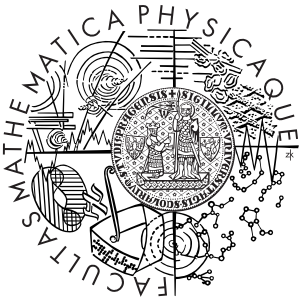
Univerzita Karlova v Praze

Matematicko-fyzikální fakulta

DIPLOMOVÁ PRÁCE



Kirill Troshkov

Statistická analýza intervalových dat

Katedra pravděpodobnosti a matematické statistiky

Vedoucí diplomové práce: prof. RNDr. Jaromír Antoch, CSc.

Studijní program: Matematika

Studijní obor: Finanční a pojistná matematika

Praha 2011

Na tomto místě bych chtěl poděkovat svému vedoucímu práce panu prof. RNDr. Jaromír Antoch, CSc. za jeho hodnotné rady a připomínky k vypracování této diplomové práce.

Prohlašuji, že jsem tuto diplomovou práci vypracoval 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 v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

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

Obsah

Název práce: Statistická analýza intervalových dat
Autor: Kirill Troshkov
Katedra: Katedra pravděpodobnosti a matematické statistiky
Vedoucí diplomové práce: prof. RNDr. Jaromír Antoch, CSc.
e-mail vedoucího: Jaromir.Antoch@mff.cuni.cz

Abstrakt: Tradiční statistická analýza začíná výpočtem základních statistických charakteristik jako je výběrová střední hodnota E, výběrový rozptyl V, kovariance či korelace. Při výpočtu těchto charakteristik se většinou předpokládá, že odpovídající hodnoty dat jsou přesně známé. Ve skutečnem světě existují situace, kdy je možné získat více vypovídající informace tím, že soubor statistických dat bude intervalového typu. Například, naměřená denní maximální a minimální teplota dává realističtější pohled na počasí než obyčejné průměrné hodnoty. Při analýze životního prostředí dostáváme naměřené hodnoty znečištění jezera $x(t)$ v různých časových okamžicích t, přičemž bychom potřebovali odhadnout statistické charakteristiky jako je střední hodnota, rozptyl nebo korelace s jinými měřeními. Jiný příklad je z finančního prostředí. Minimum a maximum cen transakcí, denně zaznamenané pro nějaký soubor akcií poskytuje víc relevantních údajů pro finanční experty, kteří vyhodnocují akcie a volatilitu ve stejný den. Pro tyto a mnohé další případy musíme modifikovat existující statistické postupy, abychom je mohli aplikovat na data intervalového typu. V této práci se pokusíme rozebrat statistické algoritmy, jejich složitost a modifikace vhodné pro aplikaci na intervalová data v případě výpočtu střední hodnoty, rozptylu, kovariance a korelace.

Klíčová slova: intervalová data, střední hodnota, rozptyl, kovariance, korelace, intervalová neurčitost, výpočetní složitost.

Title: Statistical analysis of interval data
Author: Kirill Troshkov
Department: Department of Probability and Mathematical Statistics
Supervisor: prof. RNDr. Jaromír Antoch, CSc.
Supervisor's e-mail address: Jaromir.Antoch@mff.cuni.cz

Abstract: Traditional statistical analysis starts with computing the basic statistical characteristics such as the population mean E, population variance V, covariance and correlation. In computing these characteristics, it is usually assumed that the corresponding data values are known exactly. In real life there are many situations in which a more complete information can be achieved by describing a set of statistical units in terms of interval data. For example, daily temperatures registered as minimum and maximum values offer a more realistic view on the weather conditions variations with respect to the simple average values. In environmental analysis, we observe a pollution level $x(t)$ in a lake at different moments of time t, and we would like to estimate standard statistical characteristics such as mean, variance and correlation with other measurements. Another example can be given by financial series. The minimum and the maximum transaction prices recorded daily for a set of stocks represent a more relevant information for experts in order to evaluate the stocks tendency and volatility in the same day. We must therefore modify the existing statistical algorithms to process such interval data. In this work we will analyze algorithms and their modifications for computing various statistics under interval uncertainty and their computational complexity in case of computing population mean, population variance, covariance and correlation.

Keywords: interval data, mean, variance, covariance, correlation, interval uncertainty, computational complexity.

Kapitola 1

Úvod

1.1 Struktura práce

V první části práce připomeneme základy intervalové algebry a dále vysvětlíme pojem výpočetní složitost algoritmu, se kterým budeme pracovat v následujících kapitolách.

Třetí kapitola je věnovaná formulaci problému, kdy potřebujeme odhadnout statistické charakteristiky pro intervalová data.

Ve čtvrté kapitole analyzujeme problém, kde rozebíráme výhody, omezení a nevýhody linearizace a tradičních intervalových metod při určování základních statistických charakteristik.

Pátá část se zaobírá různými druhy intervalů, na které můžeme narazit v praxi. Každý druh těchto intervalu má vlastní specifika a potřebné předpoklady pro to, abychom byli schopni najít odhady požadovaných statistických charakteristik a to především u odhadu pro horní intervalovou mez rozptylu.

Šestá kapitola vysvětluje výpočet statistik pro intervalová data. První část této kapitoly je z velké části věnovaná metodám pro určení dolní intervalové meze pro rozptyl, druhá část ukazuje různé přístupy k výpočtu horní intervalové meze pro rozptyl a poslední část této kapitoly přibližuje situaci pro některé další vybrané statistické charakteristiky.

V sedmé kapitole se pokusíme implementovat dříve vysvětlené postupy do systému Mathematica a aplikovat je na skutečná data a náhodně vygenerovaná data.

1.2 Úvodní poznámky

Statistické charakteristiky byly vytvořeny hlavně pro statistickou analýzu dat s určitými hodnotami. Ve skutečném světě však narážíme na situace, ve kterých použití daných hodnot může způsobit ztrátu důležitých informaci.

Existují případy, ve kterých je možné získat více vypovídající informace tím, že soubor statistických dat bude intervalového typu. Například, naměřená denní maximální a minimální teplota dává realističtější pohled na počasí než obyčejné průměrné hodnoty. Při analýze životního prostředí dostáváme naměřené hodnoty znečištění jezera $x(t)$ v různé časové okamžiky t, přičemž bychom potřebovali odhadnout statistické charakteristiky jako je střední hodnota, rozptyl nebo korelace s jinými měřeními. Jiný příklad je z finančního prostředí. Minimum a maximum z cen transakcí, denně zaznamenané pro nějaký soubor akcii, poskytuje víc relevantních údajů pro finanční experty, kteří vyhodnocují akcie a volatilitu ve stejný den. Pro tyto a mnohé další případy musíme modifikovat existující statistické postupy, abychom je mohli aplikovat na data intervalového typu. V této práci se pokusíme rozebrat vybrané statistické algoritmy, jejich složitost a modifikace vhodné pro aplikaci na intervalová data.

Kapitola 2

Připomenutí

V této kapitole připomeneme pravidla při práci s intervaly a objasníme pojem časová složitost algoritmu.

2.1 Intervalová algebra

Interval $[a, b]$ při $a \leq b$ je definován jako množina reálných čísel mezi a a b:

$$[a, b] = \{x |\ a \leq x \leq b\}.$$

Degenerovaný interval je interval typu $[a, a]$, který je ekvivalentní reálnému číslu.

Symboly $\in, \supset, \cap, \cup$ jsou použité ve stejném významu jako v teorii množin. Například $[c, d] \supset [a, b]$ známena, že interval $[a, b]$ je obsažen jako množina v intervalu $[c, d]$.

Dále $[a, b] = [c, d] \Leftrightarrow a = b, c = d$.

Pokud $\bullet$ je jeden ze symbolů $+, -, \cdot, /$, definujeme aritmetické operace pro intervaly jako:

$$[a, b] \bullet [c, d] = \{x \bullet y |\ a \leq x \leq b, c \leq y \leq d\},$$

přitom nedefinujeme $[a, b]/[c, d]$ pokud $0 \in [c, d]$.

Nechť A je množina intervalů. Nechť $[a, b], [c, d]$ jsou prvky A, potom platí následující vzorce:

$$[a,b] + [c,d] = [a+c, b+d]$$
$$[a,b] - [c,d] = [a-d, b-c]$$
$$[a,b] \cdot [c,d] = [min(ac,ad,bc,bd), max(ac,ad,bc,bd)]$$

pokud $0 \notin [c,d]$, potom:

$$[a,b]/[c,d] = [a,b] \cdot [1/d, 1/c].$$

2.2 Časová složitost algoritmu

Časovou složitostí algoritmu chápe [13] jako závislost jeho časových nároků na velikost konkrétního řešeného problému nebo konkrétních vstupních dat. Časovou složitost určujeme pouze počtem elementárních operací (kroků algoritmu), které budou provedeny při výpočtu programu s danými vstupními daty. Je to tedy funkce, která každé hodnotě N udávající velikost konkrétního řešeného problému přiřazuje počet operací vykonaných při výpočtu podle daného algoritmu. Tato funkce je zpravidla rostoucí.

Pro nás bude podstatné, který algoritmus je rychlejší pro velké hodnoty N. Mluvíme pak o asymptotické časové složitosti a ze dvou algoritmů prohlásíme za lepší ten, který má asymptotickou složitost menší.

Při odvozování asymptotické složitosti algoritmů je pro nás podstatná pouze rychlost růstu příslušné funkce. Není důležité znát přesný počet provedených elementárních operací a bylo by také velmi pracné a obtížné hledat vzorec, který by určoval počet operací detailně. Proto o algoritmu, jehož časovou složitost charakterizuje předpis $2N^2 + 3N + 1$, stačí zjistit, že s rostoucím N jsou jeho časové nároky úměrné N^2, neboli, že příslušná funkce časové složitosti je kvadratická. Tuto skutečnost bývá zvykem zapisovat ve tvaru $O(N^2)$.

Má-li přesné vyjádření funkce časové složitosti algoritmu tvar součtu, pak rychlost růstu této funkce je určována rychlostí růstu nejrychleji rostoucího členu. V praxi používané algoritmy mívají většinou některou z následujících složitosti: $O(log(N)), O(N), O(N \cdot log(N)), O(N^2), O(N^2 \cdot log(N)), O(N^3), ..., O(2^N)$. Všechny algoritmy, jejichž funkci časové složitosti můžeme shora omezit polynomem v N, označujeme jako algoritmy polynomiální. Algoritmy s větší časovou složitostí nazýváme exponenciální.

Další důležitý pojem je NP problém. NP (zkratka nedeterministicky polynomiální) je množina problémů, které lze řešit v polynomiálně omezeném čase na počítači, který umožňuje v každém kroku rozvětvit výpočet na n větví, v nichž se posléze řešení hledá současně. Ekvivalentně se hovoří o stroji, který na místě rozhodování uhodne správnou cestu výpočtu. Alternativně lze tyto problémy definovat tak, že je to množina problému, u kterých lze pro dodaný výsledek v polynomiálním čase ověřit jeho správnost (ale obecně nikoliv nalézt řešení v polynomiálním čase).

Kapitola 3

Formulace problému

3.1 Statistické charakteristiky

V mnohá oborech je pro nás důležitý výpočet statistických charakteristik. Například měření denní maximální a minimální teploty, denně zaznamenané maximum a minimum cen akcií nebo při analýze životního prostředí naměřené hodnoty znečištění jezera $x(t)$ v různých časech t, tady všude jsou pro nás významné základní statistické charakteristiky jako je průměr, rozptyl, kovariance či korelace s ostatními naměřenými hodnotami.

Pro každou z těchto charakteristik C máme vyjádření $C(x_1, ..., x_n)$, které nám umožňuje získávat odhady C na základě obdržených hodnot $x_1, ..., x_n$. Například:

- běžná statistika pro odhad střední hodnoty rozdělení pravděpodobnosti je populační průměr $E(x_1, ..., x_n) = \frac{1}{n} \cdot (x_1 + ... + x_n)$;
- klasická statistika pro odhad rozptylu V je výběrový rozptyl populace

$$V(x_1, ..., x_n) = \frac{1}{n} \cdot \sum_{i=1}^{n} (x_i - E)^2,$$

$$\text{resp. } V_n(x_1, ..., x_n) = \frac{1}{n-1} \cdot \sum_{i=1}^{n} (x_i - E)^2,$$

  kde $E \stackrel{def}{=} \frac{1}{n} \cdot \sum_{i=1}^{n} x_i$.

Výběrový rozptyl populace je často určován pomocí jiného vzorce $V = M - E^2$, kde $M = \frac{1}{n} \cdot \sum_{i=1}^{n} {x_i}^2$ je populační druhý moment.

V této diplomové práci budeme pracovat s odhadem V pro intervalová data, odhad $V_n = \frac{n}{n-1} \cdot V$ jednoduše získáme pomocí odhadu V.

3.2 Neurčitost intervalu

Poměrně dost často se stává, že máme možnost měřit hodnoty jenom s intervalovou neurčitosti. Například, pokud nebylo naměřeno žádné znečištění vody, hodnota znečištěni v může byt kdekoliv mezi 0 a detekčním limitem senzorů L. Jediná informace, kterou máme o hodnotě v je, že v patří do intervalu [0,L], nemáme žádnou informaci o pravděpodobnosti jiných hodnot z tohoto intervalu.

Jiným příkladem je zkoumaní vlivu znečištěné vody na ryby. Provádíme kontroly ryb jednou denně, pokud ryba byla naživu třetí den, ale mrtvá čtvrtý den, potom jedinou informaci o délce života ryby je, že tato hodnota se nachází někde uvnitř intervalu [3,4]. Nemáme žádnou informaci o rozložení jiných hodnot v tomto intervalu.

Většinou víme výsledek měření $\widetilde{x}$ nějaké požadováné hodnoty x a známe horní mez Δ absolutní hodnoty chyby měření $|\Delta x|$, $\Delta x \stackrel{def}{=} \widetilde{x} - x$ (tato horní mez je poskytovaná výrobcem měřicího zařízení), údaje o pravděpodobnosti jiných hodnot $\Delta x \in [-\Delta, \Delta]$ nemáme. Po provedení měření máme k dispozici jedinou informaci o skutečné hodnotě x a to takovou, že tato hodnota patří do intervalu $[\widetilde{x} - \Delta, \widetilde{x} + \Delta]$.

Dalším zdrojem neurčitosti intervalu je „soukromí“. Například v biomedicínských systémech statistická analýza dat často vede k zpřesnění lékařských postupů. Abychom zachovali soukromí pacientů, nemůžeme používat přesné hodnoty jejich osobních údajů. Pro každý údaj stanovíme pevnou hodnotu a každému pacientovi přiřadíme odpovídající rozsah. Například, místo toho, abychom uvedly přesný věk pacienta, zaznamenáme věk mezi 0 a 10, 10 a 20, 20 a 30, atd. To vede v řadě případů k použití statistické analýzy založené na intervalových datech.

3.3 Problém odhadu statistických charakteristik při neurčitosti intervalu

Předpokládejme, že místo skutečných hodnot $x_1, ..., x_n$ máme k dispozici pouze intervaly $\mathbf{x}_1 = [\underline{x}_1, \overline{x}_1], ..., \mathbf{x}_n = [\underline{x}_n, \overline{x}_n]$, které obsahují skutečné hodnoty měřených veličin. Obecně, pro různé hodnoty $x_i \in \mathbf{x}_i$ máme různé hodnoty odpovídajících

statistických charakteristik $C(x_1, ..., x_n)$. Protože všechny hodnoty $x_i \in \mathbf{x}_i$ jsou možné, potom všechny hodnoty $C(x_1, ..., x_n)$ odpovídající $x_i \in \mathbf{x}_i$ jsou možné odhady pro statistické charakteristiky. Možné hodnoty odpovídajících statistických charakteristik pro intervalová data $\mathbf{x}_1, ..., \mathbf{x}_n$ máji rozsah (obor hodnot)

$$C(\mathbf{x}_1, ..., \mathbf{x}_n) \stackrel{def}{=} \{C(x_1, ..., x_n) |\ x_1 \in \mathbf{x}_1, ..., x_n \in \mathbf{x}_n\}.$$

Naším cílem je upravit existující statistické algoritmy tak, aby výpočet odpovídal tomuto rozsahu (oboru hodnot).

Kapitola 4

Analýza problému

4.1 Střední hodnota

Tuto kapitolu začneme u odhadu nejjednodušší statistické charakteristiky, tj. střední hodnoty, kterou běžně odhadujeme výběrovým průměrem. Je zřejmé, že aritmetický průměr E je monotonní rostoucí funkce každé z n proměnných $x_1, ..., x_n$. Nejmenší možné hodnoty $\underline{E}$ dosáhneme tak, že každé hodnotě x_i přiřadíme nejmenší možnou hodnotu, tj. $x_i = \underline{x}_i$, a největší možnou hodnotu $\overline{E}$ získáme, když $x_i = \overline{x}_i$ pro všechna i. Rozsah $\mathbf{E}$ pro E je roven $[E(\underline{x}_1, ..., \underline{x}_n), E(\overline{x}_1, ..., \overline{x}_n)]$, kde $\underline{E} = \frac{1}{n} \cdot (\underline{x}_1 + ... + \underline{x}_n)$ a $\overline{E} = \frac{1}{n} \cdot (\overline{x}_1 + ... + \overline{x}_n)$.

