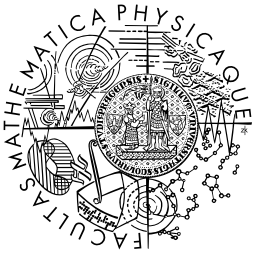


MATEMATICKO-FYZIKÁLNÍ FAKULTA
Univerzita Karlova

BAKALÁŘSKÁ PRÁCE

Kristýna Sommerová

Výpočet a aplikace MCD estimátoru pro robustní statistické analýzy

Katedra numerické matematiky

Vedoucí bakalářské práce:	Dipl.-Math Jurjen Duintjer Tebbens, Ph.D.
Studijní program:	Matematika
Studijní obor:	Obecná matematika

Praha 2016

Prohlašuji, že jsem tuto bakalářskou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne ..... Podpis autora

Název práce: Výpočet a aplikace MCD estimátoru pro robustní statistické analýzy

Autor: Kristýna Sommerová

Katedra: Katedra numerické matematiky

Vedoucí bakalářské práce: Dipl.-Math Jurjen Duintjer Tebbens, Ph.D., Ústav Informatiky AV ČR

Abstrakt: Tato práce popisuje jeden ze základních problémů robustní statistiky, který spočívá v detekci odlehlých hodnot, a jeho možné řešení pomocí Minimum covariance determinant estimátoru pro odhad střední hodnoty a varianční matice mnohorozměrných dat. Vysvětluje fungování tohoto estimátoru a zkoumá jeho vlastnosti. Zaměřuje se pak především na aproximaci pomocí algoritmu fastMCD, pro který upřesňuje numerické vlastnosti s důrazem na výpočtovou náročnost a stabilitu ve standardní implementaci v MATLABu. Diskutuje také možné úpravy algoritmu a jejich vliv na numerické vlastnosti. Na závěr na několika experimentech s reálnými daty ukazuje použití fastMCD algoritmu.

Klíčová slova: robustní statistika, minimum covariance determinant, fastMCD, C-step

Title: Computation and application of the MCD estimator for robust statistical analysis

Author: Kristýna Sommerová

Department: Department of Numerical Mathematics

Supervisor: Dipl.-Math Jurjen Duintjer Tebbens, Ph.D., Institute of Computer Science CAS

Abstract: This work describes one of the basic problems of robust statistics concerning outlier detection and its possible solution by using the Minimum covariance determinant estimator for estimates of the mean value and the covariance matrix with multivariate data. It explains how the estimator works and analyses its properties. The work concentrates on its approximation based on the fastMCD algorithm and specifies its numerical properties with emphasis on computational costs and stability of the standard implementation in MATLAB. It also discusses possible modifications of the algorithm and its effects on numerical properties. Lastly the work shows the usage of the fastMCD algorithm on a few real data experiments.

Keywords: robust statistics, minimum covariance determinant, fastMCD, C-step

Chtěla bych poděkovat svému vedoucímu bakalářské práce Dipl.-Math Jurjenovi Duintjerovi Tebbensovi, Ph.D. za odborné vedení, za cenné rady a trpělivost při zpracování této práce. Děkuji také RNDr. Janu Kalinovi, Ph.D za velkou pomoc se statistickou částí práce.

Obsah

Úvod

Ve statistice se při analýze a interpretaci dat často naráží na problém, že výsledky a odhady, které nám statistika dává, jsou pouze tak vypovídající, jak přesná data na počátku máme. Pokud získáme data, která jsou zatížená chybami, mohou tím být odhady ovlivněny.

V mnoha aplikacích se do měření dostávají tzv. odlehlé hodnoty, které v nějakém ohledu zcela vybočují z ostatních dat, protože jde například o chybná měření. Pokud máme velká data s mnoha proměnnými, nemusí být kvůli komplexnosti dat odlehlé hodnoty na první pohled nijak vidět. Tyto hodnoty ale obvykle posouvají odhady daleko od skutečných hodnot a dostáváme tak zkreslenou představu o vlastnostech námi zkoumaného problému.

Těmto potížím se snaží předejít robustní statistika [1]. Ta se především zabývá tím, jak odlehlé hodnoty v datech najít, aby se jejich vliv na výsledky mohl snížit či zcela odstranit. Metod je v dnešní době mnoho, tato práce se však zaměří na odhady střední hodnoty a rozptylu pomocí *minimum covariance determinant estimátoru* [2], zkráceně MCD estimátoru.

Tento estimátor je skutečně velmi rezistentní na odlehlé hodnoty, zvládá i data, jež obsahující téměř polovinu hodnot odlehlých. Algoritmizace estimátoru, tzv. fastMCD algoritmus [3], je navíc oproti jiným metodám poměrně rychlá a lze účinně použít i na data s velkými rozměry, které byl dříve problém analyzovat. FastMCD z důvodu zrychlení výpočtu dává obecně pouze odhad estimátoru a ne přesný MCD estimátor. Dosavadní výsledky ovšem ukazují, že odhady očekávané hodnoty a rozptylu, které fastMCD poskytuje, jsou dostatečně dobré. MCD je sice od začátku spíše heuristická myšlenka, nalézá však velké využití v mnohých aplikacích, např. v ekonometrii, medicíně, analýze obrazu a jiných.

Následuje stručný přehled jednotlivých kapitol.

První kapitola uvede základní statistické pojmosloví potřebné v práci a bude se dál podrobněji zabývat problémem robustní statistiky. Zavede veličiny měřící kvalitu robustnosti a vysvětluje, čím je hledání odlehlých hodnot složité.

Na samotný MCD estimátor se zaměří druhá kapitola. První dvě části se věnují teorii a popisu estimátoru včetně jeho vlastností. Ukáže se zde, že MCD je skutečně vysoce robustním estimátorem s dobrými vlastnostmi, ale který je obecně velmi drahý na výpočet. Poslední část kapitoly se proto zabývá konkrétní algoritmizací fastMCD, která umožňuje efektivní hledání MCD estimátoru. Přináší popis tzv. C-stepu a všech urychlení a vylepšení, která dnes v algoritmu jsou a shrnuje je do pseudokódu.

Numerickými vlastnostmi se podrobně zabývá třetí kapitola. Pokud víme, zatím nebyl v literatuře ucelený, podrobný přehled numerických vlastností uveden. Kapitola proto obsahuje paměťové náklady, poskytuje přehled výpočetních nákladů jednotlivých součástí C-stepu a zkoumá stabilitu jednotlivých částí algoritmu. Zaměřuje se i na možné úpravy algoritmu a jejich dopad na numerické vlastnosti. Konstatuje, že na úkor stability dokážeme snížit výpočetní náklady, což by v některých aplikacích mohlo být žádoucí.

V neposlední řadě obsahuje práce numerické experimenty. Každý se zaměřuje na trochu jinou problematiku.

Ukážeme, že i ve chvíli kdy najdeme přesné řešení estimátoru, neznamená

to nutně, že najdeme všechny odlehlé hodnoty, nebo že naopak některé správné hodnoty neoznačíme za odlehlé. Některé mírné odlehlé hodnoty jsou jednoduše příliš blízké k většině dat a algoritmus je nedetekuje.

U největších dat zkusíme přímočaře snížit počet kroků algoritmu a zmenšit tím výpočtovou náročnost. Příklad bude zkoumat k jakému zhoršení při hledání estimátoru dojde a zda by skutečně nešlo program takto urychlit. Pokusíme se ukázat, ža často lze snížit počet C-stepů, aniž bychom dostali výrazně horší odhady.

Nakonec vyslovíme novou myšlenku, jak snížit počet C-stepů. Pokusíme se ji začlenit do stávajícího algoritmu a budeme sledovat, zda dojde ke snížení výpočtové náročnosti a vylepšení algortimu.

V poslední kapitole shrneme zjištěné výsledky, zhodnotíme možnosti urychlení a úprav algoritmu fastMCD a formulujeme několik závěrů.

1. Úvod do robustní statistiky

1.1 Základní pojmy

V této části postupně definujeme základní statistické pojmy, potřebné v dalším textu.

Definice 1 (Náhodná veličina a náhodný vektor)**.** *Nechť Ω značí množinu (elementárních) jevů, na které existuje systém podmnožin $\mathcal{A}$, a nechť $\mathcal{P}$ je pravděpodobnostní funkce na množině $\mathcal{A}$. Potom jako náhodnou veličinu nazveme reálnou funkci ξ: $\Omega \to \mathbb{R}$, která jevu přiřadí číselnou hodnotu.*

Náhodným vektorem je potom reálná funkce $\Xi = (\xi_1, \ldots, \xi_p) : \Omega \to \mathbb{R}^p$, která jevu přiřazuje p různých vlastností vyjádřených čísly.

Náhodná veličina ξ je hodnota jevu, kterou je možné opakovaně měřit a která nabývá různých hodnot. Náhodná veličina může například být počet ok hozených na šestistěnné kostce. Náhodným vektorem je pak několik různých parametrů naměřených ke stejnému objektu. Příkladem může být měření výšky, váhy a věku u náhodného člověka.

Definice 2 (Relativní četnost)**.** *Nechť n je celkový počet náhodných pokusů, v nichž se hodnota x_i objevila $n_i$krát, potom relativní četnost f_i se rovná $f_i = \frac{n_i}{\sum_i n_i} = \frac{n_i}{n}$.*

Od relativní četnosti očekáváme, že když $n \to \infty$ bude hodnota f_i konvergovat k *pravděpodobnosti* tohoto x_i.

Definice 3 (Pravděpodobnost)**.** *Je-li Ω neprázdná množina, $\mathcal{A}$ je systém podmnožin, náhodných jevů, definovaných na Ω, pak pravděpodobností se nazývá reálná funkce (nebo obecněji míra) $\mathcal{P}$ definovaná na $\mathcal{A}$, která splňuje:*

1. *$\mathcal{P}(A) \geq 0$, pro každý jev $A \in \mathcal{A}$*
2. *$\mathcal{P}(\Omega) = 1$, $\mathcal{P}(\emptyset) = 0$*
3. *$\mathcal{P}(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} \mathcal{P}(A_i)$, pro každé $A_1, A_2, \cdots \in \mathcal{A}$ po dvou disjunktní množiny.*

Definice 4 (Distribuční funkce)**.** *Nechť ξ je náhodná veličina, potom její distribuční funkci F_ξ definujeme vztahem $F_\xi(x) = \mathcal{P}(\xi \leq x)$, $x \in \mathbb{R}$. Pokud není potřeba zdůraznit, k jaké náhodné veličině funkce patří, píše se pouze F.*

Z této definice vyplývají následující vlastnosti funkce F. Distribuční funkce je vždy neklesající, $\lim_{x\to-\infty} F(x) = 0$, $\lim_{x\to\infty} F(x) = 1$, dále F je zprava spojitá a může mít spočetně mnoho bodů nespojitosti, tzv. skoků.

Definice 5 (Diskrétní rozdělení a distribuční funkce)**.** *Rozdělení pravděpodobnosti se nazývá diskrétní, pokud existuje konečná nebo spočetná posloupnost navzájem různých čísel $\{x_i\}_{i\in\mathbb{N}_0}$, kde $\mathbb{N}_0$ je podmnožina přirozených čísel, která tvoří celou množinu Ω a k ní existuje odpovídající posloupnost pravděpodobností*

$\{p_i\}_{i\in\mathbb{N}_0}$ *taková, že* $\sum_{i\in\mathbb{N}_0} p_i = 1$. *Potom pravděpodobnost, že náhodná veličina* ξ *nabývá hodnoty* x_i *je:* $\mathcal{P}(\xi = x_i) = p_i$.

Ekvivalentně můžeme diskrétní rozdělení definovat pomocí distribuční funkce: $F(x) = \sum_{i;x_i\leq x} p_i$.

Diskrétní distribuční funkce je proto schodovitá, se skoky o velikosti p_i v bodech x_i.

Definice 6 (Střední hodnota a rozptyl diskrétního rozdělení)**.** *Střední hodnota nebo také očekávaná hodnota, značena* $\mathcal{E}$*, náhodné veličiny* ξ *je vážený průměr daného rozdělení.*

Pro diskrétní rozdělení je střední hodnota rovna $\mathcal{E}\xi = \sum_{i\in\mathbb{N}_0} x_i \cdot p_i$. *Rozptyl* var *je definován jako* $\mathrm{var}\xi = \mathcal{E}(\xi - \mathcal{E}\xi)^2 = \sum_{i\in\mathbb{N}_0}(x_i - \mathcal{E}\xi)^2 \cdot p_i$.

Definice 7 (Spojité rozdělení)**.** *Spojité rozdělení pravděpodobnosti náhodné veličiny* ξ *je takové rozdělení, které má spojitou distribuční funkci* F_ξ.

Definice 8 (Absolutně spojitá distribuční funkce a hustota)**.** *Distribuční funkce* F *se nazývá absolutně spojitá, pokud k ní existuje nezáporná borelovsky měřitelná funkce* f *taková, že* $F(x) = \int_{-\infty}^{t} f(t)\,\mathrm{d}t$. *Taková funkce* f *se nazývá hustota daného rozdělení.*

Za určitých podmínek existuje pro spojité distribuční funkce F i jejich inverzní funkce, tzv. kvantilová funkce, a hodnoty kvantilové funkce se nazývají kvantily.

Definice 9 (α-kvantil)**.** *Mějme libovolné rozdělení se spojitou a ryze monotónní distribuční funkcí* F *a nechť* $0 < \alpha < 1$, *potom číslo* x *takové, že* $F(x) = \alpha$ *se nazývá* α*-kvantil*[1].

Definice 10 (Absolutně spojité rozdělení)**.** *Absolutně spojité rozdělení je takové rozdělení, jehož distribuční funkce* F *je absolutně spojitá. Ekvivalentně je takové rozdělení jednoznačně dáno svou funkcí hustoty* f.

Definice 11 (Střední hodnota a rozptyl absolutně spojitého rozdělení)**.** *Mějme náhodnou proměnnou* ξ *s absolutně spojitým rozdělením s hustotou* $f(x)$. *Potom definujeme střední hodnotu jako* $\mathcal{E}\xi = \int_{\mathbb{R}} t f(t)\mathrm{d}t$ *a rozptyl jako* $\mathrm{var}\xi = \mathcal{E}(\xi - \mathcal{E}\xi)^2$.

Rozptyl se také dá vyjádřit v následujícím tvaru, který využijeme v další kapitole.

Lemma 1. *Pro libovolnou náhodnou veličinu* ξ *platí:* $\mathrm{var}\xi = \mathcal{E}\xi^2 - (\mathcal{E}\xi)^2$.

Důkaz. $\mathrm{var}\xi = \mathcal{E}(\xi - \mathcal{E}\xi)^2 = \mathcal{E}(\xi^2 - 2\xi\mathcal{E}\xi + (\mathcal{E}\xi)^2) = \mathcal{E}\xi^2 - 2\mathcal{E}(\xi\mathcal{E}\xi) + \mathcal{E}(\mathcal{E}\xi)^2 = \mathcal{E}\xi^2 - 2(\mathcal{E}\xi)^2 + (\mathcal{E}\xi)^2 = \mathcal{E}\xi^2 - (\mathcal{E}\xi)^2$.

□

[1] Později definová normální, χ^2 i Fisher-Snedecorovo rozdělení mají spojité a ryze monotónní funkce a existuje pro ně α-kvantil.

Příklad. Vezměme jako příklad šestistěnnou hrací kostku, na které padne každé ze šesti čísel s pravděpodobností 1/6. Distribuční funkce bude po částech konstantní $F(x) = \frac{\lfloor x \rfloor}{6}$ pro $1 \leq x \leq 6$, bude nulová pro $x < 1$ a bude se rovnat 1 pro $x > 6$. Střední hodnota $\mathcal{E}\xi = 1/6 \cdot (1 + 2 + 3 + 4 + 5 + 6) = 3.5$ nám říká následující: pokud budeme házet kostkou n-krát po sobě, očekávaná hodnota na kostce v jednom hodu bude 3.5 a celkově naházenou hodnotu můžeme očekávat kolem $3.5 \cdot n$.

Pokud nám v n náhodných hodech padla hodnota $x_1 = 1$ celkem n_1-krát, $\ldots$, $x_6 = 6$ padla n_6-krát, pak průměrná hodnota v jednom hodu je rovna $\frac{1 \cdot n_1 + 2 \cdot n_2 + \cdots + 6 \cdot n_6}{n}$, pro $n \to \infty$ se průměrná hodnota bude limitně blížit střední hodnotě $\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{6} n_i \cdot x_i = \sum_{i=1}^{6} x_i \cdot \lim_{n\to\infty} \frac{n_i}{n} = \sum_{i=1}^{6} x_i \cdot p_i = 3.5$.

Pokud podobně vezmeme spojitý případ, např. náhodně generované reálné číslo z intervalu [1, 6], dostaneme distribuční funkci $\tilde{F}(x) = \begin{cases} 0 & x < 1 \\ \frac{x-1}{5} & 1 \leq x < 6 \\ 1 & x \geq 6 \end{cases}$, která je spojitou verzí funkce F a hustota $f(x)$ se pak bude rovnat 1/5 na intervalu [1, 6], jinak bude nulová. Střední hodnota vyjde stejně jako v diskrétním případě $\mathcal{E}\xi = \int_{\mathbb{R}} x f(x) \mathrm{d}x = \int_1^6 1/5 \cdot x \mathrm{d}x = 3.5$.

Lze tedy vidět, že pokud budeme mít náhodný (diskrétní) výběr n hodnot z nějakého spojitého rozdělení $\tilde{F}$, měl by s rostoucím n dobře aproximovat distribuci a veličiny jako je střední hodnota, pokud je tento výběr naměřený přesně.

Definice 12 (Střední hodnota a varianční matice mnohorozměrného rozdělení)**.** *Mějme náhodný vektor $\xi = (\xi_1, \ldots, \xi_p)$ z nějakého absolutně spojitého rozdělení. Potom definujeme střední hodnotu náhodného vektoru jako $\mu = (\mu_1, \ldots, \mu_p)$, kde $\forall i$ $\mu_i = \mathcal{E}\xi_i$ se dá určit z jednorozměrné střední hodnoty z definice 11.*

$\Sigma \in \mathbb{R}^{p \times p}$ je varianční matice, právě když má na (i,j)-tém prvku kovarianci[2] *$\Sigma_{ij} = \mathrm{cov}(\xi_i, \xi_j) = \mathcal{E}[(\xi_i - \mu_i)(\xi_j - \mu_j)]$.*

Pro varianční matici platí, že je vždy pozitivně semidefinitní a symetrická.

Pravděpodobnostní rozdělení, se kterými budeme v textu pracovat především, jsou následující.

