvětlená funkcionalita jednotlivých programů a popsány jejich vstupní i výstupní soubory.

Jednotlivé výsledky programů *Structure*, *Structure Harvester*, *CLUMPP* a *distruct* na sebe navazují. Na základně této návaznosti byl zpracován návrh automatizace postupného zpracování genotypových dat. V rámci tohoto návrhu bylo vysvětleno, co musí uživatel specifikovat před spouštěním daných programů, a jak se jednotlivé programy budou spouštět po sobě, aby uživatel došel k relevantnímu výsledku, v podobě zpracovaných genotypových dat, a jejich vizualizaci.

Na bázi návrhu, pak byla naprogramována aplikace, která uživateli umožňuje spouštět automatizovaně programy *Structure*, *Structure Harvester*, *CLUMPP* a *distruct*. Aplikace spouští programy na pozadí, a uživatele informuje o průběhu práce. V případě programu *Structure Harvester* aplikace pracuje s verzí Python skriptu, aby chod aplikace nebyl závislý na internetu. Tato verze standartně negeneruje pro uživatele grafy výstupních hodnot, proto byl též navržen algoritmus, pro extrakci dat z výstupních souborů programu *Structure Harvester*, a následného vykreslení potřebných grafů. Uživatelské rozhraní navržené aplikace též umožňuje uživateli nastavovat hodnoty veškerých parametrů, potřebných pro chod používaných programů. Také je možné zobrazovat obsah všech výstupních souborů. Aplikace byla úspěšně testována na vzorku vstupních dat.

Aplikace pro automatizaci práce s programy *Structure*, *Structure Harvester*, *CLUMPP* a *distruct* byla navržena tak, že vyžaduje, aby měl uživatel dostupná vstupní data pro program *Structure* a postupně je nechal zpracovat jednotlivými programy. Možné pokračování ve vývoji aplikace by tedy bylo zpřístupnit nahrávání vstupních souborů pro programy individuálně.

Literatura

[1] PRITCHARD, J. K., WEN, X., FALUSH, D. Documentation for structure software: Version 2.3. *Department of Human Genetics, University of Chicago. Department of Statistics, University of Oxford*. 2010. [cit. 30. 12. 2020]. Dostupné z URL: <https://web.stanford.edu/group/pritchardlab/structure_software/release_versions/v2.3.4/structure_doc.pdf>.

[2] PRITCHARD, J. K., STEPHENS, M., DONNELLY, P. Inference of Population Structure Using Multilocus Genotype Data. *Genetics*. 2000, 155, 945–959. ISSN 0016-6731. [cit. 13. 12. 2020]. Dostupné z URL: <https://www.genetics.org/content/genetics/155/2/945.full.pdf>.

[3] FALUSH, D., STEPHENS, M., PRITCHARD, J. K. Inference of population structure: Extensions to linked loci and correlated allele frequencies. *Genetics*. 2003, 164, 1567—1587. ISSN 0016-6731. [cit. 30. 12. 2020]. Dostupné z URL: <https://www.genetics.org/content/genetics/164/4/1567.full.pdf>.

[4] FALUSH, D., STEPHENS, M., PRITCHARD, J. K. Inference of population structure using multilocus genotype data: dominant markers and null alleles. *Mol Ecol Notes*. 2007, 7, 574–578. [cit. 30. 12. 2020]. Dostupné z URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1974779/>.

[5] EVANNO, G., REGNAUT, S., GOUDET, J. Detecting the number of clusters of individuals using the software STRUCTURE: a simulation study. *Mol Ecol*. 2005, 14, 2611–2620. ISSN 1365-294X. [cit. 15. 12. 2020].

[6] EARL, D. A., VONHOLDT, B. M. STRUCTURE HARVESTER: a website and program for visualizing STRUCTURE output and implementing the Evanno method. *Conservation Genetics Resources*. 2012, 4, 359–361. [cit. 15. 12. 2020]. Dostupné z URL: <http://alumni.soe.ucsc.edu/~dearl/papers/2012structureHarvester.pdf>.