4.2 Rozptyl

Další rozšířenou statistikou je rozptyl. Na rozdíl od střední hodnoty, rozptyl[1] V nezávisí monotonně na x_i, takže postup uvedený výše zde nebude fungovat. Obecně, problém výpočtu přesného rozsahu hodnot rozptylu pro intervalová data má NP-složitost (viz [6], [7], [8], [9]), což znamená, že při nejhorším možném scénáři doba výpočtu roste exponenciálně s n. Populační rozptyl je konvexní funkce, proto můžeme použit efektivní algoritmus, abychom našli minimum $\underline{V}$ konvexní funkce $V(x_1, ..., x_n)$ na konvexní krychli $\mathbf{x}_1 \times ... \times \mathbf{x}_n$. Pro tuto konvexní funkci můžeme popsat algoritmus, který je rychlejší než pro obecný konvexní případ. Dolní mez $\underline{V}$ můžeme vypočítat v čase $O(n \cdot log(n))$.

[1] V dalším textu budeme pro jednoduchost používat slovo rozptyl i pro výběrový (populační) rozptyl.

Pro horní mez $\overline{V}$ je to složitější. Je známo, že maximum konvexní funkce na konvexní množině, je dosaženo v jednom z extrémních bodů. Maximum $\overline{V}$ je dosaženo v jednom z extrémních bodů konvexní krychli $\mathbf{x}_1 \times ... \times \mathbf{x}_n$, tj., když pro každé i proměnná x_i je rovna jedné z koncových hodnot: $x_i = \underline{x}_i$ nebo $x_i = \overline{x}_i$. Máme 2^n kombinaci pro n odpovídajících koncových bodů, takže můžeme spočítat $\overline{V}$ tak, že najdeme maximum z 2^n odpovídajících hodnot.

Jak je uvedeno v [2], tak pro jakýkoliv vektor $\mathbf{x} \in \mathbb{R}^n$ platí následující vzorec pro výpočet rozptylu V

$$V = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - E)^2 = \frac{1}{n}\left[\sum_{i=1}^{n} {x_i}^2 - \frac{1}{n-1}\sum_{1\le i\ne j\le n} x_i x_j\right] \tag{4.1}$$

$$= \frac{1}{n(n-1)}\sum_{1\le i<j\le n}(x_i - x_j)^2 \tag{4.2}$$

Označme $\mathcal{K} = \mathbf{x}_1 \otimes ... \otimes \mathbf{x}_n \subset \mathbb{R}^n$. Řešení úlohy o nalezení minimální a maximální hodnoty V je na hranici $\mathcal{K}$. Navíc řešení úlohy o nalezení maxima V se shoduje s jedním nebo více vrcholy $\mathcal{K}$. Z (4.2) plyne, že V nabývá svých extrémních hodnot v $\mathbf{x} = (a, ..., a)$, $a \in \mathbb{R}^1$, protože $\partial V/\partial x_i = 0$ pro $i = 1, ..., n$, jenom a jenom tehdy, když $x_1 = ... = x_n$. Z (4.2) je zřejmé, že V nabývá jenom svého minima v $\mathbb{R}^n$, protože $\max_{\mathbf{x}\in\mathbb{R}^n} V = \infty$ kdežto $\min_{\mathbf{x}\in\mathbb{R}^n} V = 0$ pro všechny $\mathbf{x} = (a, ..., a)$, $a \in \mathbb{R}^1$. Proto V nabývá svého minima i maxima na hranici $\mathcal{K}$, kterou označíme jako $\partial\mathcal{K}$. Musíme říci, že maximum V může také ležet uvnitř $\mathcal{K}$. Tato řešení odpovídají průniku $\mathcal{K}$ a přímky procházející počátkem a bodem $(1, ..., 1)$.

Ukážeme, že řešení úlohy o nalezení maxima V se shoduje s jedním nebo více vrcholy $\mathcal{K}$. Abychom to ukázali, uvažujme úsečku $\mathbf{U} \subset \partial\mathcal{K}$, která je tvořená hraničními body $\mathcal{K}$ a je určena dvěma vrcholy $\mathcal{K}$. Každou takovou úsečku jsme schopní zapsat ve tvaru $\mathbf{X} = \mathbf{A} + t\vec{\mathbf{v}}$, kde $t \in [0, 1], \mathbf{A}, \mathbf{B}$ jsou vrcholy $\mathcal{K}$ a $\vec{\mathbf{v}} = \mathbf{B} - \mathbf{A}$. Pro každé pevné $t \in [0, 1]$ máme jeden bod $\mathbf{x}(t)$ úsečky $\mathbf{X}$ a vypočítáme jeho rozptyl, tj.

$$f(t) = var(\mathbf{A} + t\vec{v}) = var\mathbf{x}(t) =$$

$$\frac{1}{n(n-1)}\left[t^2\sum_{1\le i<j\le n}(v_i - v_j)^2 + 2t\sum_{1\le i<j\le n}(A_i - A_j)(v_i - v_j) + \sum_{1\le i<j\le n}(A_i - A_j)^2\right].$$

Je zřejmé, že $f(t)$ je kvadratická funkce v t na $[0, 1]$ s kladným koeficientem u členu

t^2, takže táto funkce nabývá svého maxima v hraničních bodech intervalu $[0, 1]$, tj. buď v **A** nebo **B** nebo v obou vrcholech.

V obecném případě výpočet $\overline{V}$ je NP-složitý. Tato NP-složitost znamená, že neexistuje obecný algoritmus, který by vždy byl schopný vypočítat $\overline{V}$ v konečném čase.

V některých speciálních případech jsme schopní najít $\overline{V}$ v polynomickém čase.

4.3 Linearizace

Z praktického hlediska nepotřebujeme přesné intervaly, vždycky můžeme využít aproximace pomocí linearizace. Například, pokud neurčitost je způsobená chybami měření Δx_i a tyto chyby jsou malé, potom můžeme ignorovat členy, které mají druhou a vyšší mocninu v Δx_i, a dostaneme přiměřené odhady odpovídajících statistických charakteristik. Abychom odhadli interval statistiky $C(x_1, ..., x_n)$ na intervalech $[\underline{x}_i, \overline{x}_i]$,...,$[\underline{x}_n, \overline{x}_n]$, rozvineme funkci C pomocí Taylorovýho rozvoje v prostředním bodě množiny intervalů $\widetilde{x}_i \stackrel{def}{=} (\underline{x}_i + \overline{x}_i)/2$ a ponecháme členy bez vyšších mocnin. Ve výsledku zaměníme původní statistiku její linearizací $C_{lin}(x_1, ..., x_n) = C_0 - \sum_{i=1}^{n} C_i \cdot \Delta x_i$, kde $C_0 \stackrel{def}{=} C(\widetilde{x}_1, ..., \widetilde{x}_n)$, $C_i \stackrel{def}{=} \frac{\partial C}{\partial x_i}(\widetilde{x}_1, ..., \widetilde{x}_n)$ a $\Delta x_i \stackrel{def}{=} \widetilde{x}_i - x_i$. Pro každé i, kde $x_i \in [\underline{x}_i, \overline{x}_i]$, může rozdíl Δx_i nabývat všech možných hodnot od $-\Delta_i$ do Δ_i, kde $\Delta_i \stackrel{def}{=} (\overline{x}_i - \underline{x}_i)/2$. Díky lineární aproximaci jsme schopni odhadnout interval charakteristiky C jako $[C_0 - \Delta, C_0 + \Delta]$, kde $\Delta \stackrel{def}{=} \sum_{i=1}^{n} |C_i| \cdot \Delta_i$.

Pokud budeme uvažovat za statistiku populační rozptyl $C = V$, potom $C_i = \frac{\partial V}{\partial x_i} = \frac{2}{n} \cdot (\widetilde{x}_i - \widetilde{E})$, kde $\widetilde{E}$ je průměr středních bodů $\widetilde{x}_i$ a $C_0 = V_0 = \frac{1}{n} \cdot \sum_{i=1}^{n} (\widetilde{x}_i - \widetilde{E})^2$ je rozptyl hodnot středních bodů $\widetilde{x}_1, ..., \widetilde{x}_n$. Takže pro rozptyl je $\Delta = \frac{2}{n} \cdot \sum_{i=1}^{n} |\widetilde{x}_i - \widetilde{E}| \cdot \Delta_i$.

Pro rozptyl, člen s druhou mocninou v Taylorovým rozvoji je roven $\frac{1}{n} \cdot \sum_{i=1}^{n} (\Delta x_i)^2 - (\Delta E)^2$, kde $\Delta E \stackrel{def}{=} \frac{1}{n} \cdot \sum_{i=1}^{n} \Delta x_i$, a je omezen zdola 0 a shora $\Delta^{(2)} \stackrel{def}{=} \frac{1}{n} \cdot \sum_{i=1}^{n} \Delta_i^2$. Pro rozptyl **V** dostáváme interval $[V_0 - \Delta, V_0 + \Delta + \Delta^{(2)}]$.

Někdy odhad pomocí linearizace není vhodný, jelikož intervaly mohou být příliš široké, takže kvadratický člen nemůže být vynechán nebo může nastat situ-

ace, kdy budeme chtít zaručit, že rozptyl nepřesáhne určitou požadovanou mez. V takových případech potřebujeme přesný interval.

Problém nalezení přesného intervalu pro takovou jednoduchou statistickou charakteristiku jako je rozptyl má NP složitost. Neexistuje časově realizovatelný algoritmus , který by vždycky vypočítal přesný interval této charakteristiky. Musíme proto najít případy, kdy vyřešení tohoto problému bude možné pomocí algoritmů.

4.4 Tradiční intervalové metody

Tradiční intervalové metody někdy vedou k nadměrné složitosti.

4.4.1 Přímý výpočet

Historický první metodou pro výpočet přiblížení intervalu je přímá metoda. Tato metoda je založena na faktu, že každý algoritmus se skládá z elementárních operaci (aritmetické operace, min, max atd.). Pokud známe intervaly $\mathbf{a}$ a $\mathbf{b}$ pro a a b, potom můžeme najít přesný interval $f(\mathbf{a}, \mathbf{b})$ pro každou elementární operaci $f(a, b)$. V průběhu přímé metody provádíme opakující se výpočty krok za krokem, dosazují se skutečná čísla a potom se na ně aplikuje určitá operace z intervalové aritmetiky. Jako výsledek dostaneme přiblížení pro požadovaný interval.

Při výpočtu intervalu pro průměr konečné populace přímá metoda vede k přesným mezím.

Při výpočtu intervalu pro rozptyl konečné populace je situace komplikovanější. Ve vzorci pro výpočet rozptylu každá proměnná x_i se objevuje několikrát: jednak v $(x_i - E)^2$ a potom ve vyjádření E. V tomto případě přímá metoda vede k nadměrné složitosti.

Příklad: $\mathbf{x}_1 = \mathbf{x}_2 = [0, 1]$, skutečný rozptyl $V = (x_1 - x_2)^2/4$, a proto interval pro rozptyl $\mathbf{V} = [0, 0.25]$. Dále střední hodnota $\mathbf{E} = [0, 1]$ a odtud

$$\frac{(\mathbf{x}_1 - \mathbf{E})^2 + (\mathbf{x}_2 - \mathbf{E})^2}{2} = [0, 1] \supset [0, 0.25].$$

Existuje další vzorec, který můžeme použít pro výpočet rozptylu konečné populace

$$V = \frac{1}{n}\sum_{i=1}^{n} {x_i}^2 - E^2$$

V tomto vyjádření pro rozptyl se každá z proměnných x_i objevuje víckrát, proto jako výsledek dostáváme příliš široký interval: pro $\mathbf{x}_1 = \mathbf{x}_2 = [0,1]$ máme $\mathbf{E} = [0,1]$ a

$$\frac{\mathbf{x}_1^2+\mathbf{x}_2^2}{2} - \mathbf{E}^2 = [-1,1] \supset [0,0.25].$$

Výpočet přesných mezí pro rozptyl konečné populace má NP složitost, z čehož můžeme usoudit, že neexistuje žádný vhodný vzorec pro rozptyl konečné populace pro intervalová data.

4.4.2 Metoda centra

Lepšího výsledku dosáhneme použitím metody centra, kde interval hladké funkce $f(x_1, ..., x_n)$ na krychli $\mathbf{x}_1 \times ... \times \mathbf{x}_n$ je odhadnut jako

$$f(x_1, ..., x_n) \subseteq f(\widetilde{x}_1, ..., \widetilde{x}_n) + \sum_{i=1}^{n} \frac{\partial f}{\partial x_i}(x_1, ..., x_n) \cdot [-\Delta_i, \Delta_i],$$

kde $\widetilde{x}_i = (\underline{x}_i + \overline{x}_i)/2$ je střední bod intervalu a $\Delta_i = (\overline{x}_i - \underline{x}_i)/2$ je poloviční délka intervalu.

Když jsou všechny intervaly stejné, tj. když $\mathbf{x}_i = [0,1]$, potom metoda centra nevede k požadovanému intervalu. Jako výsledek nám tato metoda vždy poskytne interval centrovaný v bodě $f(\widetilde{x}_1, ..., \widetilde{x}_n)$. V tomto případě jsou všechny střední body stejné (rovné 0,5), proto, na těchto bodech, výběrový rozptyl populace je roven 0. Po použití metody centra dostaneme interval centrovaný kolem bodu 0, to znamená, že dolní konec intervalu je záporný, ale protože V je vždy nezáporné, můžeme všechny záporné hodnoty V vyloučit. Horní konec intervalu, který jsme obdrželi díky metodě centra, je také odlišný od horního konce skutečného intervalu. Pro $\mathbf{x}_1 = \mathbf{x}_2 = [0,1]$ máme $\frac{\partial f}{\partial x_1}(x_1, x_2) = (x_1 - x_2)/2$, takže

$$\frac{\partial f}{\partial x_1}(\mathbf{x}_1, \mathbf{x}_2) = \frac{\mathbf{x}_1 - \mathbf{x}_2}{2} = [-0.5, 0.5].$$

Stejný vzorec máme i po derivaci podle x_2 a pro $\Delta_i = 0.5$. Potom výsledek metody centra je

$$f(\mathbf{x}_1, \ldots, \mathbf{x}_n) \subseteq 0 + [-0.5, 0.5] \cdot [-0.5, 0.5] + [-0.5, 0.5] \cdot [-0.5, 0.5] = [-0.5, 0.5],$$

což je příliš široký interval v porovnání se skutečným intervalem [0,0.25].

4.4.3 Techniky založené na omezení předpokladů

Metody, které jsme uvedli výše, můžeme úspěšně upravit tím, že omezíme některé z předpokladů. V speciálních případech známe jak omezení na intervaly možných vstupních hodnot tak i omezení pro požadované hodnoty charakteristik. Například víme, že hodnota rozptylu $V = V(x_1, \ldots, x_n)$ je vždycky nezáporná: $V \geq 0$ neboli $V \in [0, \infty)$.

V našem numerickém příkladu jsme pomocí přímé metody zjistili, že $V \in [-1, 1]$, a dále víme, že $V \in [0, \infty)$, takže můžeme udělat závěr, že V patří do průniku těchto dvou intervalu, $[-1, 1] \cap [0, \infty) = [0, 1]$. Výsledný interval je dvakrát užší než interval $[-1, 1]$, který jsme obdrželi bez použití omezení. Výsledkem metody centra je $V \in [-0.5, 0.5]$, dále taky víme, že $V \in [0, \infty]$, potom V patří do průniku těchto dvou intervalu, $[-0.5, 0.5] \cap [0, \infty) = [0, 0.5]$. Tento interval je dvakrát užší než interval $[-0.5, 0.5]$, který jsme obdrželi bez použití omezení. V obou případech máme pořad příliš široké intervaly v porovnání se skutečným intervalem $[0, 0.25]$.

4.5 Tradiční optimalizační metody

Jednou z možnosti, abychom vyřešili problém výpočtu přesného intervalu $[\underline{V}, \overline{V}]$ pro rozptyl konečné populace, je vyřešit to jako omezený optimalizační problém. Konkrétně, abychom našli $\underline{V}$, musíme najít minimum funkce za předpokladů $\underline{x}_1 \leq x_1 \leq \overline{x}_1, \ldots, \underline{x}_n \leq x_n \leq \overline{x}_n$, a abychom našli $\overline{V}$, musíme najít maximum funkce za stejných předpokladů.

Existují optimalizační metody, které umožňují vypočítat přesné hodnoty $min(f(x))$ a $max(f(x))$, ale chování takových omezených optimalizačních algoritmů není lehce předvídatelné a časová složitost obecně může být $O(2^n)$. Zatímco pro malá n tyto algoritmy jsou použitelné, pro velká n (exponenciální) doba výpočtu roste tak rychle, že je algoritmus neuskutečnitelný.

Současné metody nejsou vždycky efektivní a neposkytují nám přesné odhady pro $\underline{V}$ a $\overline{V}$, a proto potřebujeme metody nové. V dalších kapitolách ukážeme některé nové postupy pro výpočet rozptylu a analyzujeme problém výpočtu dalších populačních parametrů pro intervalová data.

Kapitola 5

Typy problémů a intervalů

Výpočet statistik za předpokladu intervalové neurčitosti je často NP složitý. Nemůžeme proto očekávat, že najdeme proveditelný algoritmus, který by nám vyřešil všechny možné případy tohoto problému. Musíme proto specifikovat, pro jaké praktické druhy problému jsme schopni vytvořit uskutečnitelné algoritmy. V této kapitole popíšeme různé druhy intervalů, se kterými se můžeme setkat, a praktické problémy, pro které efektivní algoritmy existují.