Definice 13 (Jednorozměrné normální rozdělení)**.** *Normální rozdělení $N(\mu, \sigma^2)$, kde μ je střední hodnota a σ^2 je rozptyl, je absolutně spojité rozdělení definováno hustotou $f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma^2}$ pro $x \in \mathbb{R}$, $\mu \in \mathbb{R}$ a $\sigma^2 > 0$.*

Definice 14 (Rozdělení chí kvadrát)**.** *Mějme p nezávislých náhodných veličin ξ_1, ξ_2, $\ldots$, ξ_p z $N(0,1)$ rozdělení, potom definujeme χ_p^2 o p stupních volnosti jako rozdělení veličiny $\zeta = \sum_{i=1}^{p} \xi_i^2$, jejíž hustota je definována*

$$f(x) = \begin{cases} 0 & x \leq 0 \\ \frac{x^{n/2-1}}{2^{n/2}\Gamma(n/2)} e^{-x/2} & x > 0 \end{cases},$$

kde $\Gamma(n/2)$ je gama funkce[3]*.*

[2] Z toho důvodu se setkáváme i s označením Σ jako *kovarianční* matice, na druhou stranu má tato matice na diagonále hodnoty rozptylu - variance.

[3] Gama funkce se vypočítá jako $\Gamma(n/2) = \int_0^\infty x^{n/2-1} e^{-x} \mathrm{d}x$ pro $n > 0$.

Definice 15 (Fisher-Snedecorovo rozdělení)**.** *Fisher-Snedecorovo rozdělení $F_{\alpha,n,m}$ je distribuční funkce veličiny $\tau = \frac{\frac{1}{n}\upsilon}{\frac{1}{m}\zeta}$, kde υ a ζ jsou nezávislé veličiny z χ^2_n, resp. χ^2_m. Hustota je definována*

$$f_{n,m}(t) = \begin{cases} 0 & t \leq 0 \\ \frac{\Gamma((n+m)/2)}{\Gamma(n/2)\Gamma(m/2)}(\frac{n}{m})^{n/2}t^{n/2-1}(1+\frac{n}{m}t)^{-(n+m)/2} & t > 0 \end{cases}.$$

Definice 16 (Mnohorozměrné normální rozdělení)**.** *Náhodný vektor dimenze p $\xi = (\xi_1, \ldots, \xi_p)$ má normální rozdělení $N(\mu, \Sigma)$, právě když $\mu \in \mathbb{R}^p$ je střední hodnota ξ a $\Sigma \in \mathbb{R}^{p\times p}$ je varianční matice. Hustota má pro mnohorozměrné normální rozdělení tvar*

$$f(\xi_1, \ldots, \xi_p) = \frac{1}{\sqrt{(2\pi)^p \det \Sigma}} \exp(-\frac{1}{2}(\xi - \mu)^T \Sigma^{-1}(\xi - \mu)),$$

kde $\det \Sigma$ *značí determinant matice* Σ*.*

Hlavní úlohou statistiky je určení parametrů charakterizujících neznámé rozdělení pomocí náhodného výběru z celé populace. Tento výběr je obvykle poměrně malý vzorek populace, který by ovšem měl dostatečně dobře reprezentovat vlastnosti rozdělení.

Definice 17 (Náhodný výběr)**.** *Náhodným výběrem nazveme n-tici náhodných veličin $\xi_1, \ldots, \xi_n$ stejně rozdělených s distribuční funkcí F, které jsou nezávislé. To znamená, že naměřením hodnoty x_i náhodné veličiny ξ_i neovlivníme hodnotu ostatních náhodné veličiny. Konkrétní hodnota náhodného výběru je tedy sada již naměřených pozorování $x_1, \ldots, x_n$.*

Dále už budeme pro zjednodušení vždy chápat náhodný výběr jako už naměřenou realizaci náhodných veličin $x_1, \ldots, x_n$.

Náhodný výběr se pro mnohorozměrné rozdělení označuje také n-tice náhodných vektorů $x_i = (x_{i1}, \ldots, x_{ip})^T$, $i = 1, \ldots, n$, kde p je počet sledovaných veličin. Pak je výběr reprezentován maticí $X = [x_1, \ldots, x_n]^T \in \mathbb{R}^{n\times p}$.

1.2 Robustní statistika

Předpokládejme následující úlohu: chceme zjistit očekávanou hodnotu a varianční matici neznámého rozdělení pomocí náhodného výběru, který máme v matici $X = [x_1, \ldots, x_n]^T$ velikosti $n \times p$ kde $x_i = (x_{i1}, \ldots, x_{ip})^T$ je i-tý náhodný vektor s p proměnnými a $n > p$. Předpokládejme, že všechny vektory x_i jsou nezávislé, což znamená, že jejich hodnoty se při výběru navzájem nijak neovlivňují a že jsou vybrané ze stejného *přibližně* normálního rozdělení, čímž myslíme, že je rozdělení blízké normálnímu. Například, že je normální rozdělení kontaminované jiným rozdělením.

Definice 18 (Kontaminovaná distribuční funkce)**.** *Mějme dvě distribuční funkce F a G. Pak řekneme, že F_ϵ je distribuce kontaminovaná distribuční funkcí G v poměru ϵ, pokud existuje $0 < \epsilon < 1$ takové že $F_\epsilon(G) = (1-\epsilon)F + \epsilon G$.*

V našich datech se ale mohou objevit hodnoty, které se výrazně odlišují od ostatních. Mohou představovat výjimečná, avšak správná data, nebo to mohou být takzvané odlehlé hodnoty. Ty se objevují v důsledku chyb, například kvůli samotnému měření (chyba v přístrojích), lidskému faktoru, nebo jsme mohli například měřit i jiné objekty (například při měření záření vesmírných objektů nemusíme vědět, jestli pozorujeme hvězdu nebo vzdálenou galaxii). Odlehlé hodnoty vybočují z ostatních, ať už jsou extrémně vysoké, nebo se pohybují kolem nuly. Jsou tedy v nějakém slova smyslu abnormální, přičemž co je abnormální se mění v závislosti na dané úloze. Tím, že se hodnoty nacházejí daleko od ostatních, správně naměřených hodnot, ovlivňují výpočty průměru a rozptylu natolik, že naše řešení nemusí vůbec připomínat to skutečné.

Od metody výpočtu budeme požadovat především tzv. *robustnost*, tedy co možná největší nezávislost metody na odlehlých hodnotách. Tato definice robustnosti je běžná ve statistice, naproti tomu existuje i numerická definice robustnosti. Ta říká, že numerické metoda je robustní, pokud ji lze použít na nějakou celou třídu úloh, ne pouze pro konktrétní případ. Dalo by se říct, že jde o „univerzální“ metodu pro řešení jistých typů úloh.

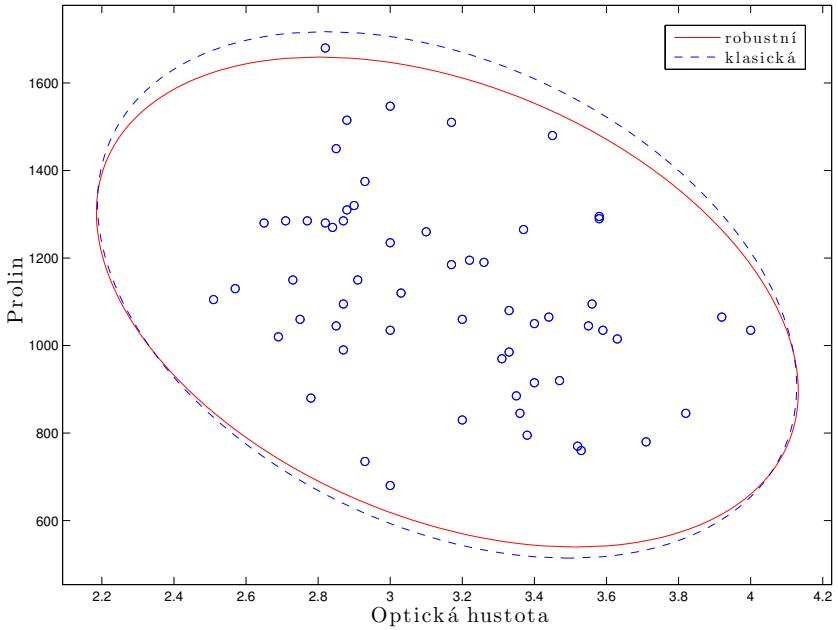
Robustní statistika v nejširším slova smyslu, tak jak je popsána v knize Hampela, Ronchettiho, Rousseeuwa a Stahela [1], se zabývá problémem jakési idealizace dat. Častokrát u metod předpokládáme konkrétní tvar rozdělení dat a vlastnosti, jako například normalitu rozdělení nebo nezávislost jednotlivých měření, aniž bychom nutně měli jistotu, že jsou tyto podmínky splněny. Musíme si vždy dát pozor, jestli můžeme takové požadavky na data klást. Pokud se naše data pouze přibližují našemu modelu, nemusíme nutně dostat očekávané výsledky, ale přinejlepším slušné odhady řešení. Robustní metody se tedy snaží zpracovat statistická data s přihlédnutím k faktu, že ve skutečnosti se měřené veličiny chovají podle našich modelů jen přibližně a měly by být rozumně efektivní.

V užším významu jde hlavně o nezávislost na odlehlých hodnotách. Pokud se na odlehlé hodnoty podíváme z pohledu distribucí, jde o problém tzv. dlouhých chvostů, kdy mají data více okrajových hodnot než kolik by podle předpokládané distribuční funkce měly mít. Odlehlé hodnoty chceme najít a případně řešit zvlášť, snížit jejich vliv na výpočet, nebo pro další výpočty nevyužít. Až poté má smysl hledat řešení problému. Následující příklad bude ilustrovat, co mohou odlehlé hodnoty udělat s naším řešením.

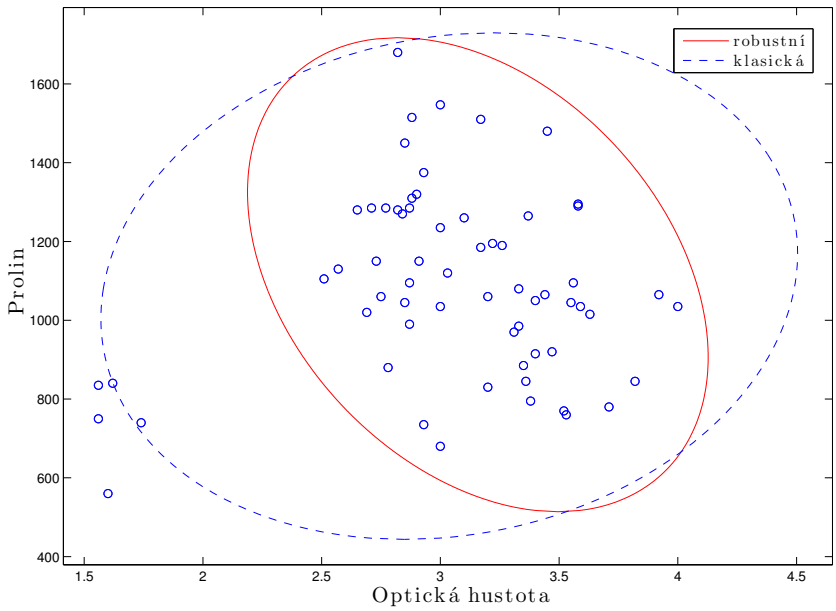
Příklad. Z dat vín [4] vezmeme dvě sady dat. Obě sady obsahují 59 vín druhu Barolo a parametry optická hustota a prolin, druhá má pak navíc 5 vín druhu Barbera se stejnými parametry. Data jsou vykreslena v MATLABu pomocí funkce *mcdcov*, kterou lze sehnat v rámci knihovny LIBRA [5].

Na Obrázku 1.1(a) jsou obě 97,5% toleranční elipsy, klasická i robustní, velmi podobné. Definice toleranční elipsy bude podána v sekci 1.4. Prozatím si stačí představit α-toleranční elipsu jako odhad elipsy obsahující α % všech hodnot skutečného, ale neznámého rozdělení. Robustní elipsa nalezla pouze jednu velmi mírnou odlehlou hodnotu. Co se týče těchto dvou parametrů, jsou všechna měřená vína v jistém smyslu podobná.

Oproti tomu v druhém případě Obrázku 1.1(b) je jasně vidět druhá skupina 5 vín, která má chemicky jiné vlastnosti. Zatímco klasická elipsa je těmito hodnotami hodně ovlivněná, snaží se je zahrnout do řešení a vlastně jen jedna hodnota



(a) jednodruhová data



(b) kontaminovaná data

Obrázek 1.1: Klasická a robustní elipsa pro konzistentní a kontaminovaná data

se výrazněji vymyká, robustní elipsa se téměř nezměnila od prvního případu a s jistotou všech 5 hodnot označuje za odlehlé hodnoty.

Faktem je, jak se argumentuje v [1], že většina dat se bez hrubých chyb neobejde. Existují případy, kdy se jejich počet pohybuje nízko, dejme tomu kolem 1 %, běžně se ale množství chyb pohybuje spíše okolo 10–20 %. Vzhledem k tomu, že se obvykle dá předpokládat, že se experiment chybám nevyhnul, můžeme poměrně s jistotou říci, že většina vybočujících dat v příkladu výše jsou skutečně špatně naměřené odlehlé hodnoty, jak tomu i je, jak v tomto příkladě naštěstí víme.

V ideálním případě bychom chtěli na výpočet střední hodnoty a varianční matice robustní metodu, která odhalí odlehlé hodnoty a z výpočtu je buď vynechá, nebo sníží jejich vliv na výpočet oproti ostatním hodnotám. V případě, který je na Obrázku 1.1(b) výše, lze samozřejmě vidět, že část dat je zatížená chybou. V reálných problémech ovšem můžeme počítat s mnohorozměrnými náhodnými vektory a s obrovskými sadami těchto vektorů. Když vezmeme značení ze začátku úvodu, stačilo by klidně už $n > 1000$ a $p > 3$ a taková data bude v podstatě nemožné projít ručně, abychom našli odlehlé hodnoty, protože už nebudou na první pohled jiná než ostatní ani nepůjdou snadno vizualizovat. Odlehlé hodnoty nemusí totiž mít některý z parametrů napádně jiný než ostatní pozorování, mohou být odlehlé ve směru nějaké kombinace parametrů, které všechny samy za sebe nejsou nápadné. Robustnost je tedy o to žádanější vlastnost, čím máme komplexnější a větší data.

1.3 Měření robustnosti

Existuje několik nástrojů pro měření robustnosti metod, jako například bod selhání a influenční funkce.

1.3.1 Bod selhání

Bod selhání je hodnota, která udává, jaký počet pozorování by musel být nahrazen libovolně velkými hodnotami tak, aby metoda selhala, tj. aby počítané výsledky šly do nekonečna, respektive do nuly.

Definice 19. *Nechť n značí celkový počet pozorování a m je minimální počet libovolně změněných hodnot, s nimiž metoda selhává. Potom bod selhání můžeme definovat jako zlomek $\frac{m}{n}$, případně se dá reprezentovat procentuálně.*

Požadavek na robustnost se pak dá interpretovat jako požadavek na co největší bod selhání a závisí nejen na metodě samotné, ale i na úloze na kterou ji používáme.

Maximální možná hodnota, které může bod selhání dosáhnout, je přibližně 1/2, přesněji pro afinně ekvivariantní metody jako je Minimum covariance determinant (MCD, viz. Kapitola 2) je maximální bod selhání $\lfloor \frac{n-p+2}{2n} \rfloor$ [6]. Jde o intuitivní omezení, protože ve chvíli kdy většina dat budou odlehlé hodnoty, metoda nedokáže rozlišit, která data jsou odlehlá a která ne.

Příklad. Jako jednoduchý příklad si uveďme následující situaci. Nechť máme náhodný výběr sestávající se z hodnot $a_1 = 2.8$, $a_2 = 3.6$, $a_3 = 4.1$, $a_4 = 5.9$, $a_5 = 6.6$ a $a_6 = 9$. Výběrový průměr, což je odhad střední hodnoty, vypočítáme jako $\frac{1}{6}\sum_{i=1}^{6} a_i$ a je tedy roven 5.3. Pokud by ale poslední z hodnot rostla, dejme tomu na 20, 100 a 1000, budeme dostávat postupně průměry 7.1, 20.5 a 170.5. Lze snadno nahlédnout, že stačí aby jediná z hodnot byla špatná a výběrový průměr pak může vzrůst na libovolně velkou hodnotu. Bod selhání je tedy v případě průměru obecně $1/n$, kde n je celkový počet pozorování.

Pro výběrový medián je situace úplně jiná. Ten dosahuje maximálního možného bodu selhání 1/2. Medián je zjednodušeně řečeno hodnota ležící uprostřed podle velikosti seřazené posloupnosti pozorování. Přesněji platí, že nejméně polovina dat je menších nebo rovných mediánu a nejméně 50 % dat je větších nebo rovných mediánu. Abychom dostali medián nekonečný, museli bychom nahradit alespoň polovinu dat nekonečnými hodnotami. Pro $a_1 = 2.8$, $a_2 = 3.6$, $a_3 = 4.1$, $a_4 = 5.9$ a $a_5 = 6.6$ je medián roven $a_3 = 4.1$, musíme změnit hodnoty a_3, a_4 i a_5. Pokud máme i pozorování $a_6 = 9$, medián se obvykle bere jako $\frac{a_3+a_4}{2} = 5$ a stačilo by také nahradit poslední 3 hodnoty, aby medián selhal. Bod selhání je proto $\frac{1}{n}\lfloor\frac{n+1}{2}\rfloor \approx \frac{1}{2}$.

Zde uvedenou definici bodu selhání lze najít např. ve skriptech [7] nebo v článku [6]. Naproti tomu se často uvádí i definice, kde bod selhání je zlomek dat, který po nahrazení libovolnými hodnotami *ještě nezpůsobí* selhání metody, ale přidáním další odlehlé hodnoty by už k selhání došlo. Tuto definici lze najít v [1] a dalších. V této definici by byl např. bod selhání výběrového průměru 0.

1.3.2 Influenční funkce

Druhou uvedenou charakteristikou odhadu je influenční funkce, která vypovídá o vlivu malé kontaminace distribuce na náš odhad. Zde se budu držet zavedení podle již zmiňované knihy Robust Statistics [1]. Mějme data z nějaké distribuce F, z nichž chceme odvodit hodnotu parametru θ, který patří do nějaké parametrické rodiny $\Theta \subseteq \mathbb{R}$. Často lze chápat θ jako veličinu závislou na distribuci, jako například střední hodnotu, s příslušnou funkcí T_n náhodného výběru $x = (x_1, \ldots, x_n)$, kdy potom $\theta \approx T_n(x_1, \ldots, x_n)$. T_n je tedy vlastně nějakým estimátorem parametru θ.

Definice 20 (Empirická influenční funkce)**.** *Nechť $x = (x_1, \ldots, x_n)$ je náhodný výběr a T_n je estimátor odhadující nějaký parametr θ. Pokud nahradíme pozorování x_i libovolnou hodnotou y, pak definujeme empirickou influenční funkci v bodě i jako*

$$EIF_i(y) = n \cdot (T_n(x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_n) - T_n(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n)).$$

Empirická influenční funkce měří, jak se změní odhad parametru θ pomocí nějakého estimátoru T_n, pokud se nám libovolně změní hodnota v jednom z pozorování. Závisí tudíž pouze na datech a změně výběru, ale nebere v potaz z jaké distribuce výběr je. Přístup influenční funkce je proto takový, že zjišťujeme co se stane s odhadem parametru, když se trochu změní distribuce dat.