[7] JAKOBSSON, M., ROSENBERG, N. A. CLUMPP: a cluster matching and permutation program for dealing with label switching and multimodality in analysis of population structure *Bioinformatics*. 2007, 23, 1801–1806. [cit. 16. 12. 2020]. Dostupné z URL: <https://academic.oup.com/bioinformatics/article/23/14/1801/188285>.

[8] JAKOBSSON, M., ROSENBERG, N. A. CLUster Matching and Permutation Program. *Department of Human Genetics, University of Michigan*. 2009. [cit. 30. 12. 2020]. Dostupné z URL: <https://rosenberglab.stanford.edu/software/CLUMPP_Manual.pdf>.

[9] CLUMPP: CLUster Matching and Permutation Program (example). *Rosenberg Lab* [online]. [cit. 19. 12. 2020]. Dostupné z URL: <https://rosenberglab.stanford.edu/clumppExample.html>.

[10] ROSENBERG, N. A. DISTRUCT: a program for the graphical display of population structure *Molecular Ecology Notes*. 2004, 4, 137–138. [cit. 19. 12. 2020]. Dostupné z URL: <https://rosenberglab.stanford.edu/papers/distructNote.pdf>.

[11] ROSENBERG, N. A. Distruct: a program for the graphical display of population structure *Department of Human Genetics, University of Michigan*. 2007. [cit. 30. 12. 2020]. Dostupné z URL: <https://rosenberglab.stanford.edu/software/distructManual.pdf>.

[12] Simulated microsatellite data with location information. [cit. 16. 5. 2021]. Dostupné z URL: <https://web.stanford.edu/group/pritchardlab/software/example-data/locprior.str>.

[13] WikiSkripta, projekt 1. lékařské fakulty a Univerzity Karlovy, příspěvek UK k výukovým zdrojům sítě lékařských fakult MEFANET. ISSN 1804-6517. [cit. 1. 1. 2021]. Dostupné z URL: <https://www.wikiskripta.eu/>.

Seznam příloh

A Pojmy z biologie a genetiky

- **gen** - Gen je základní jednotka genetické informace. Je to určitý úsek DNA na chromozomu.
- **chromozom** - Chromozom nese lineárně uspořádané geny.
- **genotyp** - Genotyp může být buď informace o genetické konstituci buňky, nebo organismu a/nebo jedince. Genotyp jedince tedy představuje jeho veškerou zděděnou genetickou výbavu, která je zaznamenána v sekvenci DNA. V užším pojetí (i z hlediska kontextu genotypu v této práci) můžeme říci, že genotyp je dvojice alel téhož genu, neb genetická výbava jedince je diploidní.
- **diploidní** - Organismus (nebo buňka) je diploidní, jestliže obsahuje dvě sady chromozomů.
- **alela** - Alela je konrétní formou genu, zajišťující jeho konkrétní fenotypový projev.
- **fenotyp** - Fenotyp je soubor všech definovatelných charakteristik (znaků) jedince. Jsou to znaky pozorovatelné a definovatelné na úrovni organismu (např. výška, hmotnost, IQ, chování jedince a další), ale i charakteristika určité fyziologické funkce (např. krevní tlak, hladina cukru v krvi atp.).
- **alelová frekvence** - relativní četnost určité alely v populaci
- **marker** - Marker je známá sekvence DNA.
- **lokus** - Lokus je genetický termín označující přesné místo (pozici) na chromozomu, na kterém se vyskytuje příslušný gen.
- **Hardy-Weinbergova rovnováha** - Hardy-Weinbergova rovnováha je teoretické rovnovážné rozložení alel v populaci.
- **klastr** - Klastr je skupina jedinců, vymezená stejnými/podobnými znaky.

Vysvětlivky k daným termínům jsou převzaty ze zdroje [13].