5.1 Úzké intervaly

Nejvíce používanou technikou ve vědních a inženýrských oborech je linearizace. Hlavní idea linearizace spočívá v tom, že chyby měření Δx_i jsou malé, takže v rozvoji Δx_i můžeme bezpečně vynechat členy s druhou a vyšší mocninou, čímž problém poměrně náročný na analýzu nahradíme jeho jednodušší lineární aproximací. Přesnost této metody je určena velikostí prvního vynechaného členu rozvoje. Proto čím menší hodnoty Δx_i máme, neboli čím užší intervaly máme, tím přesnější výsledek nám poskytne linearizace.

Ve skutečností je naším cílem určit požadovaný interval s určitou přesností. Když máme k dispozici dostatečně přesné intervaly, potom odhad pomocí linearizační metody nám poskytne výsledek s požadovanou přesností, a můžeme říct, že máme efektivní algoritmus pro řešení tohoto problému. Když jsou intervaly širší, potom už nemůžeme ignorovat členy s druhou a vyšší mocninou, takže dostáváme složitější problém. Obecně můžeme říci, že čím užší jsou intervaly, tím

lehčí je problém. Proto případ s úzkými intervaly je prvním adeptem, pro který můžeme očekávat, že bude existovat proveditelný algoritmus na výpočet statistik intervalových dat.

Okamžitě nastává přirozená otázka jak můžeme matematický popsat úzké intervaly. Protože provádíme statistickou analýzu, tak to znamená, že můžeme očekávat, že aktuální hodnoty $x_1, ..., x_n$ pochází z rozložení pravděpodobnosti a naším cílem je najít statistické charakteristiky tohoto pravděpodobnostního rozložení. V našem případě obvykle budeme pracovat se spojitým rozdělením: rovnoměrné, normální, exponenciální atd. Pro každé reálné číslo a $\rho = \int_{a-\delta}^{a+\delta} \rho(x)dx$ udává pravděpodobnost, že náhodná veličina je z intervalu $[a - \delta, a + \delta]$, tato pravděpodobnost je přibližně rovna $\rho(a) \cdot 2\delta$, což konverguje k 0 pokud $\delta \to 0$. To znamená, že pro každou hodnotu a pravděpodobnost, že náhodná veličina bude přesně a , je rovná 0. Všechny hodnoty $x_1, ..., x_n$, náhodně vybrané z původního rozdělení, s pravděpodobnosti rovnou 1, jsou různé.

Intervalová data $\mathbf{x}_1, ..., \mathbf{x}_n$ obsahují tyto různé hodnoty $x_1, ..., x_n$. Když intervaly $\mathbf{x}_i$, obklopující odpovídající body x_i, jsou úzké, potom tyto intervaly se neprotínají. Intervaly se začnou protínat, když jejích šířka je větší než vzdálenost mezi skutečnými hodnotami.

Ideální případ, kdy intervaly můžeme popsat jako úzké je ten, když žádné dva libovolné intervaly $\mathbf{x}_i$ se neprotínají.

5.2 Tak zvané „širší intervaly“

V odstavci 5.1 jsme viděli, že úzké intervaly můžeme popsat jako intervaly, které se neprotínají. Mějme množinu nějakých hodnot $x_1 < x_2 < ... < x_n$ a mějme takové intervaly kolem každé hodnoty, které jsou tak úzké, že sousední intervaly $\mathbf{x}_i$ a $\mathbf{x}_{i+1}$ se neprotínají.

Když šířka intervalů začne zvětšovat, začnou se intervaly protínat. Nejdřív se začnou protínat jenom sousední intervaly $\mathbf{x}_i$ a $\mathbf{x}_{i+1}$, ale intervaly $\mathbf{x}_i$ a $\mathbf{x}_{i+2}$ se stále neprotnou. Když šířka poroste dále, tak se začnou protínat i intervaly $\mathbf{x}_i$ a $\mathbf{x}_{i+2}$, atd. Pokud jsou intervaly příliš široké, potom se všechny intervaly protnou.

Toto umožňuje určit stupeň úzkosti (šíře) intervalů. Tento stupeň bude odpovídat počtu intervalů, které mají společný bod.

Definujeme případ tak zvaně „širších intervalů“ jako situaci, kdy pro nějaké celé číslo K žádná množina K intervalů nemá společný průnik. Případu úzkých intervalů odpovídá $K = 2$, dalším případem je $K = 3$ atd., až do obecného případu, kdy $K = n$.

Jak jsme uvedli výše, čím užší máme intervaly, tím lehčí máme výpočetní problém. Pokud budeme uvažovat, že K je mírou šíře intervalů, potom můžeme očekávat, že pro nepříliš velké hodnoty K bude existovat efektivní algoritmus pro vyřešení tohoto případu.

5.3 Jeden měřící přístroj

Ve čtvrté kapitole jsme se zmínili, že nejvíce používanou metodou spojenou s případem, kdy se musíme vypořádat s nepřesným měřením, je linearizace. Aby bylo jednoduché vypočítat interval $\mathbf{C}$ statistiky C pomocí metody linearizace, musíme se ujistit, že intervaly jsou nejenom poměrně úzké, ale také, že jsou přibližně stejné velikosti. Neboli, pokud je Δx_i^2 stejného řádu jako Δx_j, potom není dobré vynechat Δx_i^2 a ponechat pouze Δx_j. Množina intervalových dat by se neměla skládat jak z vysoce přesných měřených hodnot s úzkými intervaly, tak i z málo přesných měřených hodnot s širokými intervaly. Všechna měření by měla byt prováděna jedním měřícím přístrojem nebo alespoň měřicími přistrojí stejného typu, měřícího s touž přesností.

Tradiční linearizační metody nám poskytují jenom přibližný interval. Jak ale bude ukázáno dále, pro některé případy tyto přibližné odhady mohou byt efektivním výpočtem pro určení přesného intervalu. Protože taková možnost tady existuje, popíšeme pro jaké speciální případy je linearizace možná, tj. takové případy, kde všechna měření jsou prováděná jenom jedním přístrojem.

Jak můžeme tento druh případů popsat matematicky? To, že jeden interval je podmnožinou druhého intervalu, tj. $[\underline{x}_i, \overline{x}_i] \subseteq (\underline{x}_j, \overline{x}_j)$, nám naznačuje, že můžeme mít dva různé měřicí přístroje odlišné kvality.

Toto omezení se vztahuje pouze pro nedegenerované intervaly. Dále u takových intervalů můžeme mít počítačem změřené hodnoty s plovoucí desetinnou čárkou, které jsou tak přesné jak to jenom dokážeme, potom tyto hodnoty mů-

žeme považovat za přesně známe. Odsud, když mluvíme o měřeních provedených jedním přístrojem, můžeme také brát v úvahu i degenerované intervaly, tj. přesná čísla.

Absence takových dvojic je užitečný předpoklad, který nám umožňuje rychlejší výpočet intervalové statistiky. Dále je zřejmé, že absence takových intervalů, se může vyskytnout nejenom v případech s měřením jedním přístrojem, ale i v jiných praktických případech. Protože tato podmínka je důležitá, uvedeme následující definici.

Definice 1 Řekneme, že soubor intervalů splňuje podmínku o podmnožinách, pokud $[\underline{x}_i, \overline{x}_i] \not\subseteq (\underline{x}_j, \overline{x}_j)$ pro každé i a j takové, že intervaly $\mathbf{x}_i$ a $\mathbf{x}_j$ jsou nedegenerované.

5.4 Více měřících přístrojů

Po případu s jedním měřícím přístrojem logicky následuje případ, kdy máme několik (m) měřících přístrojů. Naše intervaly jsou rozděleny do několika podskupin, z nichž každá splňuje shora uvedenou podmínku o podmnožinách.

Jak jsme se již zmínili, případ s jedním měřicím přístrojem je nejjednodušší. Čím více měřících přístrojů vstupuje do zadání, tím složitější je výsledný problém. Pokud $m = n$, potom dostáváme obecný případ, kdy každé měření je získáno jiným měřícím přístrojem.

Protože parametr m je měřítkem složitosti, můžeme očekávat, že existuje efektivní algoritmus pro nějaký případ s pevným číslem m nebo alespoň pro hodnoty m, které nejsou nepříliš velké.

5.5 Případ soukromí

Na základě předchozích argumentů můžeme tvrdit, že neurčitost, způsobená měřením, vede přirozeně k intervalům. Měření přitom není jediným zdrojem intervalové neurčitosti, která může pocházet také z jiných důvodů. Jedním z takových příkladů je záměr uchovat osobní data v statistickým databázích. Dost často se stává, že potřebujeme shromáždit velké množství dat o lidech, například při sčítání lidu nebo během lékařských pokusů. Statistická analýza těchto dat nám

umožňuje určit potřebnou korelaci mezi věkem člověka a účinkem určitého testovaného léku nebo mezi zeměpisnou polohou a úrovní příjmů. Je potřeba vytvořit nástroje, které by umožnili statistickou analýzu těchto charakteristik, které jsou velmi užitečné. Pokud dostaneme libovolnou otázku ohledně statistické analýzy takových dat a výsledky této analýzy zveřejníme, potom dané výsledky mohou prozradit soukromé informace z databází, které nesmí být zveřejněné, neboť by byl porušen zákon o ochraně osobních dat.

Jednou z možnosti jak zachovat soukromá data je, že místo toho, abychom si zaznamenávali do databáze přesné údaje o jednotlivých osobách, jako je výše platu nebo věk, budeme zaznamenávat pevně stanovené mezní hodnoty, tj. 0, 10, 20, 30 let. Každého člověka zařadíme do odpovídajícího věkového intervalu: od 0 do 10, od 10 do 20, od 20 do 30 atd. V okamžiku, kdy v databází nejsou žádné přesné osobní údaje, odpadá i riziko zveřejnění soukromých dat.

Tato myšlenka řeší případ ochrany soukromých dat, ale tím se otevírá jiný problém: jakým způsobem budeme provádět statistickou analýzu takto ošetřených dat? Zajímají nás hodnoty statistických charakteristik $C(x_1, ..., x_n)$. Pokud známe skutečné hodnoty $x_1, ..., x_n$, potom můžeme lehce určit hodnoty těchto charakteristik. V našem případě s neurčitými intervaly, které vznikly z požadavku ochrany osobních dat, všechno, co víme, jsou intervaly $\mathbf{x}_i = [\underline{x}_i, \overline{x}_i]$ možných hodnot x_i. Obecně, různé hodnoty $x_i \in \mathbf{x}_i$ vedou k různým hodnotám $C(x_1, ..., x_n)$. Dobrým nápadem je získat interval možných hodnot charakteristiky $C(x_1, ..., x_n)$, kde $x_i \in \mathbf{x}_i$.

Z hlediska algoritmizace máme stejný problém jako jsme měli v případu neurčitosti intervalu z důvodu měření. Musíme najít požadovanou charakteristiku $C(x_1, ..., x_n)$ pro dané intervaly $\mathbf{x}_1, ..., \mathbf{x}_n$. Rozdíl mezi tímto a předchozími dvěma případy je, že v předešlých dvou případech jsme neznali přesné hodnoty, zatímco zde je možné tyto přesné hodnoty získat, ale nepoužijeme je, protože chceme zachovat soukromí.

Z matematického hlediska intervaly, vycházející z osobních dat, mají následující vlastnost: buď se shodují (pokud hodnoty dvou pacientů spadají do stejného intervalu) nebo se liší, v tomto případě se mohou protínat v mezním bodě. Dále můžeme použit přesné udaje, které byly v minulosti zaznamenané a zveřejněné,

a to tak, že těmto přesným hodnotám přiřadíme odpovídající intervaly.

Intervalová data, která mají vlastnost, že každé dva nedegenerované intervaly se shodují nebo protínají v nejvýše jednom bodě, nazveme soukromá.

Pro tento případ je podmínka o podmnožinách splněna, takže algoritmy, které vyhovují pro podmínku o podmnožinách, vyhovují i pro případ soukromí.

Občas se stává, že v situacích s ochranou osobních dat musíme nakládat s daty, která pochází z různých zdrojů. Abychom zjistili průměrný plat v České Republice a na Slovensku, musíme nakombinovat české údaje o mzdách, kde plat je od 0 do 10 000 Kč, od 10 000 do 20 000 Kč atd se slovenskými údaji, kde intervaly jsou od 0 do 500 Euro, od 500 do 1 000 Euro atd. Zde je potřeba převést tyto data do společné měny, potom dostaneme odlišné skupiny intervalů, které vyhovují podmínce o podmnožinách. K vyřešení tohoto problému, můžeme použit algoritmus vytvořený pro případ s několika měřícími přístroji.

5.6 Nedetekovatelné hodnoty

Důležitým praktickým případem je, že nejsme schopní určit hodnoty. Většina senzorů je velice přesná, ale mají svůj detekční limit DL, takže nemohou zaznamenat hodnoty pod DL, ale jsou schopny určit hodnoty rovné DL a vyšší s velmi vysokou přesností.

V tomto případě, pokud senzor ukáže hodnotu $\widetilde{x} \geq DL$, potom tato hodnota může být považována za velmi přesnou, takže ji budeme brát jako přesnou hodnotu, tj. máme degenerovaný interval $[\widetilde{x}, \widetilde{x}]$. Pokud ovšem senzor neudá vůbec žádnou hodnotu a výsledek měření je $\widetilde{x} = 0$, potom jediný závěr o měřené veličině, který můžeme udělat, je ten, že tato hodnota leží pod detekčním limitem senzoru a spadá do intervalu $[0, DL)$.

Každý interval je tedy buď přesná hodnota nebo nedetekovatelný interval $[0, DL_i)$ pro nějaké reálné číslo DL_i, které může být jiné pro různé senzory. Můžeme udělat závěr, že výsledné nedegenerované intervaly taky splňují podmínku o podmnožinách. Algoritmy, které fungují pro data splňující podmínku o podmnožinách, budou fungovat i pro nedetekovatelná data.

Algoritmy, které fungují pro soukromá data, budou také fungovat i pro nedetekovatelná data, pokud všechny senzory budou mít stejný detekční limit DL.

Kapitola 6

Výpočet statistik při intervalové neurčitosti

V této kapitole popíšeme efektivní algoritmy a jejich úpravy pro výpočet dolních a horních intervalových mezí pro vybrané statistické charakteristiky.

6.1 Dolní mez pro rozptyl

6.1.1 Algoritmus se složitosti $O(n^2)$

Jako první uvedeme algoritmus, který počítá $\underline{V}$ v kvadratickém čase neboli postup, kde je potřeba $O(n^2)$ výpočetních kroků pro n intervalových dat $\mathbf{x}_i = [\underline{x}_i, \overline{x}_i]$. Tento algoritmus můžeme najít v [6],[7],[8],[9],[10].

Algoritmus postupuje následujícím způsobem:

1. Uspořádáme všech $2n$ hodnot $\underline{x}_i, \overline{x}_i$ do posloupnosti $x_{(1)} \leq x_{(2)} \leq ... \leq x_{(2n)}$.
2. Vypočítáme $\underline{E}$ a $\overline{E}$ a vybereme všechny „malé intervaly“ $[x_{(k)}, x_{(k+1)}]$, které se protínají s intervalem $[\underline{E}, \overline{E}]$.
3. Pro každý z vybraných „malých intervalů“ $[x_{(k)}, x_{(k+1)}]$ vypočítáme poměr $E_k = S_k/N_k$, kde

$$S_k \stackrel{def}{=} \sum_{i:\underline{x}_i \geq x_{(k+1)}} \underline{x}_i + \sum_{j:\overline{x}_j \leq x_{(k)}} \overline{x}_j$$

a N_k je celkový počet takových i a j, pro které platí $\underline{x}_i \geq x_{(k+1)}$ a $\overline{x}_j \leq x_{(k)}$. Pokud $E_k \in [x_{(k)}, x_{(k+1)}]$, potom počítáme

$$V_k \stackrel{def}{=} \frac{1}{n} \cdot \left(\sum_{i:\underline{x}_i \geq x_{(k+1)}} (\underline{x}_i - E_k)^2 + \sum_{j:\overline{x}_j \leq x_{(k)}} (\overline{x}_j - E_k)^2 \right).$$

Pokud $N_k = 0$ potom $V_k \stackrel{def}{=} 0$.

4. Nejmenší hodnotu V_k položíme rovnou $\underline{V}$.

Zkontrolujeme, zda daný algoritmus opravdu potřebuje $O(n^2)$ kroků. K třídění je potřeba $O(n \cdot log(n))$ kroků. Výpočet E_k a V_k trvá $O(n)$ kroků pro každé k, ale výpočet všech možných E_k a V_k trvá $O(n^2)$ kroků. Nalezení nejmenšího V_k nám zabere $O(n)$ kroků. $\underline{V}$ tedy můžeme vypočítat pomocí $O(n^2)$ kroků.

Hlavní myšlenka tohoto algoritmu

Algoritmus pro výpočet $\underline{V}$ je založen na faktu, že když funkce V dosáhne svého minima na intervalu $[\underline{x}_i, \overline{x}_i]$, tak potom buď $\frac{\partial V}{\partial x_i} = 0$ nebo minimum je dosaženo v levém koncovém bodě intervalu $x_i = \underline{x}_i$, potom $\frac{\partial V}{\partial x_i} > 0$, nebo minimum je dosaženo v pravém koncovém bodě intervalu $x_i = \overline{x_i}$ a $\frac{\partial V}{\partial x_i} < 0$. Protože parciální derivace je rovna $(2/n) \cdot (x_i - E)$, potom máme buď $x_i = E$, $x_i = \underline{x}_i > E$ nebo $x_i = \overline{x}_i < E$. Pokud víme, kde se nachází E ve vztahu ke všem koncovým bodům, potom můžeme určit odpovídající minimalizující hodnotu x_i pro každé i: pokud $\overline{x}_i \leq E$, potom $x_i = \overline{x}_i$; pokud $E \leq \underline{x}_i$, potom $x_i = \underline{x}_i$; jinak $x_i = E$. Odpovídající hodnotu E určíme z podmínky, že E je průměr všech vybraných hodnot x_i.