Předpokládejme konkrétní tvar rozdělení dat, daný distribuční funkcí F a mějme nějakou jinou distribuční funkci G. Influenční funkce, na rozdíl od té empirické, zkoumá chování T_n, když nejsou data přesně z distribuce dané funkcí F, ale když je distribuce dat trochu ovlivněna distribuční funkcí G. Máme pak tzv. distribuční funkci F kontaminovanou distribuční funkcí G v poměru $0 < \epsilon < 1$ jako $F_\epsilon(G) = (1-\epsilon)F + \epsilon G$.

Nechť za G vezmeme distribuční funkci generovanou Diracovou mírou δ_y, pro $y \in \mathbb{R}$, která je definovaná $G_y(t) = \begin{cases} 0 & t < y \\ 1 & t \geq y \end{cases}, t \in \mathbb{R}$. Vracíme se tím ke kontaminaci v jednom bodě y. Jak za chvíli uvidíme, influenční funkce měří míru změny odhadu při kontaminaci jednou odlehlou hodnotou.

Definice 21 (Influenční funkce)**.** *Influenční funkci estimátoru T pro rozdělení dat definovaného distribuční funkcí F nazveme*

$$IF(y; T, F) = \lim_{\epsilon \to 0} \frac{T((1-\epsilon)F + \epsilon G_y) - T(F)}{\epsilon},$$

kde G_y je výše definovaná distrubuční funkce.

Vlastností, kterou především očekáváme od dobré influenční funkce, je její omezenost. Pokud by totiž mohla influenční funkce v nějakém bodě y nabývat nekonečné hodnoty, znamenalo by to, že by i příslušný estimátor mohl dosáhnout nekonečna pro některá pozorování.

1.4 Určení odlehlých hodnot

Jednou z klasických možností detekce odlehlých hodnot je použití Mahalanobisovy vzdálenosti, která ovšem není příliš vhodná, jak dál ukážeme. K Mahalanobisově vzdálenosti potřebujeme ještě tyto definice.

Definice 22 (Výběrový průměr a výběrová varianční matice)**.** *Mějme náhodný výběr $X = [x_1, \ldots, x_n]^T$ a pro každé $i \in \{1, \ldots, n\}$ x_i má p proměnných, potom definujeme výběrový průměr jako $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ a výběrovou varianční matici jako $V = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^T$.*

Výběrový průměr je odhadem střední hodnoty, stejně tak výběrová varianční matice je odhad varianční matice. Pokud nebude řečeno jinak, budeme dále vždy předpokládat, že je V pozitivně definitní.

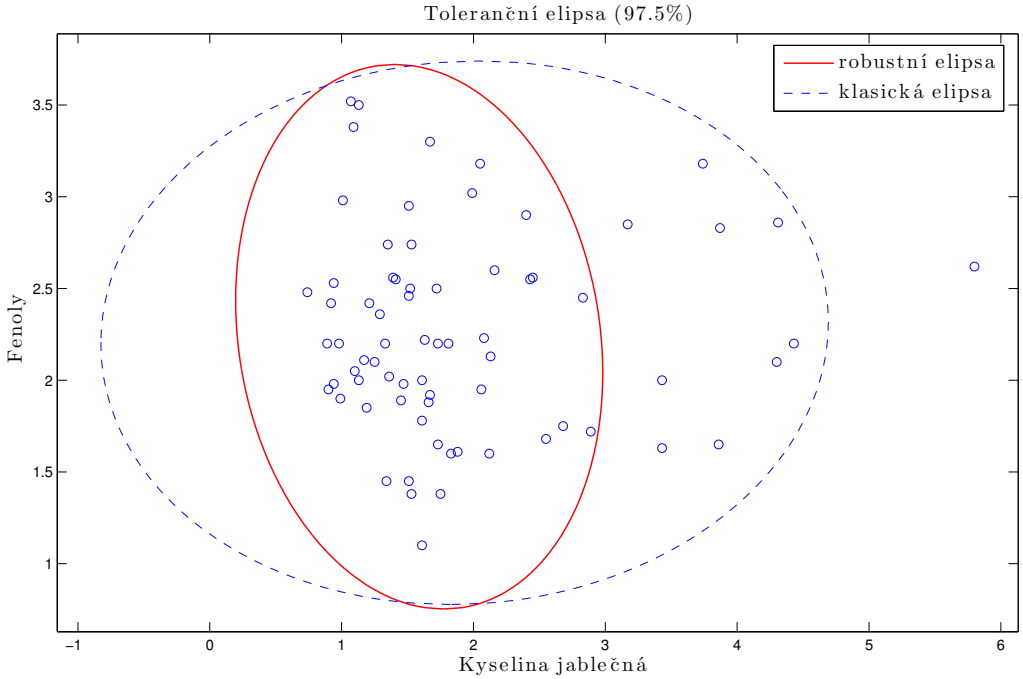
Definice 23 (Mahalanobisova vzdálenost)**.** *Mahalanobisovu vzdálenost náhodného výběru x_i definujeme vzorcem $d(x_i) = \sqrt{(x_i - \bar{x})^T V^{-1}(x_i - \bar{x})}$, kde $\bar{x}$ je výběrový průměr a V je výběrová varianční matice.*

V případě klasické tolerance považujeme vektor x_i za odlehlý, pokud jeho vzdálenost $d^2(x_i)$ je větší než 97,5% kvantil pro χ^2 rozdělení o p stupních volnosti, který značíme $\chi^2_{p,0.975}$.

Toleranci, podle které určujeme odlehlé hodnoty ze vzdálenosti lze definovat i jinak, např. přes Fisher-Snedecorovo rozdělení $F_{\alpha,n,m}$ s n a m stupni volnosti a pro nějaký α-kvantil.

Pokud je odlehlých hodnot více než jedna, může pak být Mahalanobisova vzdálenost těmito hodnotami hodně ovlivněna a pokud pro dvourozměrná data vykreslíme klasickou toleranční elipsu (tj. množina náhodných vektorů, jejichž vzdálenost je menší rovno $\chi^2_{2,0.975}$) má tendenci pokrýt všechna data. Stačí například jedna výrazně odlehlá hodnota, která posune odhady střední hodnoty a rozptylu svým směrem, aby se tím skryly jiné, menší odlehlé hodnoty v podobném směru. Působí tak tzv. maskovací efekt.

Jiným způsobem jak nalézt odlehlé hodnoty je přes výpočet robustní vzdálenosti $rd(x_i) = \sqrt{(x_i - m)^T S^{-1}(x_i - m)}$, kde m je nějaký robustní odhad střední hodnoty a S je nějaký robustní odhad varianční matice. Jde vlastně o stejný vzorec, který ale využívá přesnějších nezkreslených odhadů. Takový výpočet by potom měl být rezistentní na maskovací efekt a mnohem lépe odhalit odlehlé hodnoty [2].



Obrázek 1.2: Robustní vs. klasická elipsa a maskovací efekt

Maskovací efekt lze dobře vidět v příkladu výše na Obrázku 1.1 stejně jako na Obrázku 1.2. Zde jsou vykresleny dva parametry vín z dat [4] a to kyselina jablečná a celkové množství fenolů. Pro graf byl použitý druh Grignolino, celkem 71 vín. Zatímco klasické hledání odlehlých hodnot Mahalanobisovou vzdáleností určilo jako odlehlou hodnotu pouze jedno víno, robustní elipsa označila hned devět vín jako možné odlehlé hodnoty. Data naznačují, že označená vína mohou být kyselejší, než ostatní vína tohoto druhu.

Jednou z velmi používáných robustních metod odhadu střední hodnoty a varianční matice v poslední době je estimátor MCD (minimum covariance determinant). Touto metodou se budu v práci dále zabývat, uvedu hlavní myšlenku algoritmu a určím její numerické vlastnosti. Na závěr několika experimenty ukáži použití MCD estimátoru a jeho možné úpravy a urychlení.

2. Estimátor MCD

Původní myšlenka MCD estimátoru střední hodnoty a varianční matice pro elipticky symetrické, unimodální (jednovrcholové) distribuce [2] se poprvé objevila v roce 1984 v článku Petera Rousseeuwa *Least Median of Squares Regression* [8] a byla dál rozvedena ve článku *Multivariate estimation with high breakdown point* [9]. MCD estimátor je sice vysoce robustní (dosahuje bodu selhání $\lfloor \frac{n-p+2}{2n} \rfloor$ [6]), avšak výpočetně dost náročný. Teprve v roce 1999 uvedli Rousseeuw a Van Driessenová [3] algoritmus fastMCD. Ten nám obecně nedá samotný MCD estimátor, ale obvykle jeho *dostatečně dobrou* aproximaci. Algoritmus urychlil hledání MCD estimátoru natolik, aby šel běžně použít i na velká množství dat s hodně parametry.

2.1 Popis MCD estimátoru

Základní idea estimátoru je najít takových h pozorování z n, kde $p+1 \leq h \leq n$, pro která je determinant výběrové varianční matice minimální a z těchto h bodů zjistit výběrový průměr a výběrovou varianční matici, které budou odhadem střední hodnoty a rozptylu. Pro případ dvou nebo tří proměnných si lze ekvivalentně představit, že hledáme elipsu či elipsoid s minimálním možným objemem, který obsahuje h pozorování a odhadem střední hodnoty je pak střed elipsy nebo elipsoidu.

Pro $n/2 < h < n$ jde o odhady, které zanedbávají část dat, které jsou vzdálenější od převážnější a koncentrovanější části dat a tím se snaží zanedbat právě odlehlé hodnoty.

Pokud chceme najít h pozorování z n, pro které je determinant varianční matice minimální, musíme vzít v úvahu všech $\binom{n}{h}$ množin velikosti h, což vede na obrovské množství množin, pro které bychom museli určit výběrové průměry i výběrové varianční matice včetně determinantů. Z toho důvodu je hledání MCD estimátoru výpočetně velmi náročné, má kombinatorickou složitost.

2.2 Vlastnosti MCD estimátoru

Prvně se budeme krátce věnovat již diskutované míře robustnosti.

2.2.1 Robustnost

Jak již bylo řečeno, za hodnotu h se často bere $\lfloor \frac{n+p+1}{2} \rfloor$, protože pak dosahuje MCD estimátor nejvyššího bodu selhání, a to $\lfloor \frac{n-p+2}{2n} \rfloor$. Bod selhání je roven množství dat, které nevyužijeme k výpočtu odhadů.

Velikost množin h můžeme volit kdekoliv mezi $p+1$ a n. Protože ale chceme vědět co nám pozorování celkově říkají o chování a vlastnostech daných dat, nedává příliš smysl vzít h příliš nízké, řekněme pod hodnotu $n/2$. Jistě půjde např. najít $p+1$ pozorování, která budou blízko sebe a budou tak mít malý rozptyl, ale z hlediska celkového objemu dat nemusí mít žádný smysl. Klidně to

mohou i být odlehlé hodnoty. Pokud budeme brát h v rozmezí $\lfloor \frac{n+p+1}{2} \rfloor \leq h \leq n$, bod selhání se bude spojitě měnit od $\lfloor \frac{n-p+2}{2n} \rfloor \approx 1/2$ do 0.

Další navrhovanou hodnotou je například $h = 0.75n$, kterou např. navrhuje Croux a Haesboeck [10]. Pak máme stále hodně robustní estimátor s bodem selhání 1/4, ale přitom dosáhneme vyšší efektivity.

Za nejefektivnější považujeme estimátor, když využije všechna neodlehlá data. Pokud nastavíme hodnotu h níž než je nutné, zvyšujeme sice robustnost vůči odlehlým hodnotám, ale zároveň snižujeme počet dat, která využijeme k estimátoru, a tedy snižujeme efektivitu. V případě, že máme hrubou představu o tom, kolik odlehlých hodnot mohou data obsahovat, můžeme přizpůsobit h k našim datům.

Influenční funkcí MCD estimátoru se také zabývali Croux a Haesboeck v [10]. Funkce se zdá být pro odhad varianční matice omezená, což indikuje, že je estimátor robustní vůči libovolně velké kontaminaci v jednom bodě. Pro odhad střední hodnoty vychází influenční funkce nulová pro taková pozorování, pro která je kvadrát jejich Mahalanobisovy vzdálenosti větší než mezní hodnota $\chi^2_{p,0.975}$, tedy odlehlé hodnoty nemají na odhad střední hodnoty vliv. Stejně tak influenční funkce pro mimodiagonální prvky odhadu varianční matice bude nulová pro odlehlé prvky, ale už to neplatí pro diagonální prvky, kde influenční funkce zůstává pro všechna pozorování konstantní a nenulová. Z toho plyne, že na MCD odhad varianční matice stále mají odlehlé hodnoty nějaký vliv, i když omezený. Všechny influenční funkce jsou spojité až na skok v mezní hodnotě a případně na skok způsobený váhovou funkcí. Přesné odvození influenčních funkcí a z toho i efektivitu MCD estimátoru lze podrobně najít v [10].

2.2.2 Afinní ekvivariance

MCD estimátor má vlastnost afinní ekvivariance, nezávislosti na posun dat a škálování.

Afinní ekvivariance znamená, že pro daná data $X \in \mathbb{R}^{n \times p}$ a pro každou regulární matici $A \in \mathbb{R}^{p \times p}$ a pro každý vektor $b \in \mathbb{R}^p$ platí:

$$m(AX + b) = Am(X) + b$$
$$S(AX + b) = AS(X)A^T,$$

kde m odhaduje střední hodnotu a S odhaduje varianční matici pomocí MCD [2].

Z toho vyplývá, že pokud lineárně transformujeme dané X, příslušně se transformují i odhady a zachová se stejná množina velikosti h, která minimalizuje determinant varianční matice. MCD estimátor tudíž nezávisí na otočení, posunutí nebo přeškálování dat a odhaluje stejné odlehlé hodnoty.

2.3 Algoritmus fastMCD

Jak bylo zmíněno, při výpočtu MCD estimátoru bychom obecně měli brát v úvahu všech $\binom{n}{h}$ množin z nichž hledáme tu množinu, jejíž determinant výběrové varianční matice je minimální. Museli bychom tedy pro každou z množin počítat výběrový průměr, výběrovou varianční matici a její determinant. Pokud se n zvětšuje, poroste velmi rychle náročnost výpočtu a pro velká data by se metoda

nedala efektivně použít. Až uvedení fastMCD algoritmu znamenalo pro využití estimátoru velký obrat, protože nás zbavuje nutnosti vzít všech $\binom{n}{h}$ množin. Místo toho vezme jen omezený počet množin velikosti h a pro každou iterativně zmenšuje determinant výběrové varianční matice, kterému se občas říká zobecněný výběrový rozptyl.

K tomuto zmenšování determinantu využívá iteračního kroku nazvaného C-step, který vezme množinu velikosti h, vypočítá z ní odhady střední hodnoty a varianční matice včetně determinantu a následně vypočítá Mahalanobisovy vzdálenosti všech n pozorování, ovšem pomocí získaných robustních odhadů. C-step dále určí nových h pozorování s nejmenšími vzdálenostmi a ty použije k další iteraci. Podle později uvedené Věty 3 má varianční matice pro novou množinu pozorování menší nebo stejně velký determinant.

Později uvedená Věta 2 nám dává nejen jednoznačnost dvojice (μ, Σ), která minimalizuje determinant varianční matice, ale především nám říká, že tato dvojice je právě skutečná střední hodnota a varianční matice neznámého rozdělení. Pokud tedy najdeme globální minimum determinantu pro nějakou dvojici (ν,Δ) a použili jsme při tom jen relevantní data bez odlehlých pozorování, měli bychom dostat populační střední hodnotu a varianční matici. Díky Větě 3 zase máme zaručen, že když budeme každým krokem snižovat $\det S$, musí takový postup nutně konvergovat v konečném počtu kroků.

Důvod proč zmenšováním determinantu získáváme postupně lepší odhady vyplývá dobře z geometrické představy. Rovnice $d^2 = (x - m)^T S^{-1}(x - m)$, kde m je odhad střední hodnoty a S je odhad varianční matice, definuje body, které mají stejnou vzdálenost od m. V dvourozměrném případě půjde jak bylo řečeno o elipsu, pro $p > 3$ jde o nějaký zobecněný elipsoid, který ovšem neumíme graficky znázornit. Diagonální prvky výběrové varianční matice určují rozptyl dat ve směrech jednotlivých os elipsoidu, které jsou dané vlastními vektory matice S. Délka os pak bude úměrná vlastním číslům a objem elipsoidu je roven $\sqrt{\det S}$. Tudíž pokud by byl determinant varianční matice nulový, ležely by data v nějaké nadrovině.

Obecně je těžké interpretovat zobecněný výběrový rozptyl, protože závisí na seškálování dat. Pokud se ovšem škálování veličin během výpočtů nemění a zmenšujeme-li každým krokem determinant varianční matice, intuitivně zmenšujeme elipsoid a tím i rozptyl dat. Postupně tak zpřesňujeme odhady polohy a rozptylu hlavní části dat.

2.3.1 C-step

Nejdůležitějším krokem fastMCD je C-step. Rousseuw a Van Driessenová [3] pojmenovali tento krok písmenem C od slova concentration, tedy koncentrace, protože každý krok "zhušťuje"varianční matici těchto pozorování a zmenšuje její determinant. Také se vždy soustřeďujeme pouze na h pozorování s nejmenšími vzdálenostmi.

Myšlenka C-stepu popsaná v [3] je následující:

Vezměme naše data $X = [x_1, \ldots, x_n]^T$, kde $\forall i\ x_i = (x_{i1}, \ldots, x_{ip})^T$. Nechť $H_1 \subset \{1, \ldots, n\}$ je nějaká množina h indexů, kde $p + 1 \leq h \leq n$.

- Potom spočítáme z dat s indexy z H_1 výběrový průměr $m_1 = \frac{1}{h} \sum_{i \in H_1} x_i$ a výběrovou varianční matici $S_1 = \frac{1}{h} \sum_{i \in H_1} (x_i - m_1)(x_i - m_1)^T$.

- Pokud $\det(S_1) \neq 0$, spočítáme Mahalanobisovy vzdálenosti všech n pozorování jako $d_1(i) = \sqrt{(x_i - m_1)^T S_1^{-1}(x_i - m_1)}$ pro $i = 1, \ldots, n$.
- Následně srovnáme vzdálenosti $d_1(i)$ podle velikosti a prvních h indexů s nejmenší vzdáleností označíme jako množinu H_2 a iterujeme postup.

Jak již bylo řečeno, tento postup zajišťuje, že $\det(S_1) \geq \det(S_2) \geq \ldots$ a rovnost $\det(S_j) = \det(S_{j+1})$ nastane pouze pokud $S_j = S_{j+1}$ a $m_j = m_{j+1}$, jak bude dokázáno v následující sekci. Proces se zastaví právě když nastane rovnost determinantů ve dvou po sobě jdoucích variančních maticích, nebo pokud v nějakém kroku dostaneme determinant rovný nule.

2.3.2 Důkaz konvergence C-stepu

Podle důkazu R. Grübela [11] nejprve ukážeme, že platí věta o jednoznačnosti dvojice (μ,Σ), která minimalizuje determinant varianční matice a která je právě skutečnou střední hodnotou a varianční maticí rozdělení.