Abychom našli nejmenší hodnotu V, roztřídíme všech $2n$ koncových hodnot intervalů $\underline{x}_i, \overline{x}_i$ do řady $x_{(1)} \leq x_{(2)} \leq ... \leq x_{(2n)}$, potom pro každou oblast $[x_{(k)}, x_{(k+1)}]$, vypočítáme odpovídající hodnoty x_i, najdeme jejích rozptyl a následně určíme nejmenší z těchto rozptylů V_k.

Jak jsem se jíž zmínil, odpovídající hodnota E může být nalezena z podmínky, že E je průměr všech vybraných hodnot x_i. Pokud E je uvnitř $[x_{(k)}, x_{(k+1)}]$, potom známe všechny hodnoty x_i. Označme hodnotu E jako E_k pokud je uvnitř $[x_{(k)}, x_{(k+1)}]$, takže $n \cdot E$ by se mělo rovnat součtu těchto hodnot:

$$n \cdot E_k = \sum_{i:\underline{x}_i \geq x_{(k+1)}} \underline{x}_i + (n - N_k) \cdot E_k + \sum_{j:\overline{x}_j \leq x_{(k)}} \overline{x}_j,$$

kde N_k označíme jako celkový počet i takových, pro které $\underline{x}_i \geq x_{(k+1)}$, a j takových, pro které $\overline{x}_j \leq x_{(k)}$.

Odečtením od obou stran rovnice $(n - N_k) \cdot E$ dostaneme výraz $N_k \cdot E_k = S_k$, kde

$$S_k \stackrel{def}{=} \sum_{i:\underline{x}_i \geq x_{(k+1)}} \underline{x}_i + \sum_{j:\overline{x}_j \leq x_{(k)}} \overline{x}_j.$$

Jestliže $N_k = 0$, tak to známena, že $x_i = E_k$ pro všechny i, takže $V = 0$. Jestliže $N_k \neq 0$, potom $E_k = S_k/N_k$.

Když je E_k vypočítané, zkontrolujeme, zda $E_k \in [x_{(k)}, x_{(k+1)}]$. Pokud tato podmínka je splněna, jsme schopni vyčíslit odpovídající hodnotu V_k pro všechny zvolené hodnoty x_i

6.1.2 Algoritmus se složitosti $O(n \cdot log(n))$

Nejvíce časově náročný krok v předchozím algoritmu s kvadratickou časovou složitosti byl krok, kde se počítaly všechny možné hodnoty E_k a V_k. Přitom to byl pouze tento samotný krok, který měl kvadratickou časovou náročnost. Následující postup vylepšuje předchozí algoritmus a snižuje asymptotickou složitost na $O(n \cdot log(n))$ (viz [5],[11]).

Algoritmus postupuje následujícím způsobem:

1. Uspořádáme všech $2n$ hodnot $\underline{x}_i, \overline{x}_i$ do posloupnosti $x_{(1)} \leq x_{(2)} \leq ... \leq x_{(2n)}$.

2. Použijeme metodu půlení, abychom nalezli hodnotu k $(1 \leq k \leq 2n)$, pro kterou platí následující dvě nerovnosti:

$$\sum_{j:\overline{x}_j \leq x_{(k)}} (x_{(k)} - \overline{x}_j) \leq \sum_{i:\underline{x}_i \geq x_{(k+1)}} (\underline{x}_i - x_{(k)}); \tag{6.1}$$

$$\sum_{j:\overline{x}_j \leq x_{(k)}} (x_{(k+1)} - \overline{x}_j) \geq \sum_{i:\underline{x}_i \geq x_{(k+1)}} (\underline{x}_i - x_{(k+1)}). \tag{6.2}$$

   Při každé iteraci půlení máme interval $[k^-, k^+]$, což je garantovaný interval, který obsahuje správnou hodnotu k. Na začátku $k^- = 1$ a $k^+ = 2n$. V dalším kroku vypočítáme prostřední bod $k_{str} = \lfloor (k^- + k^+)/2 \rfloor$ a zkontrolujeme, zda obě nerovnosti (6.1) a (6.2) platí pro $k = k_{str}$. Potom:

   - Pokud obě nerovnosti (6.1) a (6.2) platí pro toto k, potom jsme našli požadované k.

- Pokud (6.1) platí, ale (6.2) neplatí, tak potom to známena, že požadovaná hodnota k je větší než k_{str}, takže k^+ ponecháme a k^- vyměníme s $k_{str}+1$.
- Pokud (6.2) platí, ale (6.1) neplatí, tak potom to známena, že požadovaná hodnota k je menší než k_{str}, takže k^- ponecháme a k^+ vyměníme s $k_{str}-1$.

3. Jakmile je k nalezeno, vypočítáme

$$S_k \stackrel{def}{=} \sum_{i:\underline{x}_i \geq x_{(k+1)}} \underline{x}_i + \sum_{j:\overline{x}_j \leq x_{(k)}} \overline{x}_j,$$

potom $r_k = S_k/N_k$, kde N_k je celkový počet takových i a j, pro které platí $\underline{x}_i \geq x_{(k+1)}$ a $\overline{x}_j \leq x_{(k)}$. Nakonec počítáme

$$\underline{V} = \frac{1}{n} \cdot \left(\sum_{i:\underline{x}_i \geq x_{(k+1)}} (\underline{x}_i - r_k)^2 + \sum_{j:\overline{x}_j \leq x_{(k)}} (\overline{x}_j - r_k)^2 \right).$$

Ukažme, jak najdeme požadované k rychleji než v předchozím algoritmu. Chceme, aby platily podmínky $x_{(k)} \leq E$ a $E \leq x_{(k+1)}$. Hodnota E je průměr všech vybraných hodnot x_i, máme

$$n \cdot E = S_k + (n - N_k) \cdot E. \tag{6.3}$$

Odečteme $(n - N_k) \cdot E$ od obou stran (6.3) a dostaneme $N_k \cdot E = S_k$, tj. že $E = S_k/N_k$. První požadovaná nerovnost $x_{(k)} \leq E$ bude mít tvar $S_k/N_k \leq x_{(k)}$, tj.

$$N_k \cdot x_{(k)} \leq \sum_{i:\underline{x}_i \geq x_{(k+1)}} \underline{x}_i + \sum_{j:\overline{x}_j \leq x_{(k)}} \overline{x}_j. \tag{6.4}$$

Na pravé straně (6.4) máme tolik členu $x_{(k)}$ kolik je členu na levé straně. Odsud odečteme všech $N_k \cdot x_{(k)}$ členů od pravé strany nerovnosti (6.4). Když $\overline{x}_j \leq x_{(k)}$, potom rozdíl $\overline{x}_j - x_{(k)}$ je záporný, takže ho můžeme přesunout na levou stranu nerovnosti, čímž dostaneme nerovnost (6.1).

Když k roste, potom roste i levá strana nerovnosti (6.1) a pravá strana se zmenšuje s k. Pokud tato nerovnost platí pro k, potom platí i pro menší hodnoty, tj. pro $k-1, k-2$ atd.

Stejným způsobem u druhé požadované nerovnosti $E \leq x_{(k+1)}$ dostaneme ekvivalentní nerovnost (6.2). Když k roste, potom levá strana teto nerovnosti roste, kdežto pravá strana klesá. Pokud nerovnost platí pro k, potom musí platit i pro $k+1, k+2$ atd.

Pokud obě nerovnosti (6.1) a (6.2) platí pro dvě různé hodnoty $k < k'$, potom by obě měly být platné i pro všechny hodnoty mezi k a k', tj. pro $k+1, k+2, ..., k'-1$. Ukažme, že obě nerovnosti nemůžou být platné pro k a $k+1$. Pokud nerovnost (6.1) platí pro $k+1$, tak to znamená, že

$$\sum_{j:\overline{x}_j \leq x_{(k+1)}} (x_{(k+1)} - \overline{x}_j) \leq \sum_{i:\underline{x}_i \geq x_{(k+2)}} (\underline{x}_i - x_{(k+1)}). \tag{6.5}$$

Levá strana teto nerovnosti není menší než levá strana (6.2) a pravá strana teto nerovnosti není větší než pravá strana (6.2). Proto (6.5) je v rozporu s (6.2). Tento rozpor ukazuje, že může být jenom jedno k pro které obě nerovností platí, a toto k můžeme najit pomoci metody půlení, jak jsme popsali v algoritmu.

Jak dlouho potrvá tento algoritmus? Na začátku naši jedinou informaci je, že k patří do intervalu $[1, 2n]$ šířky $O(n)$. V každém kroku půlení dělíme interval (obsahující k) napůl. Po provedení I iteraci zmenšíme šířku intervalu 2^I krát. Proto, abychom našli přesnou hodnotu k, musíme provést I iteraci, pro které platí $O(n)/2^I = 1$, tj. potřebujeme $I = O(log(n))$ iteraci. Na každou iteraci potřebujeme $O(n)$ kroků, takže celkově je zapotřebí $O(n \cdot log(n))$ kroků. Třídění potrvá $O(n \cdot log(n))$ kroků a výpočet rozptylu $O(n)$. Takže celkově máme algoritmus o složitosti $O(n \cdot log(n))$.

6.1.3 Algoritmus se složitosti $O(n)$

V [15] je uveden algoritmus se složitosti $O(n)$, ale tento algoritmus nevypočte $\underline{V}$ správně, proto jsme tento postup opravili. Následující algoritmus je založen na iteracích. Při každé iteraci máme 3 množiny:

- množina I^- bodů, pro které již víme, že pro optimální vektor x máme $x_j = \overline{x}_j$;
- množina I^+ bodů, pro které již víme , že pro optimální vektor x máme $x_i = \underline{x}_i$;

- množina I koncových bodů $\underline{x}_i$ a $\overline{x}_j$, pro které jsme se ještě nerozhodli, zdá patří do optimálního vektoru x.

Na začátku položíme $I^- = I^+ = \emptyset$ a I bude množina všech $2n$ koncových bodů. Při každé iteraci budeme obnovovat následující parametry:

- N^- - počet prvků množiny I^-,
- N^+ - počet prvků množiny I^+,
- $S^- = \sum_{\overline{x}_j \in I^-} \overline{x}_j$,
- $S^+ = \sum_{\underline{x}_i \in I^+} \underline{x}_i$.

Na počátku položíme $N^- = N^+ = S^- = S^+ = 0$.

Při každé iteraci postupujeme následovně:

1. Vypočítáme medián m množiny I;
2. Množinu I analyzujeme prvek po prvku tak, že tyto prvky rozdělíme do dvou podmnožin

$$P^- = \{x \in I : x \leq m\};\ P^+ = \{x \in I : x > m\};$$

   dále také určíme $m^+ = \min\{x : x \in P^+\}$;
3. Vypočítáme
   - $e^- = S^- + \sum_{\overline{x}_j \in P^-} \overline{x}_j$,
   - $e^+ = S^+ + \sum_{\underline{x}_i \in P^+} \underline{x}_i$,
   - $n^- = N^- + \#\{\overline{x}_j \in P^-\}$,
   - $n^+ = N^+ + \#\{\underline{x}_j \in P^+\}$,
   - $E = \frac{e^- + e^+}{n^- + n^+}$;
4. Pokud $E < m$, potom vyměníme I^+ za $I^+ \cup P^+$, S^+ za e^+, I za P^- a N^+ za n^+.
5. Pokud $E > m^+$, potom vyměníme I^- za $I^- \cup P^-$, S^- za e^-, I za P^+ a N^- za n^-.

6. Pokud $m \leq E \leq m^+$, potom vyměníme I^- za $I^- \cup P^-$, I^+ za $I^+ \cup P^+$, I za $\emptyset$, S^- za e^-, S^+ za e^+, N^- za n^- a N^+ za n^+.

Při každé iteraci se množina I zmenší na polovinu. Iterace pokračují dokud množina I není prázdná. Potom dostaneme hodnotu $\underline{V}$ rozptylu populace pro vektor x, pro který platí:

- $x_j = \overline{x}_j$ pro indexy j takové, že $\overline{x}_j \in I^-$,
- $x_i = \underline{x}_i$ pro indexy i takové, že $\underline{x}_i \in I^+$,
- $x_i = E$ pro všechny indexy i.

Asymptotická složitost tohoto algoritmu je $O(n)$. Během každé iteraci výpočet střední hodnoty má lineární složitost a všechny ostatní operace s množinou I mají čas výpočtu t lineární podle počtu prvků množiny I. Na začátku má množina I n prvků, při každé další iteraci se velikost této množiny zmenší dvakrát, takže máme postupně množinu o velikosti $n/2$, potom $n/4$ atd., takže celkový čas výpočtu je $0 \leq C \cdot (n + n/2 + n/4 + ...) \leq C \cdot 2n$, z čehož plyne, že algoritmus má lineární časovou složitost.

Dokažme, že výše popsaný algoritmus vždy najde $\underline{V}$

V dvou předchozích algoritmech jsme ukázali, že pokud roztřídíme všech $2n$ koncových bodů intervalů do posloupnosti $x_{(1)} \leq x_{(2)} \leq ... \leq x_{(2n)}$, potom pro nějaké $k = k_{min}$ minimum $\underline{V}$ je dosaženo pro vektor x, pro který platí:

- pro každý index j takový, že $\overline{x}_j \leq x_{(k)}$, platí $x_j = \overline{x}_j$;
- pro každý index i takový, že $\underline{x}_i \geq x_{(k+1)}$, platí $x_i = \underline{x}_i$;
- pro všechny ostatní indexy máme $x_i = E_k \stackrel{def}{=} \frac{S_k}{N_k}$, kde

$$S_k = \sum_{j:\overline{x}_j \leq x_{(k)}} \overline{x}_j + \sum_{i:\underline{x}_i \geq x_{(k+1)}} \underline{x}_i$$

$$N_k = \#\{j : \overline{x}_j \leq x_{(k)}\} + \#\{i : \underline{x}_i \geq x_{(k+1)}\}.$$

Dále pro optimální k máme $E_k \in [x_{(k)}, x_{(k+1)}]$.

Podmínka $x_{(k)} \leq E_k = \frac{S_k}{N_k}$ je stejná jako

$$N_k \cdot x_{(k)} \leq S_k = \sum_{j:\overline{x}_j \leq x_{(k)}} \overline{x}_j + \sum_{i:\underline{x}_i \geq x_{(k+1)}} \underline{x}_i.$$

Od obou stran této nerovnosti odečteme $N_k \cdot x_{(k)}$ a potom převedeme první sumu, která bude nezáporná, na levou stranu. Tím dostaneme následující nerovnost

$$\sum_{j:\overline{x}_j \leq x_{(k)}} (x_{(k)} - \overline{x}_j) \leq \sum_{i:\underline{x}_i \geq x_{(k+1)}} (\underline{x}_i - x_{(k)}). \tag{6.6}$$

Zvýšením k dosáhneme toho, že se zvětší počet členů na levé straně nerovnosti (6.6) a zmenší na pravé straně této nerovnosti. To znamená, že levá strana se nepravidelně zvětšuje a pravá strana se nepravidelně zmenšuje. Pokud nerovnost (6.6) platí pro nějaké k, potom také musí platit i pro všechny hodnoty menší než k. Tato nerovnost platí pro všechny k do nějaké určité hodnoty k_0.

Podobně podmínka $x_{(k+1)} \geq E_k = \frac{S_k}{N_k}$ je stejná jako podmínka

$$N_k \cdot x_{(k+1)} \geq \sum_{j:\overline{x}_j \leq x_{(k)}} \overline{x}_j + \sum_{i:\underline{x}_i \geq x_{(k+1)}} \underline{x}_i.$$

Od obou stran této nerovnosti odečteme $N_k \cdot x_{(k+1)}$ a potom převedeme první sumu, která bude nezáporná, na levou stranu. Tím dostaneme následující nerovnost

$$\sum_{j:\overline{x}_j \leq x_{(k)}} (x_{(k+1)} - \overline{x}_j) \leq \sum_{i:\underline{x}_i \geq x_{(k+1)}} (\underline{x}_i - x_{(k+1)}). \tag{6.7}$$

Zvýšením k dosáhneme toho, že se zvětší počet členů na levé straně a zmenší počet členů na pravé straně nerovnosti (6.7). Pokud nerovnost (6.7) platí pro nějaké k, potom také musí platit i pro všechny větší hodnoty než k. Tato nerovnost platí pro všechna k od nějaké určité hodnoty l_0.

Obě podmínky (6.6) a (6.7), které jsou ekvivalentní s podmínkou $E_k \in [x_{(k)}, x_{(k+1)}]$, vyhovují buď pro hodnotu k_{min} nebo pro více hodnot z posloupnosti $l_0, l_0 + 1, ..., k_0$. To, že podmínka vyhovuje pro více hodnot z této posloupnosti tedy znamená, že stejné minimum je dosaženo pro všechny hodnoty z této posloupnosti. Stačí ukázat, že jestliže obě podmínky (6.6) a (6.7) platí pro k a $k+1$, tak potom rozptyl V má stejnou hodnotu pro k a $k+1$. Pokud platí (6.6) pro $k+1$, potom

$$\sum_{j:\overline{x}_j \leq x_{(k+1)}} (x_{(k+1)} - \overline{x}_j) \leq \sum_{i:\underline{x}_i \geq x_{(k+2)}} (\underline{x}_i - x_{(k+1)}).$$

Levá strana v této nové nerovnosti je menší nebo rovná než levá strana (6.7) a pravá strana této nové nerovnosti je větší nebo rovná pravé straně nerovností (6.7). Jediná cesta, aby obě nerovnosti platily je, když si obě strany budou rovné. Pokud vyměníme $x_{(k)}$ za $x_{(k+1)}$ a $x_{(k+1)}$ za $x_{(k+2)}$, tak to neovlivní jaké koncové body patří do I^- a jaké do I^+, a proto odpovídající hodnota rozptylu se nezmění.

Takže:

- pro $k < k_{min}$ máme $E_k > x_{(k+1)}$,
- pro $k > k_{min}$ máme $E_k < x_{(k)}$,
- pro $k = k_{min}$ máme $x_{(k)} \leq E_k \leq x_{(k+1)}$.

Odsud:

- jestli $E_k < x_{(k)}$, potom nemůže být $k < k_{min}$ a $k = k_{min}$, a proto $k > k_{min}$;
- jestli $E_k > x_{(k+1)}$, potom nemůže být $k > k_{min}$ a $k = k_{min}$, a proto $k < k_{min}$;
- jestli $x_{(k)} \leq E_k \leq x_{(k+1)}$, potom nemůže být $k < k_{min}$ a $k > k_{min}$, a proto $k = k_{min}$;

Algoritmus najde správnou hodnotu k a tudíž i správnou hodnotu $\underline{V}$.

6.2 Horní mez pro rozptyl

V předchozích kapitolách jsme se zmínili, že výpočet $\overline{V}$ je v obecném případě NP-složitý (viz [6], [7], [8], [9]).

Věta 1. *Výpočet $\overline{V}$ je NP-složitý.*

Důkaz. Podle definice je problém NP-složitý, pokud nějaký problém z třídy NP může být zredukován na NP-složitý problém. Proto, abychom dokázali, že problém P je NP-složitý, je dostatečné zredukovat jeden známý NP-složitý problém P_0 na P.

V tomto případě, pokud je známo, že P_0 je NP-složitý problém, tak to znamená, že každý problém z třídy NP může být zredukován na P_0, a proto P_0 může být zredukován na P. Odsud původní problém z třídy NP je redukovatelný na problém P.

Pro potřeby našeho důkazu jako známý NP-složitý problém P_0 zvolíme problém podmnožiny: pro daných n kladných čísel s_i chceme zjistit, zda existují $\mu_i \in \{+1, -1\}$ takové, pro které je suma $\sum_{i=1}^{n} \mu_i \cdot s_i$ rovna 0.

Ukážeme, že problém podmnožiny může být zredukován na problém výpočtu $\overline{V}$, tj., že každý případ $(s_1, ..., s_n)$ problému P_0 můžeme dát do souvislosti s případem založeným na řešeních výpočetního problému pro $\overline{V}$. Můžeme přitom lehce určit zda požadované μ_i existují.

Uvažujme případ odpovídající intervalům $[\underline{x}_i, \overline{x}_i] = [-s_i, s_i]$. Chceme ukázat, že pro odpovídající problém $\overline{V} = C_0$, kde jsme označili $C_0 \stackrel{def}{=} \frac{1}{n} \cdot \sum_{i=1}^{n} s_i^2$, platí jen tehdy, když existují μ_i pro která $\sum \mu_i \cdot s_i = 0$.

1. Nejprve ukážeme, že pro všechny případy platí $\overline{V} \leq C_0$.

Víme, že pro výpočet rozptylu pro konečnou populaci můžeme využít následující vzorec:

$$V = \frac{1}{n} \cdot \sum_{i=1}^{n} x_i^2 - E^2.$$

Protože $x_i \in [-s_i, s_i]$, tak zřejmě $x_i^2 \leq s_i^2$, a proto $\sum x_i^2 \leq \sum s_i^2$. Protože $E^2 \geq 0$, tak můžeme usoudit, že $V \leq \frac{1}{n} \cdot \sum_{i=1}^{n} s_i^2 = C_0$. Každá možná hodnota V výběrového rozptylu je tedy menší nebo rovna C_0 a ani největší z těchto hodnot, tj. $\overline{V}$, nemůže překročit C_0, tj. $\overline{V} \leq C_0$.

2. Dále dokážeme, že pokud existují požadované hodnoty μ_i, potom $\overline{V} = C_0$.

V tomto případě pro $x_i = \mu_i \cdot s_i$ máme $E = 0$ a $x_i^2 = s_i^2$, odtud

$$V = \frac{1}{n} \cdot \sum_{i=1}^{n} (x_i - E)^2 = \frac{1}{n} \cdot \sum_{i=1}^{n} s_i^2 = C_0.$$

Rozptyl V je vždy $\leq C_0$ a nabývá hodnoty C_0 pro nějaká x_i, která můžeme vyjádřit ve tvaru $x_i = \mu_i \cdot s_i$, takže $\overline{V} = C_0$.

3. Abychom dokončili důkaz, musíme ukázat, že pokud $\overline{V} = C_0$, potom požadovaná μ_i existují.

Nechť $\overline{V} = C_0$. Rozptyl je spojitá funkce na množině $\mathbf{x}_1 \times ... \times \mathbf{x}_n$, takže maximum funkce na této množině je dosaženo v nějakých hodnotách $x_1 \in \mathbf{x}_1 =$

$[-s_1, s_1], ..., x_n \in \mathbf{x}_n = [-s_n, s_n]$. Pro odpovídající hodnoty x_i rozptyl V je roven C_0.

Poněvadž $x_i \in [-s_i, s_i]$, platí, že $x_i^2 \leq s_i^2$; protože $E^2 \geq 0$ máme $V \leq C_0$. Pokud $|x_i|^2 < s_i^2$ nebo $E^2 > 0$, potom máme $V < C_0$. Takže jediná možnost, aby $V = C_0$, je platnost $x_i^2 = s_i^2$ a $E = 0$. Z první rovnosti máme $x_i = \pm s_i$, tj. $x_i = \mu_i \cdot s_i$ pro nějaké $\mu_i \in \{-1, +1\}$. Protože podle definice je E (aritmetický) průměr hodnot x_i, tak z rovnice $E = 0$ máme $\sum_{i=1}^{n} \mu_i \cdot s_i = 0$. Takže pokud $\overline{V} = C_0$, potom požadovaná μ_i existují. Věta je dokázaná.

Poznámka: Položme si otázku, jak vypočítat horní mez pro rozptyl v obecném případě. Již víme, že kvadratická funkce na intervalu nabývá svého maxima v jednom z krajních bodů intervalu. Proto jsme vždy schopní vypočítat horní mez $\overline{V}$ v čase $O(2^n)$, neboť stačí určit rozptyl V pro všech 2^n možných vektorů $x = (x_1^{\varepsilon_1}, ..., x_n^{\varepsilon_n})$, kde $\varepsilon_i \in \{-, +\}$, $x_i^- = \underline{x_i}$ a $x_i^+ = \overline{x_i}$; největší z těchto 2^n hodnot je i požadovaná hodnota $\overline{V}$. Praktický je tento postup samozřejmě k ničemu.

6.2.1 Případ úzkých intervalů a „tzv. širších intervalů“

Pro výpočet $\overline{V}$ můžeme použit stejnou analýzu založenou na derivací, kterou jsme využili při zkoumaní $\underline{V}$. Při analýze $\overline{V}$ využijeme fakt, že když druhá derivace V je ≥ 0, tak maximum nemůže být uvnitř intervalu $[\underline{x}_i, \overline{x}_i]$.

Pokud $\overline{x}_i \leq E$, potom máme $x_i = \underline{x}_i$; pokud $E \leq \underline{x}_i$, potom máme $x_i = \overline{x}_i$; jinak musíme vzít v úvahu obě možnosti $x_i = \underline{x}_i$ a $x_i = \overline{x}_i$.

Když se intervaly neprotínají, tak potom výsledný algoritmus pro výpočet $\overline{V}$ bude mít složitost $O(n^2)$. Jak se dále ukážeme, algoritmus se složitosti $O(n^2)$ je přípustný nejenom pro původní intervaly $[\widetilde{x}_i - \Delta_i, \widetilde{x}_i + \Delta_i]$, které se neprotínají, ale také pro více obecný případ, kdy „zúžené“ intervaly $[x_i^-, x_i^+]$, kde $x_i^- = \widetilde{x}_i - \Delta_i/n$ a $x_i^+ = \widetilde{x}_i + \Delta_i/n$, se neprotínají, a tvrzení dokonce platí i pro obecnější případ, kdy pro nějaké celé číslo $K < n$ žádná podmnožina z více než K úzkých intervalů $[x_i^-, x_i^+]$ nemá společný průnik. Následující algoritmus je uveden v [6], [7], [8], [9].

$O(n^2)$ **algoritmus pro výpočet $\overline{V}$ pro případ úzkých intervalů a tzv. „širších intervalů“.**

Algoritmus postupuje následujícím způsobem:

1. Nejprve roztřídíme všech $2n$ koncových bodů „zúžených“ intervalů $\widetilde{x}_i - \Delta_i/n$

a $\widetilde{x}_i + \Delta_i/n$ do posloupnosti $x_{(1)} \leq x_{(2)} \leq \ldots \leq x_{(2n)}$. Toto nám umožní rozdělit reálnou osu na $2n+1$ částí („malých intervalů“) $[x_{(k)}, x_{(k+1)}]$, kde označíme $x_{(0)} \stackrel{def}{=} -\infty$ a $x_{(2n+1)} \stackrel{def}{=} +\infty$.

2. Vypočítáme $\underline{E} = \frac{1}{n}\sum_{i=1}^{n} \underline{x}_i$ a $\overline{E} = \frac{1}{n}\sum_{i=1}^{n} \overline{x}_i$ a vybereme všechny „malé intervaly“ $[x_{(k)}, x_{(k+1)}]$, které se protínají s $[\underline{E}, \overline{E}]$.

3. Pro každý z vybraných „malých intervalů“ $[x_{(k)}, x_{(k+1)}]$ a pro každé $i \in \{1, ..., n\}$ přiřadíme x_i následující hodnotu:

   - pokud $x_{(k+1)} < \widetilde{x}_i - \Delta_i/n$, potom položíme $x_i = \overline{x}_i$;
   - pokud $x_{(k)} > \widetilde{x}_i + \Delta_i/n$, potom položíme $x_i = \underline{x_i}$;
   - pro všechná ostatní i budeme uvažovat obě možné hodnoty $x_i = \overline{x_i}$ a $x_i = \underline{x_i}$.

   Jako výsledek, pro každý z vybraných „malých intervalů“, dostaneme jednu nebo více posloupnosti $x = (x_1, \ldots, x_n)$ (více pokud pro nějaké i se budou brát v úvahu obě možnosti $x_i = \overline{x_i}$ a $x_i = \underline{x_i}$). Pro každou z těchto posloupnosti zkontrolujeme, zda průměr E vybraných hodnot $x_1, \ldots, x_n$ je skutečně uvnitř tohoto „malého intervalu“, a pokud ano, vypočítáme rozptyl těchto vybraných hodnot.

4. Největší hodnotu z vypočtených rozptylů položíme rovnu $\overline{V}$.

Důkaz kvadratické náročnosti výše popsaného algoritmu Ukážeme, že výše popsaný algoritmus skutečně počítá $\overline{V}$ v kvadratickém čase pro všechny případy, kdy pro nějaké celé číslo $K < n$ žádná podmnožina z více než K „zúžených“ intervalů $\mathbf{x}_i$ nemá společný průnik.

1. Nechť $x_1, \ldots, x_n$ jsou hodnoty, ve kterých rozptyl nabývá svého maxima na krychli $\mathbf{x}_1 \times \cdots \times \mathbf{x}_n$. Pokud, až na jednu proměnnou x_i, všechny hodnoty proměnných zvolíme pevně, potom V bude kvadratickou funkcí v x_i. Když funkce V nabude maxima pro $x_1 \in \mathbf{x}_1, \ldots, x_n \in \mathbf{x}_n$, potom tato kvadratická funkce jedné proměnné nabude svého maxima na intervalu $\mathbf{x}_i$ v bodě x_i. Ukážeme, že tato kvadratická funkce má (globální) minimum v bodě $x_i = E_i'$, kde E_i' je průměr všech hodnot $x_1, \ldots, x_n$ mimo x_i. Tato kvadratická funkce je nezáporná a funkce

na intervalu $\mathbf{x}_i = [\underline{x}_i, \overline{x}_i]$ nabude svého maxima v jednom z koncových bodů tohoto intervalu. Libovolná kvadratická funkce jedné proměnné je symetrická s ohledem na polohu jejího globálního minima, takže její maximum na jakýmkoliv intervalu je v bodě, který se nachází co nejdále od minima. Existuje přesně jeden bod, který je stejně vzdálený od obou koncových bodů intervalu $\mathbf{x}_i$ - je to prostřední bod $\widetilde{x}_i$. Podle toho, kde se nachází globální minimum vzhledem k interval $\mathbf{x}_i$, máme následující tři případy:

- Pokud $E_i^{'}$ je vlevo od prostředního bodu $\widetilde{x}_i$, tj. pokud $E_i^{'} < \widetilde{x}_i$, potom horní koncový bod intervalu je nejvzdálenějším bodem od $E_i^{'}$. V tomto případě kvadratická funkce nabude svého maxima v horním koncovém bodě intervalu, tj. $x_i = \overline{x}_i$.

- Stejným způsobem, pokud $E_i^{'}$ je vpravo od prostředního bodu $\widetilde{x}_i$, tj. pokud $E_i^{'} > \widetilde{x}_i$, potom dolní koncový bod je nejvzdálenějším bodem od $E_i^{'}$. V tomto případě kvadratická funkce nabude svého maxima v dolním koncovém bodě intervalu, tj. $x_i = \underline{x}_i$.

- Pokud $E_i^{'} = \widetilde{x}_i$, potom maximum V je v obou koncových bodech intervalu $\mathbf{x}_i = [\underline{x}_i, \overline{x}_i]$.

2. V třetím případě máme buď $x_i = \underline{x}_i$ nebo $x_i = \overline{x}_i$. Záleží přitom pouze na tom, jestli x_i je rovné dolnímu nebo hornímu koncovému bodu, takže můžeme „nakombinovat“ odpovídajíce situace s případem 1 a 2. Dospějeme k závěru, že může nastat jedná ze dvou následujících situací:

- buď $E_i^{'} \leq \widetilde{x}_i$ a $x_i = \overline{x}_i$;

- buď $E_i^{'} \geq \widetilde{x}_i$ a $x_i = \underline{x}_i$.

3. Přeformulujeme tyto podmínky pokud jde o průměr E_{max} hodnot $x_1, ..., x_n$. Průměr $E_i^{'}$ můžeme vyjádřit jako

$$E_i^{'} = \tfrac{1}{n-1} \cdot \sum_{j \neq i} x_j,$$

Tuto sumu můžeme rozepsat jako $\sum_{j \neq i} x_j = \sum_{j=1}^{n} x_j - x_i$. Podle definice E_{max} máme

$$E_{max} = \frac{1}{n} \cdot \sum_{j=1}^{n} x_j,$$

odkud $\sum_{j=1}^{n} x_j = n \cdot E_{max}$. A proto dostáváme následující vyjádření

$$E_i^{'} = \frac{n \cdot E_{max} - x_i}{n-1}. \tag{6.8}$$

Dále aplikujeme tento vzorec na předchozí tři případy.

3.1. V prvním případě máme $\widetilde{x}_i \geq E_i^{'}$. S využitím (6.8) vzorce pro $E_i^{'}$ dostáváme nerovnost

$$\widetilde{x}_i \geq \frac{n \cdot E_{max} - x_i}{n-1}.$$

Vynásobením obou stran nerovnosti $n-1$ a využitím faktu, že $x_i = \overline{x}_i = \widetilde{x}_i + \Delta_i$, dostáváme

$$(n-1) \cdot \widetilde{x}_i \geq n \cdot E_{max} - \widetilde{x}_i - \Delta_i.$$

Všechny členy nerovnosti převedeme na levou stranu, napravo ponecháme jenom $n \cdot E_{max}$ a celou nerovnost vydělíme n, čímž dostáváme:

$$E_{max} \leq \widetilde{x}_i + \frac{\Delta_i}{n}.$$

3.2. Ve druhém případě máme $\widetilde{x}_i \leq E_i^{'}$. S využitím (6.8) vzorce pro $E_i^{'}$ dostáváme nerovnost

$$\widetilde{x}_i \leq \frac{n \cdot E_{max} - x_i}{n-1}.$$

Vynásobením obou stran nerovnosti $n-1$ a využitím faktu, že $x_i = \underline{x}_i = \widetilde{x}_i - \Delta_i$, dostáváme

$$(n-1) \cdot \widetilde{x}_i \leq n \cdot E_{max} - \widetilde{x}_i + \Delta_i.$$

Všechny členy nerovnosti převedeme na levou stranu, napravo ponecháme jenom $n \cdot E_{max}$ a celou nerovnost vydělíme n, čímž dostáváme:

$$E_{max} \geq \widetilde{x}_i - \frac{\Delta_i}{n}.$$

4. Části 3.1. a 3.2. tohoto důkazu shrneme následujícím způsobem:

- V prvním případě máme $E_{max} \leq \widetilde{x}_i + \Delta_i/n$ a $x_i = \overline{x}_i$,
- Ve druhém případě máme $E_{max} \geq \widetilde{x}_i - \Delta_i/n$ a $x_i = \underline{x}_i$.