Věta 2. *Nechť $x = (x_1, \ldots, x_p)^T$ je náhodný vektor p proměnných takový, že $\mathcal{E}\|x\|^2 < \infty$, $\mu \in \mathbb{R}^p$ je vektor středních hodnot x ($\mu_i = \mathcal{E}x_i$ pro $i \in \{1, \ldots, p\}$) a $\Sigma \in \mathbb{R}^{p \times p}$ je varianční matice vektoru x ($\Sigma = \mathcal{E}(x-\mu)(x-\mu)^T$). Předpokládejme, že Σ je pozitivně definitní.*

Potom platí, že mezi všemi páry (ν,Δ), kde $\nu \in \mathbb{R}^p$ a $\Delta \in \mathbb{R}^{p \times p}$ je symetrická pozitivně definitní matice, takovými, že splňují rovnost

$$\mathcal{E}(x - \nu)^T \Delta^{-1}(x - \nu) = p \tag{2.1}$$

existuje jediná dvojice, která minimalizuje $\det \Delta$*, a to právě* (μ, Σ).

Důkaz. Nechť $\Sigma^{-1/2}$ je symetrická matice taková, že $\Sigma^{-1/2}\Sigma^{-1/2}\Sigma = I$, kde I je jednotková matice velikosti $p \times p$. Označme $y = \Sigma^{-1/2}(x - \mu)$ a y_i je i-tý prvek y. Symbolem tr používaným v následující rovnici označujeme *trace*, tzv. stopu čtvercové matice, která je součtem prvků na hlavní diagonále matice. V rovnici také několikrát využijeme linearity operátoru $\mathcal{E}$: pro každé dva náhodné vektory x a y platí, že $\mathcal{E}(x + y) = \mathcal{E}x + \mathcal{E}y$ a pro každou náhodnou matici X a matice konstant A a B platí, že $\mathcal{E}(AXB) = A\mathcal{E}(X)B$.

$$\begin{aligned}\mathcal{E}(x-\mu)^T\Sigma^{-1}(x-\mu) = \mathcal{E}y^T y = \sum_{i=1}^{p} \mathcal{E}(y_i^2) = tr((\mathcal{E}y_i y_j)_{i,j=1}^{p}) = tr(\mathcal{E}yy^T) = \\ = tr(\Sigma^{-1/2}\mathcal{E}((x-\mu)(x-\mu)^T)\Sigma^{-1/2}) = tr(I) = p\end{aligned}$$

z čehož plyne, že dvojice (μ, Σ) splňuje rovnost z předpokladu věty.

Díky transformaci výše můžeme předpokládat BÚNO, že x má $\mu = 0$ a $\Sigma = I$, jinak použijeme náhodný vektor y.

Dále nechť ν je libovolný vektor p proměnných, Δ je libovolná symetrická, pozitivně definitní matice velikosti $p \times p$ a (ν, Δ) splňuje rovnost (2.1), potom existuje ortogonální matice U a čísla $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p > 0$ (tj. vlastní čísla matice Δ) tak, že $\Delta = U diag(\lambda_1, \ldots \lambda_p) U^T$, kde $diag(\lambda_1, \ldots \lambda_p)$ je diagonální matice. Determinant matice Δ je pak roven $\det \Delta = \lambda_1 \lambda_2 \cdots \lambda_p$, protože ortogonální transformace nemění determinant.

Označme $z = U^T(x-\nu)$ a $z_i = u_i^T(x-\nu)$ je i-tý prvek z, potom

$$\begin{aligned}\mathcal{E}(x-\nu)^T\Delta^{-1}(x-\nu) &= \mathcal{E}z^T diag(\lambda_1^{-1},\dots,\lambda_p^{-1})z = \mathcal{E}(\lambda_1^{-1}z_1^2+\dots+\lambda_p^{-1}z_p^2) = \\ &= \sum_{i=1}^{p}\mathcal{E}(\lambda_i^{-1}z_i^2) = \sum_{i=1}^{p}\lambda_i^{-1}\mathcal{E}z_i^2 = \\ &= \sum_{i=1}^{p}\lambda_i^{-1}varz_i + \sum_{i=1}^{p}\lambda_i^{-1}(\mathcal{E}z_i)^2,\end{aligned}$$

kde poslední rovnost plyne z Lemma 1 z první kapitoly..

Protože $\mathcal{E}x = 0$, plyne z výše uvedených vztahů a linearity $\mathcal{E}$ pro střední hodnotu a rozptyl z toto:

$$\begin{aligned}\mathcal{E}z &= \mathcal{E}(U^T(x-\nu)) = \mathcal{E}(U^Tx) - \mathcal{E}(U^T\nu) = U^T\mathcal{E}(x) - U^T\mathcal{E}(\nu) = -U^T\nu, \\ varz &= \mathcal{E}(z-\mathcal{E})(z-\mathcal{E})^T = \mathcal{E}(z+U^T\nu)(z+U^T\nu)^T = \\ &= \mathcal{E}(U^Tx - U^T\nu + U^T\nu)(U^Tx - U^T\nu + U^T\nu)^T = \mathcal{E}(U^TxxU) = \\ &= U^T\mathcal{E}(xx^T)U = U^T\mathcal{E}((x-\mathcal{E}x)(x-\mathcal{E}x)^T)U = U^TIU = I.\end{aligned}$$

Ze vztahu (2.1) potom plyne, že součet $\sum_{i=1}^{p}\lambda_i^{-1} < p$ pro $\nu \neq 0$ a tedy z AG nerovnosti $\sqrt[p]{\lambda_1^{-1}\cdots\lambda_p^{-1}} \leq \frac{(\lambda_1^{-1}+\dots+\lambda_p^{-1})}{p}$ plyne $\lambda_1^{-1}\cdots\lambda_p^{-1} < 1$ a tudíž $\det\Delta > 1$. Tedy jakékoliv řešení (2.1) pro $\nu \neq 0$ nemůže minimalizovat $\det\Delta$, protože víme, že dvojice $(0, I)$ je řešením. Pokud $\nu = 0$, potom $\sum_{i=1}^{p}\lambda_i^{-1} = p$, což znovu podle nerovnosti geometrického a aritmetického průměru dává právě dvě možnosti: buď se všechna $\lambda_i^{-1} = 1$ a ve výrazu $\sqrt[p]{\lambda_1^{-1}\cdots\lambda_p^{-1}} \leq 1$ nastává rovnost nebo je součin ostře menší než 1. Druhý případ by znovu neminimalizoval $\det\Delta$ a tedy $\Delta = UIU^T = I$. Z toho plyne, že jediné řešení je $(0, I)$.

□

Tuto větu využijeme k důkazu, že posloupnost determinantů bude klesající podle článku Rousseeuwa a van Driessenové [3]. Ještě jednou zformulujeme, co chceme dokázat.

Věta 3. *Nechť $X = [x_1,\dots,x_n]^T \in \mathbb{R}^{n\times p}$ je náhodný výběr, kde každé x_i má p proměnných. Nechť $H_1 \subset \{1,\dots,n\}$ je nějaká množina indexů velikosti h, kde $p+1 \leq h \leq n$.*

Označme $m_1 = \frac{1}{h}\sum_{i\in H_1}x_i$ a $S_1 = \frac{1}{h}\sum_{i\in H_1}(x_i-m_1)(x_i-m_1)^T$. Pokud $\det(S_1) \neq 0$, spočteme vzdálenosti $d_1(i) = \sqrt{(x_i-m_1)^TS_1^{-1}(x_i-m_1)}$ pro $i = 1,\dots,n$, které srovnáme podle velikosti a prvních h indexů s nejmenší vzdáleností označíme jako množinu H_2 a iterujeme postup.

Potom $\det(S_1) \geq \det(S_2) \geq \det(S_3) \geq \dots$ a rovnost $\det(S_j) = \det(S_{j+1})$ nastane pouze pokud $S_j = S_{j+1}$ a $m_j = m_{j+1}$.

Důkaz. Mějme z postupu výše H_2, m_2 a S_2. Pokud $\det S_2 = 0$, již máme minimum, předpokládejme tedy, že $\det S_2 > 0$. Pomocí m_2 a S_2 spočítejme vzdálenosti $d_2(i)$ pro $i = 1,\dots,n$, pak

$$\frac{1}{hp}\sum_{i\in H_2} d_2^2(i) = \frac{1}{hp}\sum_{i\in H_2}(x_i - m_2)^T S_2^{-1}(x_i - m_2) =$$
$$= \frac{1}{hp} tr \sum_{i\in H_2} S_2^{-1}(x_i - m_2)(x_i - m_2)^T = \frac{1}{p} tr S_2^{-1} S_2 = \frac{1}{p} tr I = \frac{p}{p} = 1.$$

Ekvivalentně lze to samé odvodit i pro H_1, m_1 a S_1. Potom pokud označíme $d_1(\pi(i))$ permutací přerovnanou řadu $d_1(i)$ takovou, že $d_1(\pi(1)) \leq d_1(\pi(2)) \leq \cdots \leq d_1(\pi(n))$, pak z definice H_2 dostaneme

$$0 < \alpha = \frac{1}{hp}\sum_{i\in H_2} d_1^2(i) = \frac{1}{hp}\sum_{i=1}^{h} d_1^2(\pi(i)) \leq \frac{1}{hp}\sum_{j\in H_1} d_1^2(j) = 1.$$

Pro Mahalanobisovy vzdálenosti $d_{(m_1,\alpha S_1)}$ dostaneme následující:

$$\frac{1}{hp}\sum_{i\in H_2} d_{(m_1,\alpha S_1)}^2(i) = \frac{1}{hp}\sum_{i\in H_2}(x_i - m_1)^T(\alpha S_1)^{-1}(x_i - m_1) =$$
$$= \frac{1}{\alpha hp}\sum_{i\in H_2} d_1^2(i) = \frac{\alpha}{\alpha} = 1.$$

Podle Grübelovy věty platí, že ze všech dvojic (m, S) takových, že splňují $\mathcal{E}(x_i - m)^T S^{-1}(x_i - m) = \frac{1}{hp}\sum_{i\in H_2} d_{(m,S)}^2(i) = 1$ právě dvojice (m_2, S_2) minimalizuje $\det S$ a to jednoznačně. Z toho vyplývá, že $\det S_2 \leq \det(\alpha S_1)$, kde $0 < \alpha \leq 1$ a tedy $\det S_2 \leq \det(\alpha S_1) \leq \det S_1$.

Rovnost $\det S_2 = \det S_1$ nastane pouze pokud $\alpha = 1$ a zároveň $\det S_2 = \det(\alpha S_1)$, což z Grübelovy věty znamená, že $(m_2, S_2) = (m_1, \alpha S_1)$, dohromady dostaneme podmínku rovnosti $(m_2, S_2) = (m_1, S_1)$.

□

2.3.3 Počáteční volba množiny H_1

Otázkou zůstává volba počáteční množiny H_1, pro iterační proces založený na Větě 3. Jak bylo uvedeno, fastMCD aplikuje C-step na omezený počet množin H_1. Uvidíme, že urychlení fastMCD vyplývá ze snížení počtu množin H_1 a že správným výběrem těchto množin zvyšujeme robustnost algoritmu. Výpočet estimátoru by měl brát v úvahu všech $\binom{n}{h}$ množin H_1 velikosti h a mezi nimi hledat množinu, pro kterou je determinant varianční matice minimální. Protože je takový postup pro větší n pomalý a hodnota $\binom{n}{h}$ bude obrovská, musíme tedy udělat omezení na počet množin, obvyklý je počet 500[1]. Nejjednodušším přístupem by bylo nagenerování zcela náhodných množin, ale pak se dá očekávat, že tyto množiny obsahují mnoho odlehlých hodnot, které by nám nepříznivě ovlivnily odhady m_1, S_1. Použijeme tedy složitější postup, díky kterému redukujeme množství odlehlých hodnot v počátečních množinách H_1, díky čemuž nabyde C-step a celý algoritmus ze statistického hlediska na robustnosti a sníží se cena výpočtu.

[1] tato hodnota je přímo navrhována v [3] a přednastavena v implementaci algoritmu fastMCD.

Hlavním problémem volby H_1 je skutečnost, že je vysoká pravděpodobnost, že náhodně vybraných h pozorování z n obsahuje odlehlé hodnoty. Číslo h se obvykle bere jako $h = \lfloor \frac{n+p+1}{2} \rfloor$ a je to tudíž více jak polovina všech pozorování[2]. Při volbě h mimo jiné musíme předpokládat, že data obsahují méně než $\lfloor \frac{n-p+2}{2} \rfloor$ odlehlých hodnot, což například pro data s $n = 2p$ znamená přibližně 25% kontaminaci dat, ale pro $n > 5p$ zvládne algoritmus i 40% kontaminaci odlehlými hodnotami, v případě pouze dvou proměnných dosáhne algoritmus bodu selhání přesně 50%.

Limitně půjde pravděpodobnost nekontaminovaného náhodného výběru do nuly. Rousseeuw a van Driessenová adresují problém výběru počáteční množiny H_1 následujícím trikem. Budeme brát množiny *nejmenší možné velikosti*, tak aby z nich spočítaný determinant varianční matice byl nenulový. To nám téměř jistě zaručí, že mezi pětisty množinami H_1 bude alespoň některá bez odlehlých hodnot a bude vhodná pro začátek algoritmu.

Poznámka. Důvod, proč pravděpodobnost výběru nekontaminované množiny se blíží nule je tento. Označme z jako počet odlehlých hodnot, n je počet všech bodů, $h = \lfloor \frac{n+p+1}{2} \rfloor > \frac{n}{2}$, potom počet možností, kterými lze vybrat nekontaminovanou množinu je: $nek := \frac{n-z}{n} \frac{n-z-1}{n-1} \dots \frac{n-z-h+1}{n-h+1}$ a počet všech možností je $vse := \binom{n}{h}$. Pravděpodobnost, že vybereme nekontaminovanou množinu je tedy $\frac{nek}{vse} = \frac{n-z}{n} \dots \frac{n-z-h+1}{n-h+1} \cdot \frac{h}{n} \frac{h-1}{n-1} \dots \frac{2}{n-h+2} \frac{1}{n-h+1}$. Prvních h faktorů můžeme shora odhadnout jako $(\frac{n-z}{n})^h$ ostatních h faktorů odhadneme shora jako $(\frac{h}{n})^h$, čímž dostaneme vyjádření $\frac{nek}{vse} \leq (\frac{n-z}{n} \frac{h}{n})^h$. Protože $\frac{n-z}{n} \leq \frac{cn}{n} = c$, kde $c < 1$ udává poměr nekontaminovaných dat a $\frac{h}{n} > \frac{n/2}{n} = \frac{1}{2}$, dostáváme výsledně $\frac{nek}{vse} \leq (\frac{c}{2})^h < (\frac{1}{2})^h \leq (\frac{1}{2})^{n/2} \to 0$ pro zvětšující se n.

Vezměme libovolnou podmnožinu $J \subset \{1, \dots, n\}$ velikosti $p + 1$[3]. Pokud se stane, že $\det S_J = 0.$, přidáváme postupně do J libovolné další indexy až dostaneme J^* takové, že $\det S_{J^*} \neq 0$. Spočítáme střední hodnotu a varianční matici z J, resp. J^*, $m_0 = \frac{1}{h} \sum_{i \in J} x_i$ a $S_0 = \frac{1}{h} \sum_{i \in J} (x_i - m_0)(x_i - m_0)^T$ a z nich dopočítáme vzdálenosti $d_0^2(i) = (x_i - m_0)^T S_0^{-1} (x_i - m_0)$ pro každé $i \in \{1, \dots, n\}$. Vzdálenosti srovnáme podle velikosti od nejmenší po nějvětší a za H_1 vezmeme prvních h indexů s nejmenší vzdáleností.

2.3.4 Zredukování množství C-stepů

Nyní jsme v situaci, kdy máme množiny H_1, které bychom chtěli pomocí C-stepu dovést do konvergence a následně vybrat tu z množin, která má nejmenší determinant varianční matice. Narážíme tím na další problém co se týče rychlosti. Každý krok C-stepu vyžaduje výpočet střední hodnoty, varianční matice a n vzdáleností, přibližně $\mathcal{O}(np^2)$ operací jak bude později spočítáno, je tedy výpočetně náročný. K urychlení algoritmu potřebujeme zmenšit počet C-stepů. Jak je naznačeno v [3], často lze vidět rozdíl mezi robustním a nerobustním řešení v rychlosti konvergence determinantu varianční matice. U vhodných, robustních množin lze pozorovat, že již po dvou nebo třech C-stepech je jejich determinant mnohem menší, než u ostatních množin a konverguje směrem k minimu.

[2]Reálně je ve fastMCD nastavena trochu jiná hodnota, blízká $h = 0.75 \cdot n$. Přesný vzorec pro h bude uveden v části 2.3.6.

[3]Menší množina nemá smysl, determinant S by byl jistě nulový

Tato myšlenka je pouhá heuristika, jistě není univerzální. Tedy nemusí platit, že množina, která po dvou krocích konverguje rychleji než jiná množina, bude konvergovat rychleji i dál. Proto algoritmus použije 2 C-stepy na všech 500 množin H_1 a dál nepoužívá pouze jednu nejlepší, ale použije jich hned několik, obvykle se používá 10 nejlepších. Ty potom iterujeme dál až do konvergence a z nich zvolíme množinu, která má nejmenší determinant.

2.3.5 Urychlení pro velká n

Další zrychlení algoritmu se uvažuje pro velká n. Protože čas výpočtu se zvyšujícím se n rychle roste, například i kvůli potřebě vypočítat vždy n vzdáleností v každém C-stepu, pokusíme se tomu vyhnout rozdělením dat na menší podmnožiny.

Rousseuw a Van Driessenová [3] rozlišují tři případy. Pro $n < 600$ používají dosavadní algoritmus beze změny, pro $600 \leq n < 1500$ uvažují rozdělení dat na maximálně čtyři disjunktní podmnožiny, které jsou přibližně stejně velké a každá obsahuje minimálně 300 pozorování. Podmnožiny se volí náhodně.

Pro poslední případ $n \geq 1500$ se postupuje obdobně. Nejdřív náhodně vybereme množiny obsahující 1500 pozorování z n a ty pak dál rozdělíme po třech stech na dalších 5 podmnožin.

V obou dvou případech, kdy $n > 600$, pokračujeme takto: v každé podmnožině vybereme několik počátečních množin H_1, tak abychom jich celkově pro všechna data dostali 500. Velikost množiny h se musí pro rozdělené podmnožiny zmenšit v poměru $h_{roz} = h \cdot \frac{n_{roz}}{n}$, kde n_{roz} značí velikost podmmnožiny. Na každé H_1 použijeme dva C-stepy a v každé podmnožině zvlášť uchováme 10 nejlepších řešení. Příslušné podmnožiny spojíme zpět do množin o maximální velikosti 1500, příslušně změníme $h_{spoj} = h \cdot \frac{n_{spoj}}{n} \approx h \cdot \frac{1500}{n}$, ke každé máme sadu 50 (nebo méně) řešení (m, S). Tato řešení pak rozšíříme na celou sloučenou množinu, tedy znovu aplikujeme 2 C-stepy využívající již všech 1500 pozorování a uložíme 10 nejlepších z nich.