Tudíž platí:

- Pokud $E_{max} < \widetilde{x}_i - \Delta_i/n$, tak to znamená, že se nemůže jednat o druhý případ, takže platí první případ a proto musí být $x_i = \overline{x}_i$.
- Pokud $E_{max} > \widetilde{x}_i + \Delta_i/n$, tak to znamená, že se nemůže jednat o první případ, takže platí druhý případ a proto musí být $x_i = \underline{x}_i$.

Jediný případ, kdy nevíme, který z koncových bodů si máme zvolit pro x_i, je když E_{max} patří do „zúženého“ intervalu $[\widetilde{x}_i - \Delta_i/n, \widetilde{x}_i + \Delta_i/n]$.

5. Teď, když víme kde se nachází E_{max} vzhledem ke koncovým bodům „zúžených“ intervalů, můžeme určit všechny optimální hodnoty x_i - mimo ty, které jsou uvnitř tohoto „zúženého“ intervalu. Když uvažujeme případ, kdy více než K „zúžených“ intervalů nemá společný průnik, potom máme ne více jak K nerozhodnutých hodnot x_i. Zkoušení všech možných kombinaci dolních a horních koncových bodů pro těchto $\leq K$ hodnot zabere $\leq 2^K$ kroků, takže celkový počet kroků je $O(2^K \cdot n^2)$. Když K je konstanta, potom celkový počet kroků je $O(n^2)$.

Čas výpočtu je kvadratický v n, ale roste exponenciálně s K. Pokud K poroste, tak potom k provedení algoritmu bude potřeba stále více a více času a jak bylo ukázáno v důkazu, výpočet zabere $O(2^K \cdot n^2)$ kroků. V nejhorším případě, když naše podmínky nejsou splněny a $K = O(n)$ „zúžených“ intervalů má společný bod, bude tento algoritmus potřebovat $O(2^n \cdot n^2)$ výpočetních kroků.

Algoritmus se složitosti $O(n \cdot log(n))$

V mnoha praktických případech pro výpočet $\overline{V}$ existují polynomiální algoritmy. Jeden z takových případů, kdy jsou měření dostatečně přesná, tj. když se „zúžené“ intervaly

$$[\widetilde{x}_i - \frac{\Delta_i}{n}, \widetilde{x}_i + \frac{\Delta_i}{n}] \tag{6.9}$$

neprotínají, jsme popsali výše. Můžeme tedy říci, že víme jak efektivně vypočítat $\overline{V}$, když pro každé $i \neq j$ máme

$$|\widetilde{x}_i - \widetilde{x}_j| \geq \frac{\Delta_i}{n} + \frac{\Delta_j}{n}. \tag{6.10}$$

Algoritmus pro výpočet $\overline{V}$ v čase $O(n \cdot log(n))$ pracuje při slabší podmínce (jedna se o obecnější případ)

$$|\widetilde{x}_i - \widetilde{x}_j| \geq \frac{|\Delta_i - \Delta_j|}{n}. \tag{6.11}$$

Tato podmínka je skutečně o dost slabší. Pro případ, kdy jsou všechna měření stejně přesná, tj. $\Delta_i = \Delta$ pro všechna i, podmínka (6.10) platí jenom pro $\Delta \leq (n/2) \cdot \min_{i \neq j} |\widetilde{x}_i - \widetilde{x}_j|$, kdežto podmínka (6.11) platí pro každé Δ. Proto můžeme mít větší měřící nejistotu Δ než v předchozí situaci a stále budeme schopní vypočítat přesnou mez $\overline{V}$ v polynomiálním čase.

Předchozí algoritmus s kvadratickou časovou složitosti vylepšíme a snížíme složitost na $O(n \cdot log(n))$. Nový algoritmus pracuje za stejných podmínek jako předchozí algoritmus, tj. pro případ, kdy pro nějaké celé číslo $K < n$ žádná podmnožina z více než K „zúžených“ intervalů $[\widetilde{x}_i - \Delta_i/n, \widetilde{x}_i + \Delta_i/n]$ nemá společný průnik. Případ úzkých intervalů a tzv. „širších intervalů“ je dílčí případ tohoto případu. Z [4],[14] máme, že $O(n \cdot log(n))$ algoritmus postupuje následujícím způsobem:

1. Uspořádáme hodnoty $\widetilde{x}_i$ do rostoucí posloupností. Můžeme předpokládat, že tyto středové body intervalů jsou uspořádané od nejmenšího po největší.

2. Pro každé celé číslo p od 0 do n vypočítáme rozptyl $V_p = M_p - (E_p)^2$ pro vektor $v^{(p)} = (\underline{x}_1, ..., \underline{x}_p, \overline{x}_{p+1}, ..., \overline{x}_n)$, kde

$$E_0 = \frac{1}{n} \sum_{i=1}^{n} \overline{x}_i;$$

$$M_0 = \frac{1}{n} \sum_{i=1}^{n} (\overline{x}_i)^2;$$

$$E_p \stackrel{def}{=} \frac{1}{n} \cdot \sum_{i=1}^{p} \underline{x}_i + \frac{1}{n} \sum_{i=p+1}^{n} \overline{x}_i;$$

$$M_p \stackrel{def}{=} \frac{1}{n} \cdot \sum_{i=1}^{p} (\underline{x}_i)^2 + \frac{1}{n} \sum_{i=p+1}^{n} (\overline{x}_i)^2.$$

Tyto hodnoty vypočítáme postupně. Jakmile známe E_p a M_p, tak potom jsme schopní určit E_{p+1} a M_{p+1} jako $E_{p+1} = E_p + \frac{1}{n}\underline{x}_{p+1} - \frac{1}{n}\overline{x}_{p+1}$ a $M_{p+1} = M_p + \frac{1}{n}(\underline{x}_{p+1})^2 - \frac{1}{n}(\overline{x}_{p+1})^2$.

3. $\overline{V}$ najdeme jako největší hodnotu z výsledných $n+1$ hodnot $V_0, ..., V_n$.

Počet výpočetních kroků

Třídění trvá $O(n \cdot log(n)))$ kroků. Výpočet počátečních hodnot M_0, E_0 a V_0 nám potrvá lineární čas $O(n)$. Pro každé p od 0 do $n-1$ potřebujeme konstantní počet kroků k výpočtu dalších hodnot M_{p+1}, E_{p+1} a V_{p+1}. K nalezení největší hodnoty V_p z $n+1$ hodnot zabere $O(n)$ kroků. Celkově potřebujeme

$$O(n \cdot log(n)) + O(n) + O(n) + O(n) = O(n \cdot log(n))$$

kroků, takže tento algoritmus vypočítá $\overline{V}$ během $O(n \cdot log(n))$ kroků.

Ověření algoritmu

Rozptyl je nezáporná funkce pro všechna x_i. Víme, že maximum této funkce na každém intervalu $[\underline{x}_i, \overline{x}_i]$ se nachází v jednom z koncových bodů tohoto intervalu. Proto maximum $\overline{V}$ rozptylu je dosaženo ve vektoru $x = (x_1, ..., x_n)$, ve kterým každá hodnota x_i je rovna buď $\underline{x}_i$ nebo $\overline{x}_i$.

Nejdříve ověříme algoritmus pro případ, kdy $|\widetilde{x}_i - \widetilde{x}_j| \geq \frac{|\Delta_i - \Delta_j|}{n}$ pro všechny $i \neq j$.

Abychom dokázali náš algoritmus, potřebujeme dokázat, že maximum V je dosaženo v jednom z vektoru $x^{(p)}$, ve kterém všechny dolní meze $\underline{x}_i$ předchází všechny horní meze $\overline{x}_i$. Toto dokážeme sporem. Nechť maximum je dosaženo ve vektoru x, ve kterém jedna z dolních mezí následuje po jedné z horních mezí. V každém takovém vektoru nechť i je index největší horní meze po které následuje dolní mez. Potom v optimálním vektoru x máme $x_i = \overline{x}_i$ a $x_{i+1} = \underline{x}_{i+1}$.

Protože maximum nabývá pro $x_i = \overline{x}_i$, potom výměnou $\underline{x}_i = \overline{x}_i - 2 \cdot \Delta_i$ snížíme hodnotu rozptylu nebo ji ponecháme nezměněnou. Popišme jak se změní rozptyl pokud provedeme tuto výměnu. V sumě M vyměníme $(\overline{x}_i)^2$ za

$$(\underline{x}_i)^2 = (\overline{x}_i - 2 \cdot \Delta_i)^2 = (\overline{x}_i)^2 - 4 \cdot \Delta_i \cdot \overline{x}_i + 4 \cdot \Delta_i^2.$$

Proto se hodnota M změní na $M + \Delta M_i$, kde

$$\Delta M_i = -\tfrac{4}{n} \cdot \Delta_i \cdot \overline{x}_i + \tfrac{4}{n} \cdot \Delta_i^2.$$

Střední hodnota E se změní na $E + \Delta E_i$, kde $\Delta E_i = -\frac{2 \cdot \Delta_i}{n}$. Proto hodnota E^2 se změní na $(E + \Delta E_i)^2 = E^2 + \Delta(E^2)_i$, kde

$$\Delta(E^2)_i = 2 \cdot E \cdot \Delta E_i + \Delta E_i^2 = -\tfrac{4}{n} \cdot E \cdot \Delta_i + \tfrac{4}{n^2} \cdot \Delta_i^2.$$

Rozptyl V se změní na $V + \Delta V_i$, kde

$$\Delta V_i = \Delta M_i - \Delta(E^2)_i = -\tfrac{4}{n} \cdot \Delta_i \cdot \overline{x}_i + \tfrac{4}{n} \cdot \Delta_i^2 + \tfrac{4}{n} \cdot E \cdot \Delta_i - \tfrac{4}{n^2} \cdot \Delta_i^2 =$$

$$\tfrac{4}{n} \cdot \Delta_i \cdot (-\overline{x}_i + \Delta_i + E - \tfrac{\Delta_i}{n}).$$

Podle definice $\overline{x}_i = \widetilde{x}_i + \Delta_i$, proto $-\overline{x}_i + \Delta_i = -\widetilde{x}_i$. Odtud máme

$$\Delta V_i = \tfrac{4}{n} \cdot \Delta_i \cdot (-\widetilde{x}_i + E - \tfrac{\Delta_i}{n}).$$

Protože V nabývá maxima v x, máme $\Delta V_i \leq 0$ a odtud

$$E \leq \widetilde{x}_i + \frac{\Delta_i}{n}. \tag{6.12}$$

Stejným způsobem, pokud maximum je dosaženo v $x_{i+1} = \underline{x}_i$, výměnou za $\overline{x}_{i+1} = \underline{x}_{i+1} + 2 \cdot \Delta_{i+1}$ buď zmenšíme hodnotu rozptylu nebo ji ponecháme nezměněnou. Popišme jak se změní rozptyl po této výměně. V sumě M vyměníme $(\underline{x}_{i+1})^2$ s

$$(\overline{x}_{i+1})^2 = (\underline{x}_{i+1} + 2 \cdot \Delta_{i+1})^2 = (\underline{x}_{i+1})^2 + 4 \cdot \Delta_{i+1} \cdot \underline{x}_{i+1} + 4 \cdot \Delta_{i+1}^2.$$

Hodnota M se změní na $M + \Delta M_{i+1}$, kde

$$\Delta M_{i+1} = \tfrac{4}{n} \cdot \Delta_{i+1} \cdot \underline{x}_{i+1} + \tfrac{4}{n} \cdot \Delta_{i+1}^2.$$

Střední hodnota E se změní na $E + \Delta E_{i+1}$, kde $\Delta E_{i+1} = \frac{2 \cdot \Delta_{i+1}}{n}$. Hodnota E^2 se změní na $(E + \Delta E_{i+1})^2 = E^2 + \Delta(E^2)_{i+1}$, kde

$$\Delta(E^2)_{i+1} = 2 \cdot E \cdot \Delta E_{i+1} + \Delta E_{i+1}^2 = \tfrac{4}{n} \cdot E \cdot \Delta_{i+1} + \tfrac{4}{n^2} \cdot \Delta_{i+1}^2.$$

Rozptyl V se změní na $V + \Delta V_{i+1}$, kde

$$\Delta V_{i+1} = \Delta M_{i+1} - \Delta(E^2)_{i+1} =$$

$$\frac{4}{n} \cdot \Delta_{i+1} \cdot \underline{x}_{i+1} + \frac{4}{n} \cdot \Delta_{i+1}^2 - \frac{4}{n} \cdot E \cdot \Delta_{i+1} - \frac{4}{n^2} \cdot \Delta_{i+1}^2 =$$

$$\frac{4}{n} \cdot \Delta_{i+1} \cdot (\underline{x}_{i+1} + \Delta_{i+1} - E - \frac{\Delta_{i+1}}{n}).$$

Podle definice $\underline{x}_{i+1} = \widetilde{x}_{i+1} - \Delta_{i+1}$, proto $\underline{x}_{i+1} + \Delta_{i+1} = \widetilde{x}_{i+1}$. Odtud máme

$$\Delta V_{i+1} = \frac{4}{n} \cdot \Delta_{i+1} \cdot (\widetilde{x}_{i+1} - E - \frac{\Delta_{i+1}}{n}).$$

Protože V nabývá maxima v x, máme $\Delta V_{i+1} \leq 0$ a odtud

$$E \geq \widetilde{x}_{i+1} - \frac{\Delta_{i+1}}{n}. \tag{6.13}$$

Dále také můžeme najednou změnit obě hodnoty x_i a x_{i+1}. V tomto případě změna ΔM v M je jednoduše suma změn vycházejících z x_i a x_{i+1}: $\Delta M = \Delta M_i + \Delta M_{i+1}$, a změna ΔE v E je suma odpovídajících změn: $\Delta E = \Delta E_i + \Delta E_{i+1}$. Takže pro

$$\Delta V = \Delta M - \Delta(E^2) = \Delta M - 2 \cdot E \cdot \Delta E - \Delta E^2,$$

máme

$$\Delta V = \Delta M_i + \Delta M_{i+1} -$$

$$2 \cdot E \cdot \Delta E_i - 2 \cdot E \cdot \Delta E_{i+1} - (\Delta E_i)^2 - (\Delta E_{i+1})^2 - 2 \cdot \Delta E_i \cdot \Delta E_{i+1}.$$

Odsud dostáváme

$$\Delta V = (\Delta M_i - 2 \cdot E \cdot \Delta E_i - (\Delta E_i)^2) + (\Delta M_{i+1} - 2 \cdot E \cdot \Delta E_{i+1} - (\Delta E_{i+1})^2)$$

$$-2 \cdot \Delta E_i \cdot \Delta E_{i+1},$$

tj.,

$$\Delta V = \Delta V_i + \Delta V_{i+1} - 2 \cdot \Delta E_i \cdot \Delta E_{i+1}.$$

Máme tedy vyjádření pro $\Delta V_i, \Delta V_{i+1}, \Delta E_i = -\frac{2 \cdot \Delta_i}{n}$ a $\Delta E_{i+1} = \frac{2 \cdot \Delta_{i+1}}{n}$, takže můžeme učinit závěr, že $\Delta V = \frac{4}{n} \cdot D(E)$, kde

$$D(E) \stackrel{def}{=} \Delta_i \cdot (-\widetilde{x}_i + E - \frac{\Delta_i}{n}) + \Delta_{i+1} \cdot (\widetilde{x}_{i+1} - E - \frac{\Delta_{i+1}}{n}) + \frac{2}{n} \cdot \Delta_i \cdot \Delta_{i+1}. \tag{6.14}$$

Protože funkce V nabývá svého maxima v x, máme $\Delta V \leq 0$ a odtud $D(E) \leq 0$.

Vyjádření $D(E)$ je lineární funkcí E. Z (6.12) a (6.13) víme, že

$$\widetilde{x}_{i+1} - \frac{\Delta_{i+1}}{n} \leq E \leq \widetilde{x}_i + \frac{\Delta_i}{n}.$$

Pro $E = E^- \stackrel{def}{=} \widetilde{x}_{i+1} - \frac{\Delta_{i+1}}{n}$ máme

$$D(E^-) = \Delta_i \cdot (-\widetilde{x}_i + \widetilde{x}_{i+1} - \frac{\Delta_{i+1}}{n} - \frac{\Delta_i}{n}) + \frac{2}{n} \cdot \Delta_i \cdot \Delta_{i+1} =$$

$$\Delta_i \cdot (-\widetilde{x}_i + \widetilde{x}_{i+1} + \frac{\Delta_{i+1}}{n} - \frac{\Delta_i}{n}).$$

Uvažujeme případ, kdy $|\widetilde{x}_{i+1} - \widetilde{x}_i| > \frac{|\Delta_i - \Delta_{i+1}|}{n}$. Protože hodnoty $\widetilde{x}_i$ jsou uspořádaný ve vzrůstajícím pořadí, máme $\widetilde{x}_{i+1} \geq \widetilde{x}_i$ a odsud

$$\widetilde{x}_{i+1} - \widetilde{x}_i = |\widetilde{x}_{i+1} - \widetilde{x}_i| > \frac{|\Delta_i - \Delta_{i+1}|}{n} \geq \frac{\Delta_i}{n} - \frac{\Delta_{i+1}}{n}.$$

Takže můžeme udělat závěr, že $D(E^-) > 0$.

Pro $E = E^+ \stackrel{def}{=} \widetilde{x}_i + \frac{\Delta_i}{n}$ máme

$$D(E^+) = \Delta_{i+1} \cdot (\widetilde{x}_{i+1} - \widetilde{x}_i - \frac{\Delta_i}{n} - \frac{\Delta_{i+1}}{n}) + \frac{2}{n} \cdot \Delta_i \cdot \Delta_{i+1} =$$

$$\Delta_{i+1} \cdot (-\widetilde{x}_i + \widetilde{x}_{i+1} + \frac{\Delta_i}{n} - \frac{\Delta_{i+1}}{n}).$$

Zde $|\widetilde{x}_{i+1} - \widetilde{x}_i| > \frac{|\Delta_i - \Delta_{i+1}|}{n}$, takže taky můžeme udělat závěr, že $D(E^+) > 0$.

Protože lineární funkce $D(E)$ je kladná pro oba koncové body intervalu $[E^-, E^+]$, musí být kladná pro každou hodnotu E z tohoto intervalu, což je ve sporu s naším závěrem, že $D(E) \leq 0$ pro střední hodnotu $E \in [E^-, E^+]$. Spor ukazuje, že maximum rozptylu V je skutečně dosaženo v $x^{(p)}$. Algoritmus je ověřen.

Geometrický význam podmínky (6.11)

Podmínka $|\widetilde{x}_i - \widetilde{x}_j| \geq \frac{|\Delta_i - \Delta_j|}{n}$ znamená, že pokud $\widetilde{x}_i \geq \widetilde{x}_j$, tak potom máme

$$\widetilde{x}_i - \widetilde{x}_j \geq \frac{\Delta_i - \Delta_j}{n},$$

tj.,

$$\widetilde{x}_i - \frac{\Delta_i}{n} \geq \widetilde{x}_j - \frac{\Delta_j}{n}$$

a taky

$$\widetilde{x}_i - \widetilde{x}_j \geq \frac{\Delta_j - \Delta_i}{n},$$

tj.,

$$\widetilde{x}_i + \frac{\Delta_i}{n} \geq \widetilde{x}_j + \frac{\Delta_j}{n}.$$

To znamená, že žádný „zúžený“ interval (6.9) není podintervalem vnitřku jiného „zúženého“ intervalu.

Pokud jeden z „zúžených“ intervalů je podintervalem jiného, potom podmínka (6.11) neplatí. Podmínka (6.11) tedy znamená, že „zúžené“ podintervaly nejsou podintervaly navzájem.

6.2.2 Úzké intervaly splňující podmínku o podmnožinách

V předchozím textu jsme popsali $O(n \cdot log(n))$ algoritmus pro výpočet $\overline{V}$ v případě úzkých intervalů. V této kapitole ukážeme, že existuje lineární algoritmus pro výpočet $\overline{V}$ v případě úzkých intervalů.

Tento algoritmus pracuje i pro případ, kdy žádný z „zúžených“ intervalů $[\widetilde{x}_i - \Delta_i/n, \widetilde{x}_i + \Delta_i/n]$ není podintervalem vnitřku jiného „zúženého“ intervalu, tj. když $|\widetilde{x}_i - \widetilde{x}_j| \geq \frac{|\Delta_i - \Delta_j|}{n}$ pro všechna $i \neq j$.

Definice 2 Řekneme, že soubor intervalů odpovídá podmínce o podmnožinách pro „zúžené“ intervaly, pokud žádný z „zúžených intervalů“ $[x_i^-, x_i^+]$, kde $x_i^- \stackrel{def}{=} \widetilde{x}_i - \Delta_i/n$ a $x_i^+ \stackrel{def}{=} \widetilde{x}_i + \Delta_i/n$, není podintervalem vnitřku jiného „zúženého“ intervalu, tj. když $|\widetilde{x}_i - \widetilde{x}_j| \geq \frac{|\Delta_i - \Delta_j|}{n}$ pro všechna $i \neq j$.

Tento případ obsahuje případ úzkých intervalů a případ jednoho měřícího přístroje.

Lineární algoritmus pro výpočet $\overline{V}$ pro „zúžené“ intervaly splňující podmínku o podmnožinách.

Nejvíce časově náročný krok v $O(n \cdot log(n))$ algoritmu popsaném v Kapitole 6.2.1 je třídění, které samotné má složitost $O(n \cdot log(n))$. Abychom snížili časovou složitost, potřebujeme obejít nutnost použít třídění.

Hlavní myšlenkou tohoto algoritmu je použití lineárního algoritmu pro výpočet mediánu místo použití $O(n \cdot log(n))$ algoritmu pro tříděni.

Algoritmus uvedený v [15] je nepřesný a pro některé případy nenajde řešeni, proto ho opravíme a doplníme. Následující algoritmus je založen na iteracích. Při každé iteraci máme 3 množiny:

- množina I^- všech indexů i od 1 do n, pro které již víme, že pro optimální

vektor x máme $x_i = \underline{x}_i$;

- množina I^+ všech všech indexů j těch prvků, pro které již víme, že pro optimální vektor x máme $x_j = \overline{x}_j$;

- množina I indexů i, pro které jsme se ještě nerozhodli.

Na začátku položíme $I^- = I^+ = \emptyset$ a $I = \{1, ..., n\}$. Při každé iteraci obnovujeme dvě pomocné proměnné $S^- \stackrel{def}{=} \sum_{i \in I^-} \underline{x}_i$ a $S^+ \stackrel{def}{=} \sum_{j \in I^+} \overline{x}_j$. Tyto hodnoty můžeme počítat tak, že jednoduše spočítáme tyto sumy, ale abychom zrychlili výpočet, při každé iteraci budeme obnovovat tyto pomocné proměnné rychlejším způsobem než přepočítávání odpovídajících sum. Zpočátku máme $I^- = I^+ = \emptyset$ a $S^- = S^+ = 0$.

Při každé iteraci postupujeme následujícím způsobem:

- Nejdříve vypočítáme medián m množiny I (medián z hlediska třídění podle $\widetilde{x}_i$);

- Analyzujeme prvky nerozhodnuté množiny I prvek po prvku tak, že je rozdělíme do dvou podmnožin

$$P^- = \{i : \widetilde{x}_i \leq \widetilde{x}_m\};$$

$$P^+ = \{j : \widetilde{x}_j > \widetilde{x}_m\};$$

  Pokud máme několik prvků rovných $\widetilde{x}_m$, tak potom pouze jeden z nich zařadíme do množiny P^-, zbylé prvky zařadíme do množiny P^+.

- Vypočítáme

$$e^- = S^- + \sum_{i \in P^-} \underline{x}_i$$

$$e^+ = S^+ + \sum_{i \in P^+} \overline{x}_j;$$

- Pokud $n \cdot x_m^- < e^- + e^+$, potom vyměníme I^- s $I^- \cup P^-$, S^- s e^- a I s P^+;

- Pokud $n \cdot x_m^- > e^- + e^+$, potom vyměníme I^+ s $I^+ \cup P^+$, S^+ s e^+ a I s P^-;

- Pokud $n \cdot x_m^- = e^- + e^+$, potom vyměníme I^- s $I^- \cup P^-$, I^+ s $I^+ \cup P^+$, I s $\emptyset$;

- Pokud velikost množiny I je 1 a $I = \{i\}$ pro nějaké $i \in \{1, ..., n\}$, potom vyměníme I s $\emptyset$, prázdnou pomocnou množinu I_1^- vyměníme s $I^- \cup \{i\}$ a prázdnou pomocnou množinu I_1^+ vyměníme s $I^+ \cup \{i\}$.

Při každé iteraci množina nerozhodnutých indexů je zmenšená napůl. Iterace pokračuje dokud všechny indexy nejsou roztříděné, potom vrátíme $\overline{V}$ jako maximální hodnotu populačního rozptylu pro jeden z vektorů x. Pro první platí $x_i = \underline{x}_i$ pro $i \in I^-$ a $x_j = \overline{x}_j$ pro $j \in I^+$, pro druhý platí $x_i = \underline{x}_i$ pro $i \in I_1^-$ a $x_j = \overline{x}_j$ pro $j \in I^+$, pro třetí platí $x_i = \underline{x}_i$ pro $i \in I^-$ a $x_j = \overline{x}_j$ pro $j \in I_1^+$.

Pokud nějaký interval $\mathbf{x}_i$ je degenerovaný, tj. $\mathbf{x}_i = [x_i, x_i]$, potom potřebujeme tento algoritmus upravit následujícím způsobem:

- Nejdřív počáteční množině I přiřadíme všechny indexy, které odpovídají nedegenerovaným intervalům;

- Potom přepočítáme sumu e pro všechny přesně známé hodnoty x_i (odpovídající degenerovaným intervalům);

- Následně při každé iteraci, místo toho, abychom porovnávali $n \cdot x_m^-$ se sumou $e^- + e^+$, porovnáváme $n \cdot x_m^-$ se sumou $e^- + e^+ + e$;

- Nakonec při výpočtu populačního rozptylu, který bude vracen jako $\overline{V}$, musíme zařadit i degenerované hodnoty x_i.

Zkontrolujme, zda nový algoritmus pro výpočet $\overline{V}$ má lineární složitost. Při každé iteraci výpočet mediánu zabere lineární čas a ostatní operace s I zaberou čas t lineární v počtů prvků $|I|$ množiny I: $t \leq C \cdot |I|$ pro nějaké C. Začneme s množinou I o velikosti n, při další iteraci máme množinu o velikosti $n/2$, potom $n/4$ atd. Proto celkový výpočetní čas je $\leq C \cdot (n + n/2 + n/4 + ...) \leq C \cdot 2n$, tj. lineární v n.

Pokud žádné dva „zúžené“ intervaly nejsou podmnožinami nějakého jiného intervalu, potom mohou být lineárně uspořádané v lexikografickém pořadí. Odsud

máme $x_1^- \leq x_2^- \leq ... \leq x_n^-$, $x_1^+ \leq x_2^+ \leq ... \leq x_n^+$, a proto průměry $\widetilde{x}_i = (x_i^- + x_i^+)/2$ jsou uspořádané do řady: $\widetilde{x}_1 \leq \widetilde{x}_2 \leq ... \leq \widetilde{x}_n$.

Podle tohoto třídění hodnotu $\overline{V}$ získáme pro jeden z vektorů $x^{(k)} = (\underline{x}_1, ..., \underline{x}_k, \overline{x}_{k+1}, ..., \overline{x}_n)$, tj. $V = V(x^{(k)})$ pro nějaké k.

Když vyměníme $x^{(k)}$ s $x^{(k-1)}$, tj. když vyměníme $\underline{x}_k$ s $\overline{x}_k = \underline{x}_k + 2\Delta_k$, tak potom platí $V_{k-1} - V_k = \frac{4\Delta_k}{n} \cdot (x_k^- - E_k)$, kde $E_k \stackrel{def}{=} E(x^{(k)})$.

Odsud $V_{k-1} < V_k$ tehdy a jen tehdy pokud $x_k^- < E_k$. Vynásobením obou stran této nerovnosti n dostáváme ekvivalentní nerovnost $n \cdot x_k^- < n \cdot E_k$, kde $n \cdot E_k = \sum_{i=1}^{k} \underline{x}_i + \sum_{j=k+1}^{n} \overline{x}_j$. Stejným způsobem $V_{k-1} > V_k$ tehdy a jen tehdy pokud $x_k^- > E_k$, a $V_{k-1} = V_k$ tehdy a jen tehdy pokud $x_k^- = E_k$.

Když přecházíme od k k $k+1$, vyměníme větší hodnotu $\overline{x}_{k+1}$ v sumě $n \cdot E_k$ za menší hodnotu $\underline{x}_k$. Tudíž řada $n \cdot E_k$ je přímo klesající s k, kdežto x_k^- je rostoucí s k. Jakmile máme $n \cdot x_k^- < n \cdot E_k$, tj. $V_{k-1} < V_k$, tak tyto nerovnosti taky platí i pro menší k. Stejným způsobem jakmile máme $n \cdot x_k^- > n \cdot E_k$, tj. $V_{k-1} > V_k$, tak tyto nerovnosti taky platí i pro větší k.

Jakmile $n \cdot x_k^- = n \cdot E_k$, tj. $V_{k-1} = V_k$, potom máme $V_k > V_{k+1} > ...$ a $V_k = V_{k-1} > V_{k-2} > ...$, tj. $V_k = V_{k-1}$ bude největší hodnota V.

Tato řada V_k nejdřív roste ($V_{k-1} < V_k$),potom začíná klesat ($V_{k-1} > V_k$) a má jednu nebo dvě horní hodnoty (extrémy).

Pro každé m, pokud $V_{m-1} < V_m$ (tj. pokud $n \cdot x_m^- < n \cdot E_m$), tak to potom znamená, že hodnota k_{max}, odpovídající maximu V, je $\leq m$, a proto pro všechny indexy $\leq m$ vždy víme, že pro optimální vektor x platí $x_i = \underline{x}_i$. Tyto indexy mohou být přidány do množiny I^-.

Pokud $V_m < V_{m-1}$ (tj. pokud $n \cdot x_m^- > n \cdot E_m$), tak to znamená, že hodnota k_{max}, odpovídající maximu V, je $> m$, a proto pro všechny indexy $> m$ vždy víme, že pro optimální vektor x platí $x_i = \overline{x}_i$. Tyto indexy mohou být přidány do množiny I^+.

Nakonec, pokud $V_m = V_{m-1}$ (tj. pokud $n \cdot x_m^- = n \cdot E_m$), tak to znamená, že maximum je dosaženo.

6.2.3 Případ jednoho měřícího přístroje (podmínka o podmnožině)

První výsledek

Pro případ, kdy je splněna podmínka o podmnožinách (což je ekvivalentní úloze s jedním měřícím přístrojem), můžeme roztřídit intervaly do lexikografického pořadí: $\mathbf{x}_i \leq \mathbf{x}_j$ tehdy a jen tehdy pokud $\underline{x}_i < \underline{x}_j$ nebo ($\underline{x}_i = \underline{x}_j$ a $\overline{x}_i \leq \overline{x}_j$).

Můžeme dokázat, že maximum V je vždy dosaženo pokud pro nějaké k platí, že prvních k hodnot x_i je rovno $\underline{x}_i$ a dalších $n-k$ hodnot x_i je rovno $\overline{x}_i$. Tento výsledek dokážeme sporem: pokud v maximalizujícím vektoru $x = (x_1, ..., x_n)$ nějaké $\overline{x}_i$ je před nějakým $\underline{x}_j, i < j$, potom můžeme zvýšit hodnotu V tak, že E ponecháme nezměněné - což je spor s předpokladem, že vektor x je maximalizující. Speciálně, abychom zvýšili hodnotu V, můžeme použít následující obrat: pokud $\Delta_i \leq \Delta_j$, tak vyměníme $\overline{x}_i$ s $\underline{x}_i = \overline{x}_i - 2\Delta_i$ a $\underline{x}_j$ s $\underline{x}_j + 2\Delta_i$, jinak vyměníme $\underline{x}_j$ s $\overline{x}_j = \underline{x}_j + 2\Delta_j$ a $\overline{x}_i$ s $\overline{x}_i - 2\Delta_j$.

Ve výsledku se dostáváme k následujícímu postupu: nejdříve roztřídíme intervaly $[\underline{x}_i, \overline{x}_i]$ podle lexikografického uspořádání; potom pro $k = 0, 1, ..., n$ vypočítáme hodnotu $V = M - E^2$ pro odpovídající vektory $x^{(k)} = (\underline{x}_1, ..., \underline{x}_k, \overline{x}_{k+1}, ..., \overline{x}_n)$. Když přecházíme od vektoru $x^{(k)}$ k vektoru $x^{(k+1)}$, tak se v x změní pouze jeden člen, tj. v každé sumě E a M se změní právě jeden člen.

Jak je dobrý tento algoritmus, co se týče časové složitosti? Třídění má složitost $O(n \cdot log(n))$, výpočet počátečních hodnot E a M zabere lineární čas $O(n)$. Pro každé k výpočet nových hodnot E a M potrvá konstantní počet kroků, takže celkově výpočet všech n hodnot E, M (a tedy i V) zabere lineární čas. Odtud plyne, že celkový výpočetní čas tohoto algoritmu je $O(n \cdot log(n))$.

Druhý výsledek

Případ, kdy je splněna podmínka o podmnožinách, je speciálním případem případu splněné podmínky o podmnožinách pro „zúžené“ intervaly. Výše popsaný algoritmus s lineární časovou složitosti pro výpočet $\overline{V}$ pro případ „zúžených“ intervalů splňujících podmínku o podmnožinách může být také použit pro tento případ.

6.2.4 Porovnání $O(n)$ algoritmu s $O(n \cdot log(n))$ algoritmem

Jak jsme viděli v předchozích odstavcích, v různých případech jsme se setkali jak s algoritmem s lineární časovou složitosti tak i s $O(n \cdot log(n))$ algoritmem pro výpočet stejných mezí ($\underline{V}$ nebo $\overline{V}$). Vyvstává otázka, jak moc praktické je použití $O(n)$ algoritmu? Pro jaké n jsou tyto algoritmy lepší než dříve navrhované $O(n \cdot log(n))$ algoritmy pro výpočet $\underline{V}$ nebo $\overline{V}$?