Následně rekurzivně sloučíme množiny velikosti 1500 do kompletní sady dat a stejným způsobem rozšíříme dosavadní řešení, až posledních 10 řešení dovedeme do konvergence (nebo do přednastavené hodnoty počtu C-stepů) a získáme tak nejlepší řešení celé úlohy.[4]

2.3.6 Pseudokód fastMCD

V této sekci shrneme, jak algoritmus funguje.
Vstup: $X \in \mathbb{R}^{n \times p}$

1. Zvolíme $0.5 \leq \alpha \leq 1$, potom $h = \lfloor 2(1-\alpha)\lfloor \frac{n+p+1}{2} \rfloor + (2\alpha - 1)n \rfloor$ nebo necháme přednastavené $\alpha = 0.75$ a $h = \lfloor \frac{1}{2}\lfloor \frac{n+p+1}{2} \rfloor + \frac{n}{2} \rfloor$.

2. Pokud $\alpha = 1$, použijeme všechna data k odhadům střední hodnoty m a varianční matice S a algoritmus ukončíme.

[4]Hodnoty jako 500 počátečních množin H_1, podmnožiny velikosti cca. 300, uchovávání 10 nejlepších řešení, aj. jsou přednastavené hodnoty ve fastMCD, které však jdou změnit.

3. Pokud p=1 vypočítají se odhady podle [3] pomocí algoritmu uvedeného v [12] a algoritmus ukončíme.

4. Dále už máme $h < n$, $p \geq 2$. Pro $n < 600$ použijeme všechna data.

   - Množství počátečních množin označíme $ntrial = 500$ a 500krát opakujeme:
     - vezměme libovolnou množinu H_0 velikosti $p + 1$ a podle popisu v 2.3.3 sestrojíme množinu h indexů H_1,
     - na H_1 použijeme 2 C-stepy popsané v části 2.3.1, dostaneme 500 množin H_3 s příslušnými odhady varianční matice S_3.
   - Vezmeme 10 dosavadně nejlepších řešení s nejmenším $\det S_3$, a použijeme na ně C-stepy až do konvergence, maximálně však použijeme 100 C-stepů na každou z deseti množin H_3.
   - Určíme řešení (m, S) s nejmenším $\det S$, vrátíme ho, ukončíme algoritmus.

5. Pokud $n \geq 600$ potom budeme data rozdělovat jak bylo řečeno v sekci 2.3.5.

   - Pro $n > 1500$ rozdělíme nejprve data náhodně na disjunktní množiny velikosti 1500 a ty pak rozdělíme dál po třech stech na 5 podmnožin.

     Pokud $n \leq 1500$ rozdělíme data na maximálně 4 disjunktní podmnožiny přibližně stejné velikosti $n_{roz} \geq 300$.
   - Rozdělíme 500 startů co nejrovnoměrněji mezi všechny podmnožiny a v každé podmnožině tak opakujeme zhruba $\frac{500}{\text{počet podmnožin}}$krát:
     - pomocí $p + 1$ velké množiny H_0 zkonstruujeme množinu H_1 velikosti $h_{roz} = h \cdot \frac{n_{roz}}{n}$,
     - v podmnožině použijeme na H_1 2 C-stepy,
     - uložíme 10 nejlepších odhadů (m_{roz}, S_{roz}).
     - Pokud $n \leq 1500$, pak:
       - sloučíme zpět podmnožiny do kompletních dat,
       - na všech 40 uložených řešení (m_{roz}, S_{roz}) použije C-step až do konvergence, nejvíc však použijeme 100 C-stepů na každé řešení,
       - najdeme nejlepší řešení (m, S) s nejmenším determinantem, vrátíme ho a ukončíme algoritmus.
     - Jinak $n > 1500$ a potom:
       - sloučíme vždy 5 podmnožin velikosti 300 do příslušné množiny velikosti $n_{spoj} = 1500$,
       - v každé spojené množině použijeme na 50 dosavadních řešení 2 C-stepy pro $h_{spoj} = h \cdot \frac{n_{spoj}}{n}$ a uložíme 10 nejlepších řešení,
       - nakonec sloučíme všechny množiny do kompletní sady dat a pro dosavadní nejlepší řešení použijeme několik C-stepů (již ne 100, ale počet C-stepů se odvíjí od velikosti n a p),
       - najdeme řešení (m, S) s nejmenším determinantem, vrátíme ho a skončíme.

V závislosti na velikosti n má tím pádem algoritmus různý počet fází, které používají C-stepy. Pro $n < 600$ a $600 \leq n < 1500$ máme 2 fáze. První používá C-step vždy dvakrát a finální používá až 100 C-stepů na každou množinu. Pro $n \geq 1500$ máme 3 fáze, první a druhá používají 2 C-stepy, finální fáze má počet C-stepů různý podle n a p.

Posledním vylepšením algoritmu je pak použití váhové funkce.

2.3.7 Váhy ve fastMCD

Algoritmus MCD je nejrobustnější pro $h = \lfloor \frac{n+p+1}{2} \rfloor$, kde je dosaženo nejvyššího možného bodu selhání. V takovém případě k výpočtu odhadů střední hodnoty a varianční matice použijeme h dat, místo této hodnoty h můžeme ekvivaletně brát $\alpha = 0.5$. Zároveň však má algoritmus pro toto h nízkou efektivitu pro normální rozdělení, více o tomto lze najít v [2] a [10], viz též 2.2.1.

Pokud zvýšíme α a budeme odhady počítat s větším množstvím dat, efektivita se bude zvyšovat, ale na úkor robustnosti. Z důvodu zvýšení efektivity a současného zachování robustnosti, se používá tzv. vážený odhad. Ten využívá nějakou váhovou funkci, která by ideálně měla přiřadit nízkou či nulovou váhu odlehlým hodnotám a naopak by měla upřednostnit data nejblíže k centru dat, střední hodnotě.

FastMCD využívá velmi jednoduché váhové funkce, která přiřazuje nulu odlehlým hodnotám a jedničku všem ostatním. Je to funkce robustních vzdáleností $d_k(i) = \sqrt{(x_i - m_k)^T S_k^{-1} (x_i - m_k)}$, která podobně jako toleranční elipsa využívá k rozlišení odlehlých hodnot jako mezní hodnotu $\chi^2_{p,0.975}$. Tvar váhové funkce je pak $W(d_k^2) = \psi(d_k^2(i) \leq \chi^2_{p,0.975})$ $i \in \{1, \ldots, n\}$, kde ψ je charakteristická funkce, která přiřazuje i-tému pozorování jedničku pokud nerovnost $d_k^2(i) \leq \chi^2_{p,0.975}$ platí, jinak mu přiřadí nulu.

Získáváme tím vážené následující odhady pro střední hodnotu a varianční matici:

$$m_k = \frac{\sum_{i=1}^{n} W(d^2(i)) x_i}{\sum_{i=1}^{n} W(d^2(i))},$$
$$S_k = c \cdot \frac{1}{n} \sum_{i=1}^{n} W(d^2(i))(x_i - m_k)(x_i - m_k)^T.$$

Konstanta c je tzv. konzistenční faktor, který zajišťuje pro normální rozdělení konzistenci, což je vlastnost, že pro zvětšující se n se odhad parametru limitně blíží k samotnému parametru.

2.3.8 Exaktní fit

Zajímavou vlastností MCD estimátoru je možnost použití na exaktní fit, to znamená, když h nebo více pozorování leží v nějaké nadrovině menší dimenze než p. I v takovém případě dostaneme odhad střední hodnoty m a varianční matice S, která v takovém případě bude singulární. Algoritmus pak z (m, S) vypočítává i rovnici nadroviny.

Pokud máme data s $n > 600$, rozdělujeme množinu pozorování na menší podmnožiny a pro $h_{roz} = h \cdot \frac{n_{roz}}{n}$ hledáme odhady (m_{roz}, S_{roz}). Může se stát, že

narazíme na případ, kdy $\det S_{roz} = 0$. Z toho víme, že existuje h_{roz} nebo více pozorování ležících ve stejné nadrovině. Zkusíme projít celou sadu dat a když najdeme h nebo více pozorování v této nadrovině, dopočítáme k ní střední hodnotu a varianční matici (m, S), které budou konečným výsledkem a zastavíme algoritmus. Jinak pokračujeme, s tím že (m_{roz}, S_{roz}) pro které $\det S_{roz} = 0$ budeme jistě předávat jako jeden z 10 nejlepších výsledků do sloučených množin a případně postup opakujeme.

3. Numerické vlastnosti algoritmu fastMCD

Kvalitu algoritmu fastMCD zde budeme posuzovat z hlediska tří kritérií: výpočetních nákladů, paměťových nákladů a stability. Zaměříme se především na výpočetní náklady a stabilitu, které nejsou tak přímočaré jako paměťové náklady. Výpočetní náklady nám dávají odhad, jak rychle poroste množství operací, a tím i čas potřebný k výpočtu, v závislosti na velikosti úlohy. Stabilita zase určuje, jak moc se na výsledky můžeme spolehnout, vzhledem k vlivu konečné aritmetiky. Tedy, jestli lze říct, že spočítaný výsledek je blízko přesnému řešení.

V původním článku o fastMCD z roku 1999 [3] se ze jmenovaných numerických vlastností algoritmu řeší pouze výpočetní náklady v závislosti na n. Předpokládá se p velmi malé a proto se zde tvrdí, že C-step má náročnost $\mathcal{O}(n)$. Protože se ale fastMCD používá i na data s $p > 50$, má i p vliv na odhad výkonnosti algoritmu. Zkusíme proto upřesnit a doplnit numerické vlastnosti i vzhledem k velikosti p.

3.1 Paměťové náklady

Začněme u paměťových nákladů. Podívejme se na nejnáročnější případ, kdy $n > 1500$ a data dělíme. Kromě samotné matice X, která má np hodnot, uchováváme také 10 nejlepších řešení pro každou podmnožinu velikosti 300, tj. ukládáme odhady středních hodnot, variančních matic a jejich hodnoty determinantů a příslušné indexy pro každou množinu. Pro každou podmnožinu tedy potřebujeme $p + p^2 + 1 + h$ hodnot, podmnožin je přibližně $\frac{n}{300} = \gamma$, tedy konstantní (malé) množství. Celkově budeme v nejhorším případě potřebovat uložit $np+\gamma\cdot(h+p^2+p+1) = \mathcal{O}(np)$ čísel. Standardně jsou uložena jako číslo s plovoucí řádovou čárkou typu double, kdy každé číslo zabere 8 bytů.

3.2 Výpočetní náklady

FastMCD je výpočetně náročná metoda. C-step jako nejdůležitější součást algoritmu je také nejdražší. Vypočítává vždy odhad střední hodnoty, varianční matice, její determinant a všech n Mahalanobisových vzdáleností jednotlivých pozorování. Soustředíme se na variantu algoritmu, který je v knihovně LIBRA [5] pro MATLAB. Tato verze využívá v průběhu ekonomický singulární rozklad.

Definice 24 (Ekonomický singulární rozklad)**.** *Pro každou matici $A \in \mathbb{R}^{h\times p}$ hodnosti r existují ortogonální matice $U \in \mathbb{R}^{h\times r}$ a $V \in \mathbb{R}^{p\times r}$ a kladná čísla $\rho_1 \geq \cdots \geq \rho_r > 0$ tak, že $A = U\mathrm{diag}(\rho_1, \ldots, \rho_r)V^T = URV^T$.*

Singulární čísla ρ_i jsou odmocninami z vlastních čísel matice A, což budeme dále využívat. C-step rozdělíme do jednotlivých kroků podle Věty 3 a budeme počítat počty elementárních operací - sčítání/odčítání, násobení, dělení aj. v závislosti na velikosti úlohy, tj. na množství dat n, na počtu proměnných p a v závislosti na velikosti podmnožiny indexů h. Omezíme se na tzv. floating point operations (flops).

- $m_k = \frac{1}{h} \sum_{i \in H_k} x_i$

  v sumě sčítáme h vektorů velikosti p, jednou dělíme $1/h$ a násobíme číslo krát vektor velikosti p

  počet operací: $(h-1) \cdot p + 1 + p = \mathcal{O}(hp)$

- S_k se explicitně nevypočítává, pouze určíme $X_k - M_k$ z výrazu

$$S_k = \frac{1}{h} \sum_{i \in H_k} (x_i - m_k)(x_i - m_k)^T = \frac{1}{h}(X_k - M_k)^T (X_k - M_k),$$

  kde matice X_k i M_k jsou velikosti $h \times p$, X_k má v řádcích řádky matice X s indexy z H_k, matice M_k má v každém řádku vektor výběrové střední hodnoty m_k

  počet operací: $2hp = \mathcal{O}(hp)$

- získáme spektrální rozklad S_k ve tvaru $V_k R_k^2 V_k^T$, kde $V_k \in \mathbb{R}^{p \times p}$ je unitární matice a $R_k^2 \in \mathbb{R}^{p \times p}$ je diagonální matice vlastních čísel

  použije se ekonomický SVD rozklad $X_k - M_k = U_k R_k V_k^T$, který stojí zhruba $6hp^2 + 20p^3$, potom

$$S_k = \frac{1}{h} V_k R_k^T U_k^T U_k R_k V_k^T = \frac{1}{h} V_k R_k^2 V_k^T$$

  počet operací: $6hp^2 + 20p^3 = \mathcal{O}(hp^2)$

- $\det S_k = \prod_{i=1}^{p} r_{ii}$, kde r_{ii} jsou diagonální prvky matice R_k^2

  počet operací: $p - 1 = \mathcal{O}(p)$

- výpočet vzdáleností $d_k^2(i) = (x_i - m_k)^T S_k^{-1} (x_i - m_k)$ pro $i = 1, \ldots, n$

  nejprve se zinvertuje matice R_k^2 na $(R_k^2)^{-1} = diag(\frac{1}{r_{11}}, \ldots, \frac{1}{r_{pp}})$, potom spočítáme $[(X_k - M_k)V_k]$. Vektor d_k^2 lze počítat celý naráz. Můžeme vypočítat

$$\begin{aligned} D_k :&= (X_k - M_k) S_k^{-1} .* (X_k - M_k) = \\ &= (X_k - M_k)(V_k R_k^2 V_k^T)^{-1} .* (X_k - M_k) = \\ &= [(X_k - M_k)V_k](R_k^2)^{-1} .* [(X_k - M_k)V_k], \end{aligned}$$

  kde operací $.*$ značí násobení po prvcích a následně $d_k^2(i) = \sum_{j=1}^{n} d_{ij}$, kde d_{ij} je (i,j)-tý prvek D_k.

  počet operací: $p + np^2 + 2np = \mathcal{O}(np^2)$

- srovnání $d_k(i)$ pomocí quicksort algoritmu

  počet operací: $\mathcal{O}(n \cdot \log n)$

Protože předpokládáme, že $p < n$ a že $h < n$ je pevné číslo, dostáváme dohromady tento odhad složitosti jednoho C-stepu:

$$\mathcal{O}(2hp + hp^2 + p + np^2 + n \cdot \log n) = \mathcal{O}(np^2).$$

3.3 Stabilita

Stabilita je důležitá numerická vlastnost algoritmů. Vypovídá o přesnosti výpočtů v konečné aritmetice. Přímá stabilita přímo poměřuje rozdíl přesného řešení a výpočtem získaného řešení.

Definice 25 (Přímá stabilita)**.** *Nechť máme dána nějaká data x a algoritmus f. Označme jako $f(x)$ přesný výsledek algoritmu a jako $\mathrm{fl}(f(x))$ výsledek v konečné aritmetice. Potom přímá analýza měří relativní chybu výsledků $\frac{\|f(x)-\mathrm{fl}(f(x))\|}{\|f(x)\|}$ a algoritmus je pak přímo stabilní, když je tato chyba velmi malá, nejlépe když je úměrná strojové přesnosti u.*

Jak moc malá chyba je ještě přípustná pro přímou stabilitu záleží na konkrétních datech a algoritmu. Analyzovat však takto chybu je obvykle nemožné, protože jednoduše neznáme přesný výsledek. Proto se častěji používá zpětná stabilita.

Definice 26 (Zpětná stabilita)**.** *O algoritmu řekneme, že je zpětně stabilní, pokud spočtené řešení dané úlohy je přesným řešením pozměněné úlohy, jejíž vstupní data jsou blízká původním datům dané úlohy. Tedy pokud máme $\mathrm{fl}(f(x)) = f(\tilde{x})$ pro nějaká data $\tilde{x}$, pak je algoritmus f zpětně stabilní, když $\frac{\|x-\tilde{x}\|}{\|x\|} \approx \mathcal{O}(u)$.*

Prozkoumáme teď stabilitu pro fastMCD, přičemž se omezíme na regulární výběrové varianční matice S_k.

3.3.1 Stabilita fastMCD

Jak bylo výše řečeno, fastMCD z knihovny LIBRA využívá v průběhu C-stepu ekonomického singulárního rozkladu. SVD rozklad je zpětně stabilní [13], norma chyby půjde odhadnout jako $\|E\| \approx u \cdot \|X_k - M_k\|$. Obecně se SVD pokládá za nejstabilnější rozklad. Za stabilitu ovšem platíme velkou výpočetní náročností.

Rozklad matice $X_k - M_k$ počítáme pomocí SVD přes bidiagonalizaci, tj. převedeme $X_k - M_k$ pomocí unitárních matic P_k a Q_k na bidiagonální matici B_k, kterou teprve rozložíme SVD rozkladem:

$$P_k^T(X_k - M_k)Q_k = B_k = \hat{U}_k R_k \hat{V}_k^T.$$

Matice $X_k - M_k$ je potom rovna:

$$X_k - M_k = P_k(\hat{U}_k R_k \hat{V}_k^T)Q_k^T = (P_k\hat{U}_k)R_k(Q_k\hat{V}_k)^T = U_k R_k V_k^T.$$

Z bidiagonální matice už dostaneme singulární čísla s relativní chybou úměrnou strojové přesnosti u [13, str. 129]. Problém může nastat při násobení matic. Když například násobíme $\mathrm{fl}(P_k^T(X_k - M_k)) = P_k^T((X_k - M_k) + E_k)$, kde pro chybu E_k platí, že $\|E_k\| \approx u\|X_k - M_k\|$, viz [13, str. 70] a [14, str. 100]. Norma chyby závisí tedy hlavně na normě matice $X_k - M_k$, tedy na nejvetším ze singulárních čísel.

Přesnost, či chybu, jednotlivých singulárních čísel počítáme relativně k normě matice kterou rozkládáme, tj. relativně k největšímu ze singulárních čísel, proto mohou být nejmenších singulární čísla mnohem nepřesnější. Záleží vždy na tom jak jsou od sebe řádově daleko singulární čísla, viz Corollary 8.6.2. v [14]. Naproti tomu pravé i levé singulární vektory, což jsou sloupcové vektory v_i a u_i matic V_k a

U_k z rozkladu $X_k - M_k$, přímo stabilní obecně nejsou. Záleží na blízkosti přesných a vypočtených singulárních čísel, viz Theorem 8.6.5 z [14].

Stejně tak budeme násobit matice při výpočtu vzdálenosti. Znovu bude platit, že je maticové násobení zpětně stabilní (více znovu [14, str. 100], ale norma chyby je závislá na normě A_k.

3.3.2 Použití spektrálního rozkladu

Protože je singulární rozklad hodně drahý, lze ho alternativně nahradit roznásobením matic $S_k = \frac{1}{h}(X_k - M_k)^T(X_k - M_k)$ a použít spektrální rozklad na výslednou matici S_k.

Definice 27 (Spektrální rozklad)**.** *Uvažujme symetrickou matici $A \in \mathbb{R}^{p\times p}$, pak existuje její spektrální rozklad $A = U\Lambda U^T$, kde U je ortogonální matice, která ve sloupcích obsahuje vlastní vektory matice A a $\Lambda = diag(\lambda_1, \ldots, \lambda_p)$ je diagonální matice obsahující vlastní čísla matice A.*

Výpočetní náročnost se řádově nezmění, prakticky se ale skutečně sníží díky zmenšení konstant a členů nižšího řádu. Singulární rozklad potřebuje zhruba $6hp^2 + 20p^3$ flopů, zatímco se spektrálním rozkladem se dostaneme přibližně na $hp + hp^2 + 4/3 \cdot p^3$, protože roznásobení matic $(X_k - M_k)^T(X_k - M_k)$ nás bude stát přibližně $\mathcal{O}(hp^2)$ a následný rozklad, který je pro symetrickou matici poměrně efektivní, stojí $4/3 \cdot p^3 = \mathcal{O}(p^3)$.

Použitím spektrálního rozkladu $S_k = U_k\Lambda_k U_k^T$ můžeme vypočítat determinant a vzdálenosti $d_k(i)$ podobně snadno jako předchozím případě. Pro determinant platí: $\det S_k = \det \Lambda_k = \prod_{i=1}^{p} \lambda_i$. Ke stabilitě tohoto výpočtu se dostaneme později. K výpočtu Mahalanobisových vzdáleností využijeme toho, že jde matici Λ_k rozložit na $\Lambda_k = \Lambda_k^{1/2}\Lambda_k^{1/2}$. To plyne z toho, že pokud je S_k pozitivně definitní, má všechna vlastní čísla kladná a lze je odmocnit. Potom vzdálenosti vypočítáme přes vzorec

$$d_k^2(i) = [(x_i - m_k)^T U_k \Lambda_k^{-1/2}][\Lambda_k^{-1/2} U_k^T (x_i - m_k)],$$

kde stačí spočítat pouze jednu z hranatých závorek, druhou pak získáme transpozicí. Pak už stačí přímočaře roznásobit. Případně jde znovu rovnou využít maticového zápisu $[(X_k - M_k)U_k\Lambda_k^{-1/2}]. * [(X_k - M_k)U_k\Lambda_k^{-1/2}]$ a potom $d_k^2(i)$ je součtem i-tého řádku této matice.

Náročnost výpočtu determinantu ani n vzdáleností se proto skutečně nezmění, zůstanou $\mathcal{O}(p)$, resp. $\mathcal{O}(np^2)$. Z hlediska výpočtové náročnosti jsou tedy obě možnosti rozkladu a výpočtu řádově stejně nákladné.

Rozdíl nastane ve stabilitě. Zatímco v případě použití singulárního rozkladu jsme používali matici $(X_k - M_k)$ s podmíněností $\kappa(X_k - M_k)$, zde rozkládáme matici S_k s podmíněností $(\kappa(X_k - M_k))^2$. Z Corollary 8.1.6. v [14] vyplývá pro citlivost vlastních čísel λ_i, že chyba ve výpočtu každého vlastního čísla je shora odhadnuta $\|E\|_2 \approx u\|S_k\|_2 = u\|(X_k - M_k)^T(X_k - M_k)\|_2 = u\|X_k - M_k\|^2$. Pokud má tedy matice $X_k - M_k$ velké číslo podmíněnosti, bude spektrální rozklad mnohem méně stabilní než SVD rozklad.

Problém může nastat také kvůli ztrátě informace při roznásobování matic $(X_k - M_k)^T(X_k - M_k)$. Může například dojít ke snížení hodnosti, jak je zmíněno v [14] a [15]. Následující jednoduchý příklad z [15] dobře ilustruje problém.

Příklad. Nechť u značí strojová přesnost počítače a $\epsilon \in (0, \sqrt{u})$. Definujme matici $A = \begin{vmatrix} 1 & 1 \\ \epsilon & 0 \end{vmatrix}$. A má jistě plnou hodnost a nenulový determinant rovný $-\epsilon$. Pokud spočítáme $A^T A = \begin{vmatrix} 1+\epsilon^2 & 1 \\ 1 & 1 \end{vmatrix} = \begin{vmatrix} 1 & 1 \\ 1 & 1 \end{vmatrix}$, protože počítač nedokáže čísla 1 a $1+\epsilon^2$ odlišit a výsledná matice je singulární.

Zde navíc zmenšujeme dimenze. Z matice $h \times p$ dostaneme pouze $p \times p$ matici, přičemž p je obvykle několikanásobně menší než h.

Výpočet determinantu a n Mahalanobisových vzdáleností je prakticky stejný jako u singulárního rozkladu. Stabilita těchto výpočtů se proto nezmění.

3.3.3 Použití Choleského rozkladu

Druhou možností je rozložit matici S_k pomocí Choleského rozkladu.

Definice 28 (Choleského rozklad)**.** *Mějmě matici $A \in \mathbb{R}^{p\times p}$, která je symetrická a pozitivně definitní, tj. pro každé $x \neq 0 \in \mathbb{R}^p$ platí $x^T Ax > 0$, pak existuje jednoznačný rozklad $A = LL^T$, kde $L \in \mathbb{R}^{p\times p}$ je dolní trojúhelníková matice s kladnými prvky na diagonále.*

Výpočet nás bude opět stát $\mathcal{O}(hp^2)$ za získání matice S_k, Choleského rozklad potom stojí pouze $1/3 \cdot p^3 = O(p^3)$ operací, stojí proto 4krát méně než spektrální rozklad.

Determinant lze snadno dostat jako $\det S_k = \det(L_k {L_k}^T) = \det L_k \cdot \det {L_k}^T = (\prod_{i=1}^{p} l_{ii})^2$, kde l_{ii} jsou diagonální prvky matice L. Choleského rozklad také používáme pro výpočet vzdáleností $d_k(i)$:

$$\begin{aligned} d_k^2(i) &= [(x_i - m_k)^T ({L_k}^T)^{-1}][(L_k)^{-1}(x_i - m_k)] = \\ &= [({L_k}^{-1})(x_i - m_k)]^T [({L_k}^{-1})(x_i - m_k)], \end{aligned}$$

nebo můžeme stejně jako v předchozích případech použít maticovou verzi:

$$d_k^2 = [(X_k - M_k)(L_k^T)^{-1}]. * [(X_k - M_k)(L_k^T)^{-1}]$$

Pak stačí vypočítat pouze $Y = (X_k - M_k)(L_k^T)^{-1} \in \mathbb{R}^{n\times p}$. Díky tomu, že je L_k^T v horním trojúhelníkovém tvaru, půjde snadno najít matici Y jako soustavu lineárních rovnic pomocí přímé substituce, která vyžaduje $\mathcal{O}(p^2)$ operací.

$$Y = (X_k - M_k)({L_k}^T)^{-1} \iff Y L_k^T = (X_k - M_k)$$

Dále stejným způsobem najdeme $d_k^2(i)$ jako součet i-tého řádku matice $Y. * Y$, kde znovu $.*$ značí násobení po prvcích.

Choleského rozklad má také dobré numerické vlastnosti. Je totiž bezpodmínečně zpětně stabilní, což lze najít včetně důkazu v [13]. Bezpodmínečná zpětná stabilita nám říká, že nezávisle na prvcích matice L se použitím Choleského rozkladu dopustíme jen malé chyby.

Přímá analýza chyb pro výpočet soustavy $Ax = b$ s použitím Choleského rozkladu matice A se ukazuje v [14], část 4.2.6 a 10. kapitola z [15]. Říká, že chyba spočteného řešení je odhadnuta jako $\|E\|_2 \leq \beta_n u \|A\|$, kde β_n je nějaká malá

konstanta závislá na n. Navíc během rozkladu nenastane problém s odmocňováním záporného čísla, pokud platí odhad $\gamma_n u \kappa_2(A) \leq 1$ pro nějakou jinou malou konstantu γ_n.

Pokud máme špatně podmíněnou matici A, může nastat při Choleského rozkladu chyba. Protože pro symetrickou pozitivně definitní matici jsou singulární a vlastní čísla stejná, můžeme počítat podmíněnost matice A pomocí vlastních čísel jako $\kappa_2(A) = \frac{\lambda_{max}(A)}{\lambda_{min}(A)}$, kde λ_{max} resp. λ_{min} značí největší a nejmenší vlastní číslo. Stabilita proto závisí na tom, jak daleko od sebe λ_{max} a λ_{min} jsou. V našem případě proto stabilita zavisí na číslu podmíněnosti matice S_k, potažmo L_k.

Pro substituci při řešení soustavy $YL_k^T = (X_k - M_k)$ záleží přesnost řešení na číslu podmíněnosti matice L_k. Obecně lze říci, že dobře podmíněné matice řeší soustavy s trojúhelníkovou maticí s velkou přesností. Pro špatně podmíněné matice obecně nelze nic říci, ale může být velký rozdíl mezi řešením soustavy s maticí L_k a s L_k^T. Jedna matice může mít totiž číslo podmíněnosti nízké, zatímco druhá druhá matice může být velmi špatně podmíněná. Přímá stabilita chyby je omezená exponenciálně klesající funkcí vzhledem k pořadí prvku, který počítáme. Proto později počítané prvky Y jsou přesnější než již vypočítané prvky, více v [15], Lemma 8.6 a diskuze za ním.

Z výsledků výše jasně plyne, že lepší stabilitu získáme pouze na úkor výpočetních nákladů. Pokud od algoritmu vyžadujeme především stabilitu, je nejlepší použít SVD rozklad. Naopak chceme-li docílit nižších výpočetních nákladu, bude vhodnější použít spektrální nebo Choleského rozklad. Protože však spektrální a Choleského rozklad mají srovnatelnou stabilitu, dá se za účelem zrychlení doporučit spíše Choleského rozklad, jehož výpočet stojí zhruba čtvrtinu oproti spektrálnímu rozkladu.

3.3.4 Stabilita invertování matice

Přestože v C-stepu se běžně invertuje pouze diagonální matice vlastních čísel (pokud nepoužijeme Choleského rozklad), objevuje se v algoritmu *mcdcov* v knihovně LIBRA i invertování plné matice. V jiné verzi fastMCD algoritmu pro MATLAB, kterou lze také běžně sehnat v [16], se dokonce místo SVD rozkladu invertuje celá matice S_k v každém C-stepu.

Inverzi matice se doporučuje vždy vyhnout, pokud je to možné. Invertování je zaprvé drahé, zadruhé je operace nestabilní a může během ní dojít k poměrně velkým chybám, obzvlášť pokud je matice blízko k množině singulárních matic, tj. má velké číslo podmíněnosti.

Příklad. Vezměme matici z MATLABu, která je špatně podmíněná a zkusme vyřešit soustavu rovnic $Ax = b$ různými způsoby.

Nechť $A = \mathsf{gallery}('\text{moler}', 20, -1)$, což je symetrická pozitivně definitní matice

s číslem podmíněnosti $1.72 \cdot 10^{13}$. Tvar této matice je:

$$A = \begin{pmatrix} 1 & -1 & -1 & -1 & \cdots & -1 \\ -1 & 2 & 0 & 0 & \cdots & 0 \\ -1 & 0 & 3 & 1 & \cdots & 1 \\ -1 & 0 & 1 & 4 & \cdots & 2 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -1 & 0 & 1 & 2 & \cdots & 20 \end{pmatrix}$$

Zvolme náhodný vektor x_{presny} =rand(20,1) a určeme příslušnou pravou stranu $b = A * x_{presny}$.

Zkusíme vyřešit soustavu $Ax = b$ třemi různými způsoby. Funkce $A\backslash b$ z MATLABu využívá k výpočtu pro symetrické pozitivně definitní matice Choleského rozklad, dále se dopočítá x pomocí přímé a zpětné substituce: $Ly = b$ a $L^T x = y$. SVD rozklad $A = URV^T$ se použije k určení x jako $x = V(R^{-1}(U^T b))$. Poslední způsob je pomocí přímé inverze $x = A^{-1}b$. Určíme chybu spočteného řešení $\hat{x}$ od přesného $\|\hat{x} - x_{presny}\|$ a relativní residuum $\|b - A\hat{x}\|/(\|A\|\|b\|)$.

metoda výpočtu	$\|\hat{x} - x_{presny}\|$	$\|b - A\hat{x}\|/(\|A\|\|b\|)$
$A\backslash b$	$9.8368 \cdot 10^{-5}$	$8.7349 \cdot 10^{-18}$
SVD	$4.21 \cdot 10^{-2}$	$2.0393 \cdot 10^{-17}$
$A^{-1}b$	$1.1 \cdot 10^{-3}$	$6.6409 \cdot 10^{-9}$

Už z chyby ve výsledku se zdá být ze všech tří metod Choleského rozklad nejlepší, ovšem opravdový rozdíl je vidět v reziduu. Reziduum nám v podstatě dává zpětnou stabilitu, jasně říká, že výpočet s inverzí má mnohem horší zpětnou stabilitu oproti oběma ostatním metodám. Jak lze najít v [15] nejlepší možný odhad zpětné chyby pomocí rezidua je pro invertování matice:

$$|b - A\hat{x}| \leq \frac{nu}{1 - nu}|A||A^{-1}||b|,$$

kde u značí strojovou přesnost, n je velikost úlohy a $|\cdot|$ je chyba braná po jednotlivých prvcích. Pro příklad výše nám dává horní odhad $|b - A\hat{x}|$ pro všechny prvky chybu omezenou číslem 0.17.

3.3.5 Determinant

Při používání determinantu musíme dbát na dvě věci, jak můžeme najít v [15]. Velikost determinantu nijak nevypovídá o podmíněnosti matice. Pokud bychom vzali ortogonální matici A a přenásobili jí nějakým číslem γ, dostaneme matici γA, jejíž determinant bude roven $\pm\gamma^n$ a může proto být libovolně velký pro různé hodnoty γ, přestože matice γA má číslo podmíněnosti vždy γ.

Skutečný problém však souvisí s reprezentací dat a omezením počítače. Může se totiž poměrně snadno stát, že při výpočtu determinantu přeteče nebo podteče možnosti reprezentace čísla v počítači, což závisí na velikosti prvků matice a případně i na velikosti matice samotné. Například při podtečení dostaneme tak nízké číslo, že je numericky považováno za nulu, přestože matice vůbec nemusí být singulární. Způsobem jak tomu zabránit je logaritmování determinantu. Pro

diagonální matici D to znamená, že vypočítáme $\log|\det(D)| = \log\prod_{i=1}^{n}|d_{ii}| = \sum_{i=1}^{n}\log|d_{ii}|$.

V *mcdcov* se determinant používá k určení hodnosti matice $X_k - M_k$ a tím i S_k a dále k porovnávání dosavadně nejlepších výsledků. LIBRA by mohla mít problém právě s přetékáním či podtékáním, data ale standardizuje (transformací pomocí mediánu), takže problémům dochází méně často. Přesto by bylo vhodné do algoritmu logaritmování začlenit, čímž by se problém odstranil úplně.

4. Numerické experimenty

Všechny výpočty využívají počítač s 2,66 GHz i7 procesorem a 8 GB RAM, program MATLAB 2012a [17] a používají fastMCD skript *mcdcov* z knihovny LIBRA [5].

Před začátkem experimentů ještě konstatujme, že se v algoritmu sice uvádí, že „vybíráme množiny velikosti $p + 1$ náhodně“, ale v realizaci *mcdcov* to tak ve skutečnosti není. Výběr je pseudonáhodný a pokud bychom algoritmus pustili několikrát se stejnými parametry na stejná data, dostaneme stejný výsledek. Náhodnost by však šlo v MATLABu jednoduše přidat příkazem randperm.

Uveďme také, že nalezení malého determinantu nutně nezaručuje dobrou detekci odlehlých hodnot. Pořád je fastMCD algoritmus, který pouze aproximuje MCD estimátor, a ten zase představuje heuristickou myšlenku. To ukáže následující příklad.

Příklad 4.1. První z experimentů je z dat o ionosféře [18]. Tato data mají 351 pozorování a 34 proměnných, první dvě pozorování však jsou z výpočtu vynechána, vedou totiž k exaktnímu fitu. Navíc je u těchto dat známé, která data jsou *správná* a která jsou *špatná* a měla by vyjít odlehlá. Správná data jsou v tomto příkladě taková data z radaru, která nějakým způsobem ukazují, co se v ionosféře nachází. Naproti tomu špatná, odlehlá data, jsou taková, kdy vlny z radaru projdou skrze ionosféru a neodrazí se.

Nesprávných dat je 126 z celkových 351 pozorování. Budeme zde zkoumat jak dobře algoritmus odhaluje odlehlé hodnoty, když budeme měnit parametr α, měníme tak přímo poměr odlehlých hodnot, které by měl algoritmus zvládnout.

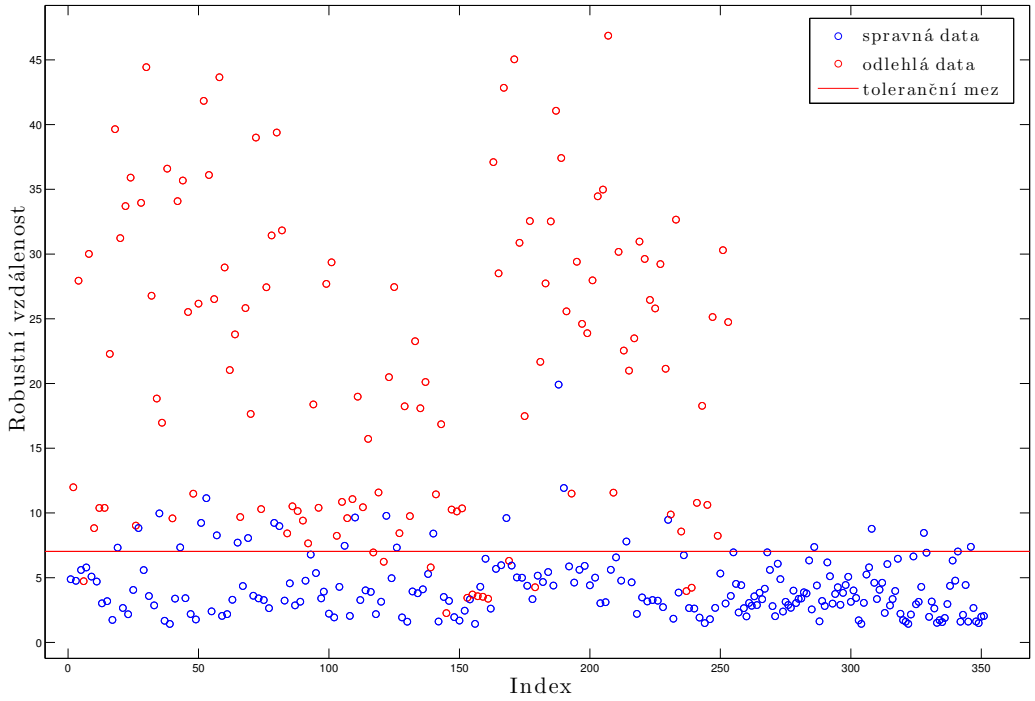
V následující Tabulce 4.3 jsou shrnuta měření, kde jsme pomocí fastMCD zjišťovali počet odlehlých hodnot, počet nesprávně klasifikovaných hodnot, dále hodnotu determinantu varianční matice a počet použitých C-stepů (v závorce příslušné pořadí množiny H_1), než algoritmus našel v datech nejmenší determinant.

α	h	#odlehlých	#nesprávných	determinant	#C-stepů k det
0.5	192	206	92	$5.8695 \cdot 10^{-60}$	36 (12)
0.625	231	172	64	$1.1182 \cdot 10^{-59}$	303 (101)
0.75	271	137	39	$3.8985 \cdot 10^{-48}$	1287 (429)
0.875	311	102	36	$2.2858 \cdot 10^{-37}$	1194 (378)
1	351	140	42	$6.545 \cdot 10^{-48}$	X

Tabulka 4.1: Ionosférická data pro různá α

Při pohledu na Tabulku 4.1 vidíme, že byly dosažené velmi malé hodnoty determinantů, o mnoho menší než je strojová přesnost. Vznikly v důsledku násobení $p = 32$ vlastních čísel.

Na první pohled asi překvapí poměrně vysoký počet špatně určených pozorování, tj. takových pozorování, která jsou určena jako odlehlé hodnoty, přestože jde o správná měření a naopak. Objevuje se zde problém, že i měření vln, které se od ionosféry neodrazí, mohou být velmi podobná správným datům, více v [19]. Na



Obrázek 4.1: Určené a skutečné odlehlé hodnoty v ionosférických datech, $\alpha = 0.75$

Obrázku 4.1 jsou vykresleny správná i odlehlá data a toleranční mez $\chi^2_{32,0.975}$. Je vidět, že na to, jak jsou si správná i některá odlehlá data blízká, určení odlehlých hodnot je poměrně *rozumné*.

Z podobného důvodu se i stalo, že nejlepší výsledek nám dal případ, kdy $\alpha = 0.875$, přestože podle počtu odlehlých hodnot, který známe, by měl být nejpřesnější výsledek pro $\alpha = 0.625$.

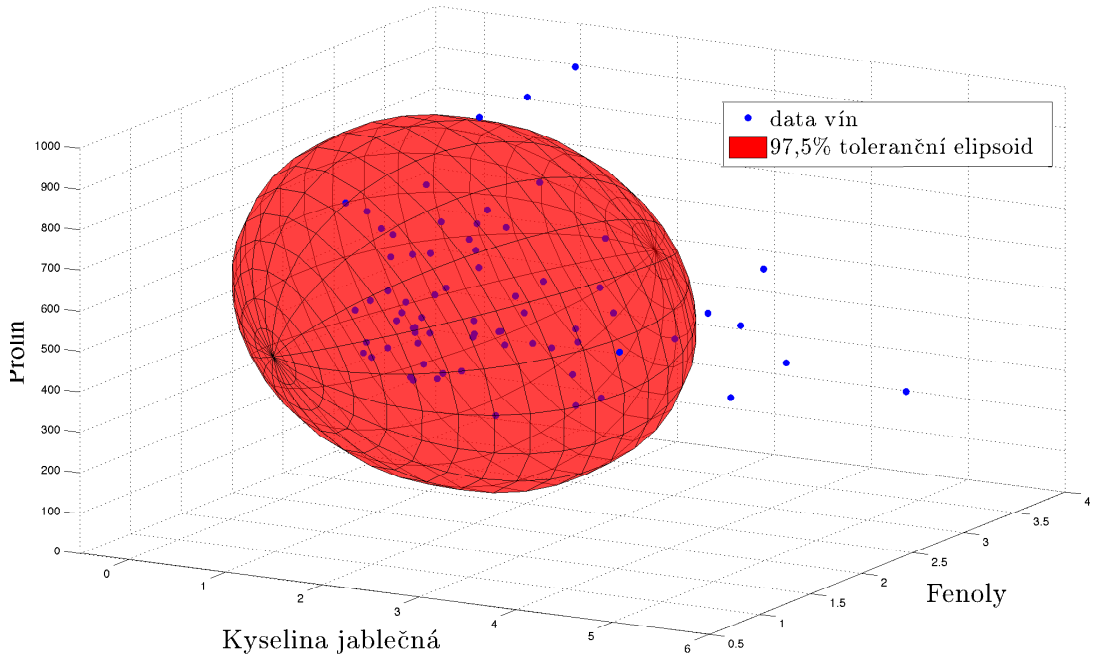
Tento experiment nikdy nepotřeboval všech 500 počátečních množin H_1, nejmenší determinant se vždy nalezl dříve. Dokonce nikdy nebylo třeba finální fáze C-stepů (viz. 2.3.6), kdy se bere 10 množin s dosavadně nejnižším determinantem, které se pak až 100krát iterují, protože ke zlepšení už v žádném z případů nedošlo. Na druhou stranu, to jestli množinu s nejmenším determinantem dostaneme už během vyzkoušení prvních 100 množin H_1, nebo až blízko pětisté množiny, je zcela náhodné. Není tudíž možné bez rozsáhlejšího experimentu s měněním hodnoty *ntrial* říct, zda je v tomto případě hodnota $ntrial = 500$ zbytečně velká.

Příklad 4.2. Druhý z experimentů se vrací k datům vín [4]. V kapitole 3 jsme upozornili na to, že počet proměnných může mít důležitý vliv na výpočetní náklady. Zde se pokusíme naznačit, že když uměle snížíme počet proměnných, se kterými budeme počítat odhady pomocí fastMCD, můžeme dostat jinou klasifikaci odlehlých hodnot.

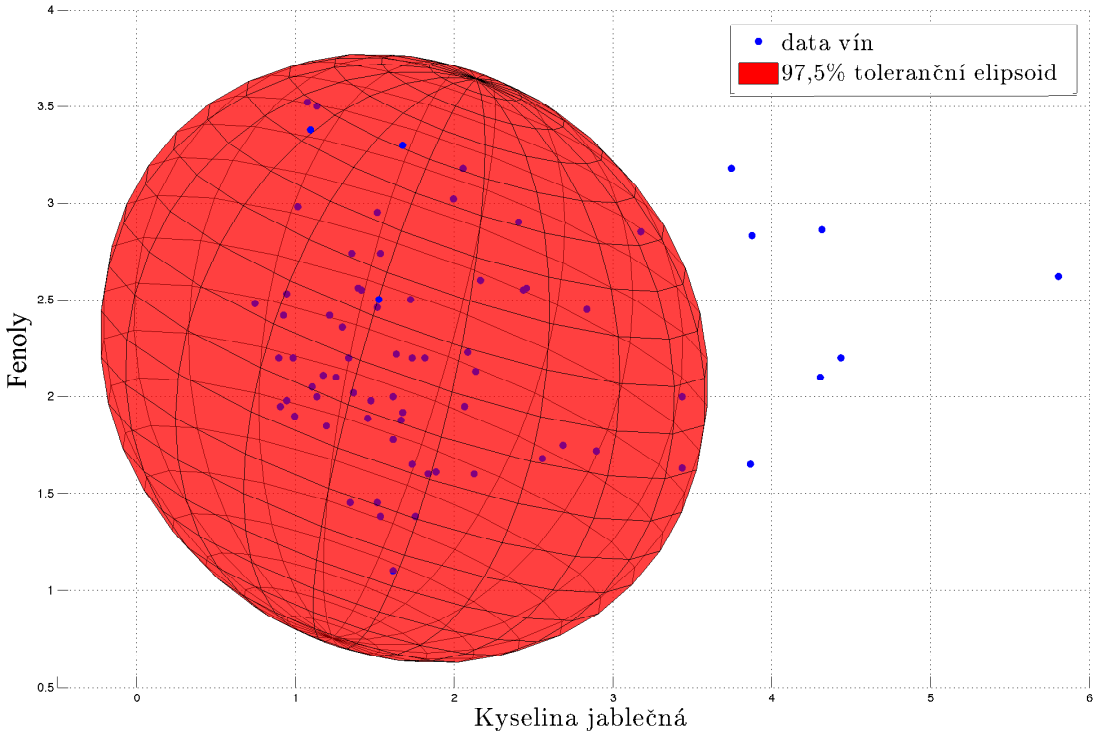
Zde se vzaly tři parametry: kyselina jablečná, fenoly a prolin, a vykreslil se 97,5% elipsoid, viz Obrázek 4.2. Pokud se podíváme na elipsoid pouze v jedné rovině (kyselina jablečná–fenoly) a vykreslíme příslušnou toleranční elipsu pro stejné dva parametry, můžeme jednoduše vizuálně srovnat změnu v určení odlehlých hodnot, viz Obrázek 4.3. Použití fastMCD na tři parametry oproti dvěma parametrům se projevilo v natočení hlavní osy elipsoidu mimo rovinu, navíc vyšší

dimenzí p se zvyšuje i mezní hodnota χ^2 a elipsoid tak zahrnuje některá pozorování, která v dvourozměrné případě jsou odlehlá.

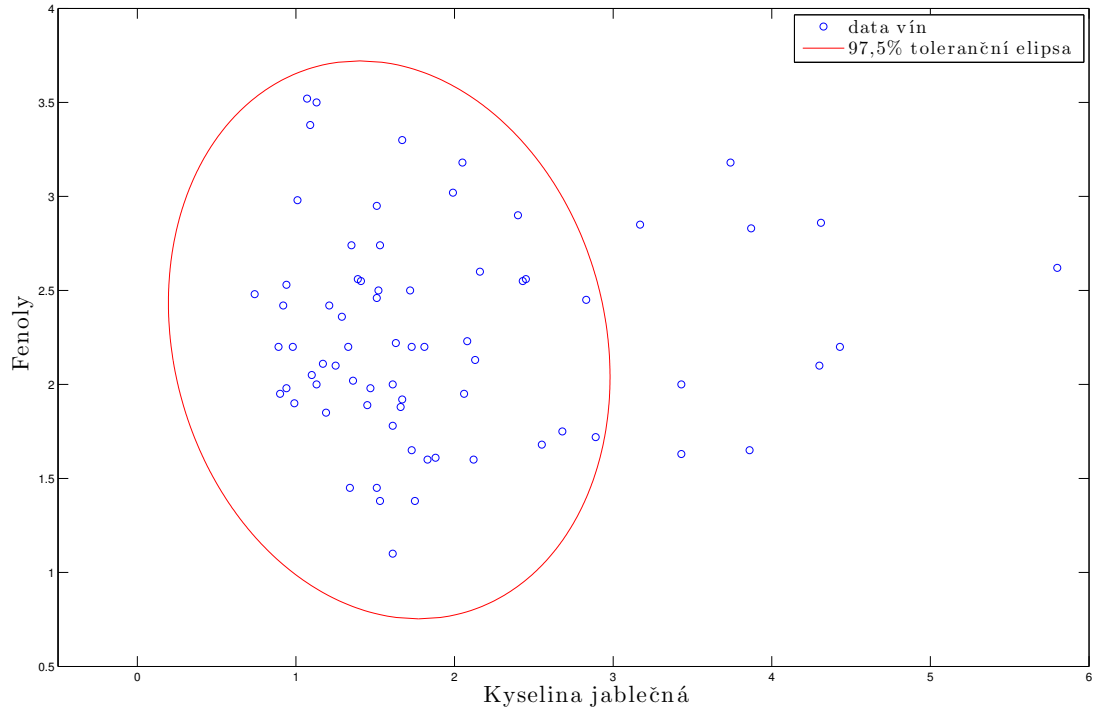
Tento příklad tedy ilustruje, že není možné data procházet po množinách s menším p a následně se snažit z takových odhadů vytvořit představu pro celá data, protože nám uniknou spojitosti mezi parametry. Pro odhady bychom měli vždy využít všechny relevantní parametry.



Obrázek 4.2: Robustní vzdálenosti a odlehlé hodnoty



(a) Toleranční elipsoid pro $p = 3$



(b) Toleranční elipsa pro $p = 2$

Obrázek 4.3: Porovnání určení odlehlých hodnot

Příklad 4.3. Následující data vypovídají o cukrovce [20]. U 442 pacientů bylo měřeno 10 základních hodnot, 45 interakcí mezi nimi (kovariancí) a 9 kvadratických členů, celkem $p = 64$ parametrů. U každého pacienta je navíc známo, jak u něj behem jednoho roku nemoc postoupila. Z dat se má obvykle odvodit, zda je možné předvídat průběh nemoci u nových pacientů a jaké parametry mají na zhoršování nemoci největší vliv.

U tohoto experimentu budeme měnit hodnotu *ntrial*, tedy počet počátečních množin velikosti $p+1$, a budeme zkoumat jak dobré odhady dostáváme (jak velký je determinant výběrové variační matice), kolik C-stepů celkově jsme použili a jak se liší posouzení odlehlých hodnot v jednotlivých případech.

V Tabulce 4.2 znovu vidíme, že hodnota determinantu je o mnoho řádů pod strojovou přesností. Uvedené hodnoty vznikají v algoritmu po transformaci, standardizaci dat, která mimo jiné zvětšuje hodnoty determinantů. Po zpětné transformaci mají výsledné varianční matice prvky v řádu 10^{-5} a menším, takže při výpočtu determinantu s $p = 64$ skutečně dojde k podtečení, jak bylo popsáno v části 3.3.5. Místo abychom dostali výsledek zhruba kolem 10^{-300}, dostaneme numericky nulu. Proto doporučujeme pro data s velkým p používat logaritmování determinantu.

ntrial	#odlehlých hodnot	celkový #C-stepů	hodnota determinantu
500	176	1539	$1.7663 \cdot 10^{-77}$
400	177	1239	$1.2530 \cdot 10^{-77}$
300	179	946	$1.9503 \cdot 10^{-77}$
200	174	642	$3.2744 \cdot 10^{-77}$
100	177	341	$3.4085 \cdot 10^{-77}$
5000	176	15041	$1.1883 \cdot 10^{-77}$
50000	176	150041	$1.1883 \cdot 10^{-77}$
100000	176	300041	$1.1883 \cdot 10^{-77}$

Tabulka 4.2: Data o cukrovce pro různá *ntrial* a fixní $\alpha = 0.75$ bez náhodnosti výběru množin velikosti $p + 1$

Vezměme obvyklé nastavení $ntrial = 500$ jako referenční bod. Může možná překvapit, že jsme dostali menší odhad pro $ntrial = 400$. Zde se stane to, co bylo naznačeno v 2.3.4. Pro $ntrial = 500$ dostaneme jinou sadu 10 nejlepších množin, která se determinanty po dvou krocích zdá lepší, než sada řešení pro 400 počátečních množin H_1. V tomto případě přesně nastalo, že determinant množiny, která se do 10 nejlepších množin pro 500 počátečních množin nedostala, nakonec dalšími C-stepy klesne níže než pro množiny, jejichž determinant se do té doby zdál lepší.

V případě $ntrial = 100$ jsme dostali přibližně dvakrát větší determinant než pro $ntrial = 500$, tedy na první pohled horší řešení, přesto je pouze 13 pozorování klasifikováno jako odlehlá či neodlehlá hodnota opačně než pro $ntrial = 500$. Dostali jsme sice nepřesnější řešení, ale přitom se zdá blízké k řešení s obvyklým vstupním nastavením a navíc vyžadovalo téměř pětinu C-stepů.

Provedli jsme proto ještě pokus, kdy jsme algoritmus spustili s náhodným výběrem množin $p + 1$, jak bylo řečeno na začátku kapitoly, a 100krát jsme vypočítali hodnotu determinantu pro obě nastavení $ntrial = 500$ a $ntrial = 100$.

Výsledné zprůměrované hodnoty determinantu jsou vypsány v následující Tabulce 4.3. Průměrně tedy vychází odhad determinantu pro $ntrial = 100$ jen o málo horší, než s klasický nastavením za výrazného snížení počtu C-stepů.

ntrial	průměrný #C-stepů	průměrná hodnota determinantu
500	1540	$1.3902 \cdot 10^{-77}$
100	344	$1.72299 \cdot 10^{-77}$

Tabulka 4.3: Průměrné hodnoty determinantu pro 100 spuštění

Vratíme-li s k Tabulce 4.2, vidíme že zkoušením $ntrial > 500$ jsme došli k lepšímu odhadu s menším determinantem kovarianční matice. Překvapivě ovšem počet opačně kasifikovaných hodnot mezi případy pro $ntrial = 100$ a $ntrial = 100000$ je znovu pouze 13 hodnot ze 442. V tomto příkladu je vidět, že pro slušný odhad může stačit i méně počátečních množin H_1. Následující grafy na Obrázku 4.4 vykreslují robustní vzdálenosti jednotlivých pozorování pro 2 hodnoty *ntrial* a mezní hodnotu $\chi^2_{64,0.975}$, která odděluje odlehlé hodnoty od ostatních. Lze jasně vidět, že velké odlehlé hodnoty jsou určeny téměř identicky, rozdíly tudíž nastanou u mírných odlehlých hodnot.

Příklad naznačuje, že pokud nás z nějakého důvodu zajímají pouze velké odlehlé hodnoty, je pravděpodobně možné zmenšením veličiny *ntrial* výpočty urychlit a přesto dostat, co chceme.

Pro vykreslení Obrázku 4.4 používá *mcdcov* Mahalanobisovy vzdálenosti spočítané přímo pomocí inverze varianční matice. V části 3.3.4 jsme popsali možné důsledky pro stabilitu výpočtu. Tady uvedeme příklad.

Příklad. Pro $ntrial = 500$ dává *mcdcov* vektor Mahalanobisových vzdáleností

$$d_k = (d_k(1), \ldots, d_k(n)).$$

Pokud bychom počítali stejné vzdálenosti pomocí spektrálního resp. Choleského rozkladu dostaneme vektory vzdáleností

$$d_k^{spe} = (d_k^{spe}(1), \ldots, d_k^{spe}(n)) \quad \text{a} \quad d_k^{chol} = (d_k^{chol}(1), \ldots, d_k^{chol}(n)),$$

které se oba liší od d_k.

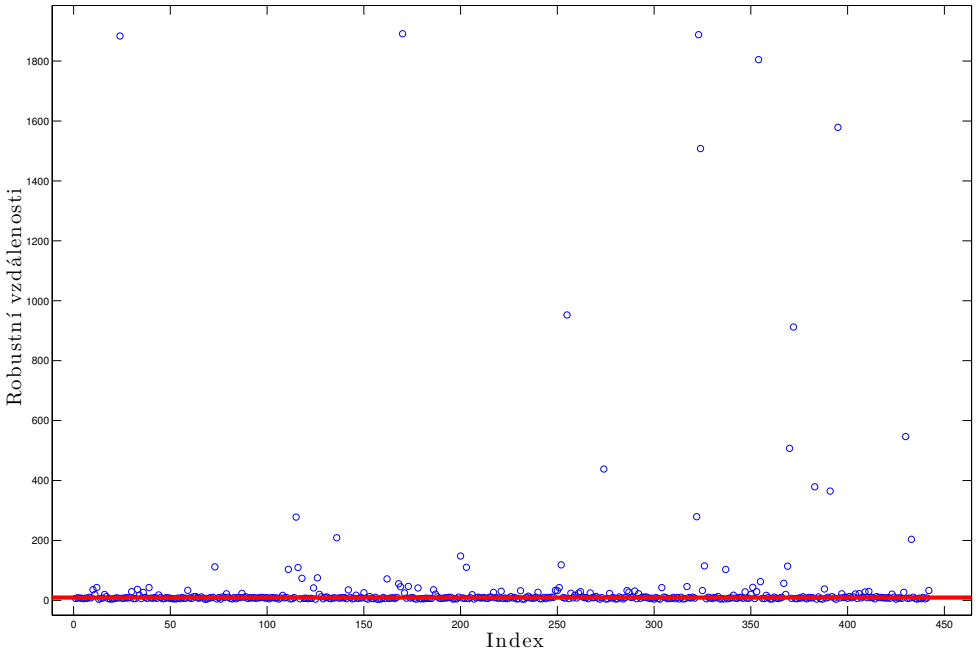
Pro spektrální rozklad dostaneme normu rozdílu a relativní chybu:

$$\|d_k - d_k^{spe}\| = 0.0025 \quad \text{a} \quad \frac{\|d_k - d_k^{spe}\|}{\|d_k^{spe}\|} = 5.626 \cdot 10^{-7}$$

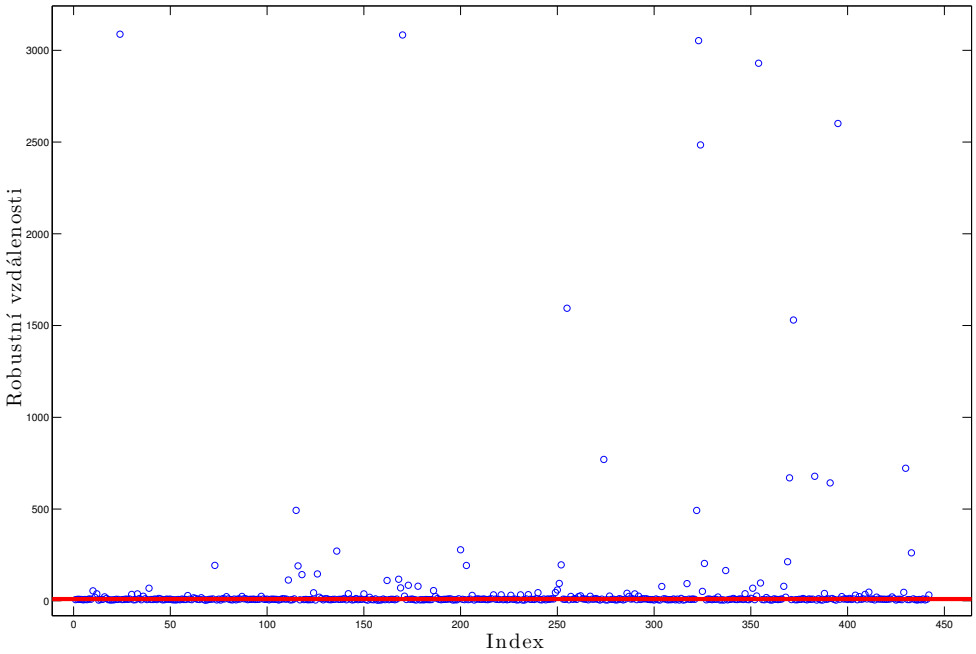
a pro Choleského rozklad dostaneme příslušné chyby

$$\|d_k - d_k^{chol}\| = 0.0023 \quad \text{a} \quad \frac{\|d_k - d_k^{chol}\|}{\|d_k^{chol}\|} = 5.1470 \cdot 10^{-7}.$$

Rozdíl je proto poměrně významný a může hrát roli při určení odlehlých hodnot. V tomto případě však tyto rozdíly vliv neměly, seřazení prvků d_k, d_k^{spe} i d_k^{chol} dávají stejně srovnané indexy pozorování. Ovšem pro jiná data s větším p to již pravda být nemusí.



(a) $ntrial = 100$



(b) $ntrial = 50000$

Obrázek 4.4: Graf vzdáleností a odlehlých hodnot pro dvě hodnoty *ntrial*

Příklad 4.4. Následující experiment je z dat o katetrech [21]. Tyto data popisují dětské pacienty s vrozenými srdečními vadami. Celkem bylo u $n = 12$ dětí změřeny 3 parametry: výška, váha a jako třetí parametr je vhodně určená délka katetru. Katetr se v podobných případech zavádí do srdce, aby se přesněji zjistila funkčnost srdce. Délku katetru obvykle musí doktor uhodnout, proto se z dat zkoumá, jestli lze odvodit vhodnou délku z výšky a váhy pacienta.

V tomto případě se pokusíme ukázat, že pro data o malém rozsahu se dá výrazně zmenšit počet C-stepů.

V datech se pokusíme najít odlehlé hodnoty, tedy případy, kdy bylo potřeba použít kratší či delší katetr než by se dalo podle tělesné stavby očekávat. Díky malému rozsahu dat lze jednoduše najít množinu s minimálním determinantem. Můžeme tedy zjišťovat, kolik počátečních množin a C-stepů stačí k nalezení globálního minima, tj přesného MCD estimátoru, pokud ho nalezneme.

Pokud nezasahujeme do nastavení *mcdcov*, použijí se hodnoty: $\alpha = 0{,}75$, $h = 10$, počet počátečních množin velikosti $p + 1$ je $ntrial = 495$ (místo 500, protože $\binom{n}{p+1} = \binom{12}{4} = 495$, projdeme všechny podmnožiny). V dalších pokusech budeme měnit pouze hodnotu proměnné *ntrial*, α a tím ani h měnit nebudeme.

Pro tento příklad navíc upravíme algoritmus na náhodný výběr $p+1$ množin. Protože případ s $ntrial = 495$ projde všechny možnosti, máme jistotu, že najde globální minimum. Pro $ntrial = 100, 20, 10$ a 5 se použil fastMCD algoritmus vždy 10krát. V tabulce je vždy pro danou hodnotu *ntrial* vypsáno, kolikrát se podařilo najít globální minimum, kolik bylo průměrně potřeba C-stepů k minimu (pouze z úspěšných pokusů, zaokrouhleno na celá čísla) a kolik celkově C-stepů bylo použito.

ntrial	#nalezení glob. minima	#C-stepů k min. det	celkový #C-stepů
495	vždy nalezne	6	1515
100	10/10	26	330
20	10/10	13	90
10	9/10	8	52
5	5/10	9	31

Tabulka 4.4: Data katetrů pro $\alpha = 0{,}75$

Jak je vidět v Tabulce 4.4, i pro pouhých 10 počátečních množin našel algoritmus fastMCD globální minimum s 90% úspěšností.. Důvodem je například to, že s původními nastaveními hned 111 ze 495 $p + 1$ množin vedlo na stejnou množinu s minimálním determinantem. Pouze pro $ntrial = 5$ jsme v polovině případů nedostali minimální determinant, tedy samotný MCD estimátor, ale jen odhady, které vždy označili správně jen jednu z odlehlých hodnot.

Evidentně bereme zbytečně mnoho $p+1$ množin, jednodušší by bylo brát rovnou všechny h množiny, kterých je $\binom{n}{h} = \binom{12}{10} = 66$ a použití množin velikosti $p+1$ úplně vynechat, protože opakovaně prochází stejné množiny H_1. Ušetřili bychom tím 500 C-stepů použité na množiny velikosti $p+1$ na prvotní určení množin H_1 a snížili bychom celkový počet použitých C-stepů přibližně na 160. Pro takto malá data sice není zrychlení příliš podstatné, pokud by se ovšem algoritmus pouštěl mnohokrát za sebou pro podobně malá data, už se pravděpodobně urychlení projeví. Do algoritmu by proto šlo přidat podmínku, že když $\binom{n}{h} \leq 500$, vezmeme

rovnou všechny množiny H_1.

Příklad 4.5. Na závěr se znovu na datech o cukrovce pokusíme snížit počet C-stepů následující heuristikou.

Pro každou z 500 počátečních $p+1$ množin vypočítáme Mahalanobisovy vzdálenosti se seškálovanou varianční maticí $\tilde{S}_k = \frac{1}{\sqrt[p]{\det S_k}} S_k$. Můžeme očekávat, že součet prvních h nejmenších Mahalanobisových vzdáleností, označíme $mahsum$, bude měřit kvalitu odhadu podobně jako samotný determinant varianční matice, protože malý $\det S_k$ implikuje, že i $mahsum$ bude malý [22]. To lze nahlédnout znovu z geometrické interpretace. Pokud se zmenší hodnota determinantu, zmenší se tím objem příslušného p-rozměrného elipsoidu. Body v elipsoidu se zhustí blíže ke středu tohoto elipsoidu, čímž se zmenší jejich Mahalanobisova vzdálenost.

Pro každou množinu tedy nejprve určíme $mahsum = \sum_{i=1}^{h} d_k(\pi(i))$, kde $d_k(\pi(i))$ jsou vzdálenosti přerovnané podle velikosti od nejmenší po největší. Zároveň srovnáme i množiny obsahující h příslušných indexů, použitých k výpočtu každé ze sum. Pro další iterace zkusíme vzít pouze 100 množin H_1, které mají nejmenší hodnotu $mahsum$, protože by se dalo předpokládat, že mezi nimi najdeme dobrý odhad, nebo přímo globální minimum. Dál už pokračujeme stejně.

Snížíme tím výrazně počet C-stepů. Výpočtová náročnost ve výpočtech s $p+1$ množinami se řádově nezmění, protože přibylo pouze dělení prvků matice S_k hodnotou $\frac{1}{\sqrt[p]{\det S_k}}$ a součet h čísel.

Podíváme se, jak se změní výkonnost algoritmu.

Pro $ntrial = 500$ a $ntrial = 230$ použijeme $mcdcov$ s přidanou náhodností výběru $p+1$ množin, ale bez úpravy popsané v této sekci. Hodnota 230 je vybrána, proto, že bude potřebovat podobný počet C-stepů jako třetí měřený případ, který úpravy využije přesně jak je zde popsána. Dostaneme tedy srovnání, zda za stejný počet C-stepů jako pro $ntrial = 230$ získáme lepší odhad determinantu.

Všechny tři případy pustíme 1000krát za sebou a spočítáme průměrnou hodnotu determinantu a průměrný počet celkově použitých C-stepů, zaokrouhlených na celá čísla.

případ	průměrný #C-stepů	průměrná hodnota determinantu
$ntrial = 500$	1538	$1.4463 \cdot 10^{-77}$
$ntrial = 230$	734	$1.9503 \cdot 10^{-77}$
upravený $mcdcov$	742	$1.7419 \cdot 10^{-77}$

Tabulka 4.5: Srovnání výkonnosti $mcdcov$ a upraveného algoritmu

Modifikací algoritmu jsme sice snížili počet C-stepů přibližně na polovinu, ale z Tabulky 4.5 se zdá, že je zkoumaná úprava sice o něco lepší než přímé snížení počtu C-stepů snížením hodnot $ntrial$, ale ani nedává tak dobrý odhad jako původní verze algoritmu s $ntrial = 500$. Úpravu by bylo dobré ještě podrobit dalšímu zkoumání s jinými, většími daty.

Závěr

Práce se zabývala estimátorem Minimum covariance determinant. Na úvod shrnula v čem spočívají problémy spojené s odhadem střední hodnoty a rozptylu, když máme mnohorozměrná data obsahující odlehlé hodnoty. Vysvětlovala tak úlohu robustní statistiky, která se snaží získat nezkreslené výsledky nehledě na odlehlých hodnotách.

Tím se práce dostala k MCD estimátoru a jeho algoritmizaci fastMCD. Popsala vlastnosti estimátoru, který je obecně těžké určit kvůli kombinatorické složitosti počtu množin, které bychom museli projít. Proto rozvádí jak funguje fastMCD algoritmus, který sice obecně nedává samotný estimátor, jen jeho odhad, ale zato urychluje výpočet natolik, aby šel algoritmus použít na mnohorozměrná data.

Dále se už práce posouvá od shnutí již známých faktů k určení numerických vlastností algoritmu fastMCD, které se zatím, pokud víme, podrobně v literatuře neobjevily. Zaměřuje se především na výpočetní náklady a stabilitu. Výpočetní náklady byly rozšířeny tak, aby nezávisely pouze na velikosti úlohy, ale i na počtu parametrů, čímž reflektují potřebu používat fastMCD i na mnohem větší úlohy, než jak bylo v původním článku zamýšleno.

Stabilita je určena nejen pro samotný algoritmus fastMCD a jeho verzi z MATLABu, ale zahrnuje také jiné možnosti výpočtů v C-stepech. Došli jsme zde k závěru, že je možné snížit výpočtovou náročnost použitím spektrálního nebo Choleského rozkladu, ovšem právě na úkor stability.

Navrhujeme také několik zlepšení stability fastMCD. Především diskutujeme úplné odstranění invertování plných matic, které se obecně nikdy nedoporučuje, a přidání logaritmování při výpočtu determinantu pro data s velkým počtem proměnných, kde hrozí přetečení nebo podtečení možností reprezentace čísel v počítači.

Nakonec obsahuje práce numerické experimenty. Ukazujeme zde několik poznatků. Jedním z nich je to, že ani když dostaneme jako výsledek přesnou hodnotu estimátoru, nemusí to nutně znamenat, že byly zcela správně detekovány odlehlé hodnoty. To ovšem vzhledem k heuristické povaze celé myšlenky MCD estimátoru není příliš překvapivé.

Zabývali jsme se i tím, proč není žádoucí uměle snižovat počet proměnných, se kterými počítáme. Uvedli jsme, že fastMCD používá pro malá data příliš mnoho C-stepů, čímž prochází některé množiny i několikrát a navrhli jsme zjednodušení, které vede ke snížení výpočtové náročnosti, aniž bychom ztratili přesnost řešení.

Nejdůležitější výsledek vyplývá z pokusů se snížením počtu počátečích množin, které pro výpočet používáme, pro větší data. Přímočaře jsme snižovali počet C-stepů a zkoumali jsme dopad na hodnoty determinantu. Podle našich výsledků se zdá, že často pro dobrý odhad estimátoru stačí mnohem menší počet C-stepů než kolik fastMCD obvykle používá. Dala by se tak výrazně zmenšit výpočtová náročnost algoritmu, přičemž bychom neměli dostat o mnoho horší odhad estimátoru.

V posledním z experimentů jsme uvedli nový návrh na vylepšení výběru počátečních množin. Myšlenku jsme zakomponovali do stávajícího algoritmu a zkoumali jsme zda dosáhneme dobrých odhadů estimátoru při snížení počtu C-stepů.

Z našeho pokusu vyplynulo, že námi upravený algoritmus dosahuje o něco lepších odhadů, než kdybychom jednoduše zmenšili počet používaných C-stepů, na druhou stranu jsme dostali horší odhady než pro klasické nastavení algoritmu. Zdá se, že úprava funguje, i když ne tak dobře, jak bychom chtěli. Bylo by proto vhodné dál úpravu zkoumat s dalšími daty.

Seznam použité literatury

[1] F. R. Hampel, E. M. Ronchetti, P. J. Rousseeuw, and Stahel W. A. *Robust statistics*. John Wiley and Sons, 1986. ISBN: 0-471-82921-8.

[2] M. Hubert and M. Debruyne. Minimum covariance determinant. *WIREs Computational Statistics*, 2:36–43, 2010.

[3] P. J. Rousseeuw and K. Van Driessen. A Fast Algorithm for the Minimum Covariance Determinant Estimator. *Technometrics*, 41(3):212–223, 1999.

[4] M. Lichman. UCI machine learning repository, http://archive.ics.uci.edu/ml, 2013. University of California, Irvine, School of Information and Computer Sciences.

[5] K. Van Driessen and B. Rombouts. LIBRA: a MATLAB library for robust analysis, https://wis.kuleuven.be/stat/robust/libra, 28. 7. 2016. KU Leuwen.

[6] M. Hubert and M. Debruyne. Breakdown value. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1(3):296–302, 2009.

[7] J. Jurečková. *Robustní statistické metody*. Nakladatelství Karolinum, 2001. ISBN: 80-246-0259-8.

[8] P. J. Rousseeuw. Least median of squares regression. *Journal of the Americas Statistical Association*, pages 871–880, 1984.

[9] P. J. Rousseeuw. Multivariate estimation with high breakdown point. *Mathematical Statistics and Application*, pages 283–297, 1985.

[10] C. Croux and G. Haesbroeck. Influence function and efficiency of the minimum covariance determinant scatter matrix estimator. *Journal of Multivariate Analysis*, 71(2):161 – 190, 1999.

[11] R. Grübel. A minimal characterization of the covariance matrix. *Metrika*, 35(1):49–52, 1988.

[12] P. J. Rousseeuw and A. M. Leroy. *Robust Regression and Outlier Detection*. John Wiley & Sons, Inc., New York, NY, USA, 1987.

[13] J. Duintjer Tebbens, I. Hnětynková, M. Plešinger, Z. Strakoš, P. Tichý. *Analýza metod pro maticové výpočty: základní metody*. Matfyzpress, 2012. ISBN: 978-80-7378-201-6.

[14] Gene H. Golub and Charles F. Van Loan. *Matrix Computations (4th Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 2013.

[15] N. Higham. *Accuracy and Stability of Numerical Algorithms*. Society for Industrial and Applied Mathematics, second edition, 2002.

[16] P. J. Rousseeuw and K. van Driessen. fastmcd.m , https://www.mathworks.com/matlabcentral/fileexchange/21384-continuous-sound-and-vibration-analysis/content/fastmcd.m.

[17] The MathWorks, Inc. Matlab 2012a. Natick, Massachusetts, United States.

[18] M. Lichman. UCI machine learning repository, http://archive.ics.uci.edu/ml, 2013. University of California, Irvine, School of Information and Computer Sciences.

[19] E. Roelant, S. Van Aelst, and G. Willems. The minimum weighted covariance determinant estimator. *Metrika*, 70:177–204, 2009.

[20] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–451, 2004.

[21] S. Weisberg. A statistic for allocating c_p to individual cases. *Technometrics*, 23:27–31, 1981.

[22] E. Roelant, S. Van Aelst, and G. Willems. The minimum weighted covariance determinant estimator. *Metrika - online*, 70:177–204, 2009.

[23] L. Eldén. *Matrix Methods in Data Mining and Pattern Recognition*. SIAM, 2007. ISBN: 978-0-898716-26-9.

[24] V. Dupač, M. Hušková. *Pravděpodobnost a matematická statistika*. Karolinum, 2013. ISBN: 978-80-246-2208-8.

[25] J. Kalina, J. Duintjer Tebbens, and A. Schlenker. Robustness of high-dimensional data mining. *ITAT 2014*, 2014.

[26] H.P. Lopuhaä and P. J. Rousseeuw. Breakdown points of affine equivariant estimators of multivariate location and covariance matrices. *Tha Annals of Statistics*, 19:229–248, 1991.