Odpověď na danou otázku obdržíme z následující úvahy. Obecně čas $O(f(n))$ znamená, že aktuální výpočetní čas je $\leq C \cdot f(n)$ pro nějakou konstantu $C > 0$. Pro známé $O(n \cdot log(n))$ algoritmy máme konstanty $C \approx 1$. Jak můžeme vidět z důkazu nově navrženého algoritmu, konstanty jsou stejné jako pro známé lineární algoritmy pro výpočet mediánu, tj. ≈ 20 (viz [3]). Odsud dostáváme, že $O(n \cdot log(n))$ algoritmus je lepší, když $log_2(n) > 20$, tj. když $n > 10^6$.

Ve většině praktických situaci se nesetkáme s případem, kdy budeme muset pracovat s miliony intervalů, ale pokud ano, tak potom lineární algoritmus pro výpočet $\underline{V}$ nebo $\overline{V}$ je skutečně rychlejší. Pro menší databáze je $O(n \cdot log(n))$ algoritmus rychlejší.

6.2.5 Případ více měřících přístrojů

V případě více měřících přístrojů můžeme ukázat, že pokud roztřídíme intervaly podle lexikografického uspořádání pro každý měřící přístroj, tak potom V nabude maxima, když z intervalů odpovídajících každému měřicímu přistrojí máme hodnoty x_i, odpovídající každému měřícímu přístroji, z řady $(\underline{x}_1, ..., \underline{x}_{k_j}, \overline{x}_{k_j+1}, ..., \overline{x}_{n_j})$, kde n_j je celkový počet intervalů odpovídajících j-tému měřícímu přístroji.

Abychom našli maximum, musíme najít hodnoty $k_1, ..., k_m$ odpovídající m měřícím přístrojům. Pro tyto hodnoty $V = M - E^2$, kde $M = \sum M_j$ a $E = \sum E_j$, kde označené E_j a M_j jsou průměry x_i a x_i^2, které jsou měření získaná použitím j-tého měřícího přístroje.

Pro každý měřící přístroj j můžeme vypočítat všech $n_j + 1$ možných hodnot E_j a M_j v lineárním čase.

Máme $\leq n^m$ kombinaci k_i a k tomu potřebujeme $O(n^m)$ kroků. Dále potřebujeme $O(n \cdot log(n))$ kroků ke třídění, $O(n)$ kroků pro výpočet počáteční hodnoty V a potom $O(n^m)$ kroků pro výpočet hodnot V, odpovídajících všem možným

kombinacím $(k_1, ..., k_m)$, a pro nalezení největší z těchto hodnot, tj. $\overline{V}$. Takže celkově potřebujeme $O(n^m)$ kroků.

6.2.6 Případ soukromí a nedetekovatelných hodnot

Tyto případy jsou podpřípady případu s platnou podmínkou o podmnožinách a také podpřípady případu „zúžených“ intervalů s platnou podmínkou o podmnožinách. Všechny známé a popsané algoritmy v Kapitolách 6.2.2 a 6.2.3 mohou být použitý i na tyto dva případy.

6.2.7 Algoritmus použitelný pro všechny popsané případy

Analýza předchozích případů

Každý může lehce zkontrolovat, že všechny případy, pro které známe efektivní algoritmus pro výpočet horní meze rozptylu, jsou jednotlivé případy pro jeden ze tří následujících případů:

- Případ „zúžených“ intervalů s podmínkou o podmnožinách, kdy žádné dva „zúžené“ intervaly $[\widetilde{x}_i - \Delta_i/n, \widetilde{x}_i + \Delta_i/n]$ nejsou navzájem vlastní podmnožiny. Pro tento případ je použitelný lineární algoritmus.

- Případ tzv. „širších zúžených intervalů“ - když každý soubor obsahující více než K „zúžených“ intervalů $\mathbf{X}_i$ má prázdný průnik: $\mathbf{X}_{i_1} \cap \cdots \cap \mathbf{X}_{i_{c+1}} = \emptyset$, $c \geq K$. Pro tento případ je vhodný $O(n \cdot log(n))$ algoritmus.

- Pro případ $m > 1$ měřících přístrojů máme k dispozici $O(n^m)$ algoritmus.

Abychom mohli připravit obecný případ, musíme popsat případ, který obsahuje tyto tři situace jako podpřípady.

Nové poznatky

Uvažujme případ určený dvěma parametry $m \geq 1$ a $K \geq 1$. V tomto případě můžeme rozdělit intervaly $\mathbf{x}_i$ do m podtříd, takových že:

- prvních $m - 1$ podtříd má vlastnost, že uvnitř každé podtřídy žádné dva „zúžené“ intervaly nejsou navzájem vlastní podmnožiny;

- poslední třída buď má stejnou vlastnost nebo má vlastnost, že každý soubor obsahující více než K „zúžených“ intervalů z této třídy má prázdný průnik.

Algoritmus pracuje v čase $O(n \cdot log(n))$, když $m = 1$, a čase $O(n^m)$, když $m > 1$. Je vidět, že všechny tři popsané výše případy jsou skutečně speciální případy situace popsané výše. Jmenovitě, případ „zúžených“ intervalů s podmínkou o podmnožinách a případ tzv. „širších zúžených intervalů“ odpovídají $m = 1$, a navíc $m = 1$ se skládá právě z těchto dvou případů.

Hlavní myšlenka algoritmu

Optimalizační výběrové postupy použité ve třech předchozích algoritmech se nezmění, pokud místo uvažování všech n intervalů budeme brát v úvahu pouze podmnožiny těchto intervalů.

Pokud podposloupnost původní posloupnosti intervalů má vlastnost, že žádné ze dvou „zúžených“ podintervalů z této podposloupnosti nejsou navzájem vlastní intervaly, tak pro tuto podposloupnost maximální hodnota V nabude v jedné z posloupnosti typu $(\underline{x}_1, ..., \underline{x}_k, \overline{x}_{k+1}, ..., \overline{x}_n)$.

Menší rozdíl je u třetího případu, protože u tohoto případu už nemluvíme o intervalu, který obsahuje průměr E - ten je jednotný pro všechny intervaly.

Teď můžeme sestavit následující:

Algoritmus

Pro $m = 1$ můžeme použit buď algoritmus pro aktuální případ nebo algoritmus pro třetí případ. Takže, abychom popsali algoritmus, je potřeba uvažovat případ, kdy $m \geq 2$.

V článku [1] je popsaný následující algoritmus. Soubor intervalů může být rozdělen do m skupin. Uvnitř každé skupiny provedeme vhodné třídění. Víme, že hodnota $\overline{V}$ bude dosažena, když pro každou z $m - 1$ podskupin máme posloupnost typu $(\underline{x}_1, ..., \underline{x}_k, \overline{x}_{k+1}, ..., \overline{x}_n)$ pro vhodné $k = k_j$. Pro poslední skupinu máme posloupnost, pro kterou:

- Pokud $\overline{x}_i \leq x_{(k)}$, potom $x_i = \underline{x}_i$;
- Pokud $x_{(k+1)} \leq \underline{x}_i$, potom $x_i = \overline{x}_i$;

pro nějaké $k = k_m$.

Podobně pro třetí případ pro každou kombinaci $(k_1, ..., k_{m-1})$ kontrola všech možných hodnot k_m potrvá čas $O(n)$. Odsud plyne, že pro všechny $\leq n^{m-1}$ možné kombinace $(k_1, ..., k_{m-1})$ potřebujeme vynaložit čas $O(n)$ - celkově $O(n^m)$.

Další konstatování: často se stává, že nepotřebujeme vědět, který interval patří do které skupiny.

V našem popisu nového algoritmu jsme předpokládali, že původní množina n intervalů může být rozdělená do m podmnožin a že víme, který interval patří do které podmnožiny. Ukazuje se, že v případě, kdy všech m podmnožin má vlastnost, že to nejsou vlastní podmnožiny (nVP vlastnost), tak není potřeba popisovat odpovídajících m podmnožin a stačí vědět, že je možné rozdělit původní množinu n intervalů do m podmnožin s touto vlastnosti.

Dále si ukážeme, jak tuto možnost můžeme popsat. Jsme schopní vytvořit následující orientovaný graf, přitom budeme vycházet z původních intervalů $\mathbf{x}_i$:

- Vrcholy grafu, jejichž je n, jsou původní intervaly.
- Hrana $\mathbf{x}_i \to \mathbf{x}_j$ jde od intervalu $[\underline{x}_i, \overline{x}_i]$ k intervalu $[\underline{x}_j, \overline{x}_j]$ tehdy a jen tehdy, pokud i-tý interval je vlastní podmnožina (vnitřku) j-tého intervalu, tj. tehdy a jen tehdy, pokud $[\underline{x}_i, \overline{x}_i] \subseteq (\underline{x}_j, \overline{x}_j)$.

Tento graf je acyklický - což znamená, že každá cesta obsahuje nejvýše n prvků. *Výškou* h tohoto grafu budeme rozumět největší délku cesty $\mathbf{x}_{i_1} \to \mathbf{x}_{i_2} \to ... \to \mathbf{x}_{i_h}$ v tomto grafu.

Následující tvrzení popisuje vztah mezi výškou grafu a počtem podskupin:

- Pokud intervaly mohou být rozdělený do m podskupin s nVP vlastností, potom výška odpovídajícího grafu je $\leq m$;
- Obráceně, pokud výška odpovídajícího grafu je m, potom můžeme účinně rozdělit původní intervaly do m podskupin s nVP vlastností.

Skutečně, pokud jsme schopní rozdělit intervaly do m podskupin s nVP vlastností, potom nemůžeme mít cestu délky $> m$; naopak, alespoň dva intervaly z cesty budou ve stejných podskupinách, a proto každé dva členy cesty jsou navzájem vlastní podmnožiny, což poruší nVP vlastnost.

Pokud máme graf výšky h, potom můžeme udělat následující:

- Vezmeme všechny prvky. které nejsou ovlivněný nikým jiným než první podskupinou. Tato skupina má nVP vlastnost.

- Když odstraníme prvky z první skupiny, můžeme znova zvážit, které prvky nejsou ovlivněný některým prvkem ve zbývajícím grafu, a z nich vytvoříme druhou podskupinu.

- atd.

Každý interval $\mathbf{x}_i$ bude přiřazen ke skupině, jejíž číslo k je největší délka cesty vedoucího do $\mathbf{x}_i$. Poněvadž výška grafu je m, rozdělíme všech n původních intervalů do m podskupin s nVP vlastnosti. Tím je naše konstatování potvrzeno.

6.2.8 Heuristický algoritmus

Informatika jako věda o informacích a jejich zpracování chápe heuristiku jako postup získání řešení problému, které však není přesné a nemusí být nalezeno v krátkém čase. Slouží však nejčastěji jako metoda rychle poskytující dostatečné a dosti přesné řešení, které však nelze obecně dokázat. Nejčastější použití heuristického algoritmu nalezneme v případech, kde není možné použít jiného lepšího algoritmu, poskytujícího přesné řešení s obecným důkazem.

V tomto odstavci popíšeme heuristický algoritmus, který je uveden v [12], pro výpočet horní hodnoty $\overline{V}$ a užitečné meze $V_{max} \geq \overline{V}$ pro rozptyl intervalu. Další problém, který budeme chtít vyřešit v souvislosti s heuristickým algoritmem, je odhad chyby v porovnání se skutečným řešením.

Nejdříve uvedeme některé vlastnosti rozptylu V.

1. Rozptyl lze vyjádřit jako funkci

$$V(x_1, ..., x_n) = \tfrac{1}{n} \cdot \sum_{i=1}^{n} (x_i - \tfrac{1}{n} \cdot \sum_{j=1}^{n} x_j)^2.$$

2. Základní vlastnosti rozptylu hodnot $x_1, ..., x_n$ je, že vyjádření $\phi(x, t) = \frac{1}{n} \cdot \sum_{i=1}^{n} (x_i - t)^2$ je minimalizováno průměrem x_i, tj. $t^* = (1/n) \cdot \sum_{i=1}^{n} x_i$. Takže nalezení $\overline{V}$ je ekvivalentní s optimalizačním problémem

$$\overline{V} = \max_{x_1,...,x_n} \min_{t} \sum_{i=1}^{n} (x_i - t)^2, \tag{6.15}$$

kde $\underline{E} \leq t \leq \overline{E}$, $x_i \in [\underline{x}_i, \overline{x}_i]$, $i = 1, ..., n$.

Dále použijeme sup-inf nerovnost pro obecnou funkci $\Phi : \mathbb{R}^p \times \mathbb{R}^q \longrightarrow \mathbb{R}$:

$$\sup_{y\in\mathbb{R}^p} \inf_{x\in\mathbb{R}^q} \Phi(x,y) \leq \inf_{x\in\mathbb{R}^q} \sup_{y\in\mathbb{R}^p} \Phi(x,y). \tag{6.16}$$

Aplikaci sup-inf nerovnosti (6.16) na $\phi(x,t) = k \cdot \sum_{i=1}^{n} (x_i - t)^2$, kde k je koeficient, který obvykle je $k = 1/n$ nebo $k = 1/(n-1)$, získáme obecnou horní mez hodnoty $\overline{V}$. Když vezmeme v úvahu, že funkce $\phi(x,t)$ je spojitá a problém (6.15) je řešen na omezené množině, potom místo sup a inf v (6.16) máme max a min. Takže dostáváme

$$\overline{V} = \max_{x_i\in[\underline{x}_i,\overline{x}_i]} \min_{t\in[\underline{E},\overline{E}]} \sum_{i=1}^{n} (x_i - t)^2 \leq \min_{t\in[\underline{E},\overline{E}]} \max_{x_i\in[\underline{x}_i,\overline{x}_i]} \sum_{i=1}^{n} (x_i - t)^2 = \min_{t\in[\underline{E},\overline{E}]} F(t),$$

kde funkce $F(t)$ je definovaná pro dané intervaly $[\underline{x}_i, \overline{x}_i]$ následujícím způsobem

$$F(t) = \max_{x_i\in[\underline{x}_i,\overline{x}_i]} \sum_{i=1}^{n} (x_i - t)^2 \text{ pro } t \in [\underline{E}, \overline{E}].$$

Pro pevnou hodnotu $t_0 \in [\underline{E}, \overline{E}]$ můžeme vyjádřit funkci $F(t)$ jako

$$F(t_0) = \sum_{i=1}^{n} \max\{(\underline{x}_i - t_0)^2, (\overline{x}_i - t_0)^2\}. \tag{6.17}$$

Z (6.17) dostáváme horní mez pro $\overline{V}$

$$\min_{t\in[\underline{E},\overline{E}]} F(t), \tag{6.18}$$

kde $F(t)$ je dáno (6.17).

Obecně, nerovnost (6.17) je přesná a díky ni můžeme spočítat „mezeru“ (tj. vzdálenost mezi $\overline{V}$ a V_{max}, které je řešením (6.18)).

Tuto „mezeru“ můžeme vyčíslit následujícím způsobem. Nechť $\widehat{x} = (\widehat{x}_1, ..., \widehat{x}_n)$ je optimální řešení pro $\overline{V}$ a nechť $F(t^*)$ je minimum (6.18), $\overline{V} = V(\widehat{x})$ je maximum $\overline{V} = \max\{V(x_1, ..., x_n) | x_i \in [\underline{x}_i, \overline{x}_i], i = 1, ..., n\}$ a $\widehat{E} = (1/n) \sum_{i=1}^{n} \widehat{x}_i$ je průměr prvků $\widehat{x}$. Potom máme

$$V(\widehat{x}) = \sum_{i=1}^{n} (\widehat{x}_i - \widehat{E})^2 = \sum_{i=1}^{n} (\widehat{x}_i - t^* + t^* - \widehat{E})^2 = \sum_{i=1}^{n} (\widehat{x}_i - t^*)^2 - (t^* - \widehat{E})^2.$$

Z $F(t^*) = \max_{x_i \in [\underline{x}_i, \overline{x}_i]} \sum_{i=1}^{n} (x_i - t^*)^2 \geq \sum_{i=1}^{n} (\widehat{x}_i - t^*)^2$ dostáváme následující nerovnost

$$\overline{V} = V(\widehat{x}) \leq F(t^*) - (t^* - \widehat{x})^2. \tag{6.19}$$

Nejlepší horní mez dostaneme, když $t^* = \widehat{E}$ (obvykle $\widehat{E}$ je neznámé).

Heuristický postup nám umožní vypočítat horní hodnotu $\overline{V}$ a mez $V_{max} \geq \overline{V}$ pro interval rozptylu.

Nechť t^* je optimální řešení (6.18) a hodnota $F(t^*) \geq \overline{V}$ je horní mez optimálního řešení. Postup je založen na následujících poznatcích:

1. Maximum $V(x)$ je vždycky dosaženo na hranici množiny, proto prvky $\widehat{x}_j$ řešení $\widehat{x}$ jsou rovné buď $\underline{x}_j$ nebo $\overline{x}_j$;
2. $\overline{V} = V(\widehat{x}) \leq F(\widehat{E})$, ale $\widehat{E}$ není známé.

Budeme předpokládat, že t^* se shoduje nejvýše s jedním z prostředních bodů intervalů.

Táto podmínka je obvykle splněna v případě, kdy všechny prostřední body intervalů jsou navzájem různé, tj. $\underline{x}_i + \overline{x}_i \neq \underline{x}_j + \overline{x}_j$ pro všechny $i \neq j$.

Obecná myšlenka postupu je následující: víme, že t^* je různé od všech středních bodů intervalů s nejvýše jednou výjimkou, a odtud máme nejvýše jeden interval k takový, že $(\underline{x}_k - t^*)^2 = (\overline{x}_k - t^*)^2$. Pokud takový interval k existuje, vybereme $\underline{x}_k$ nebo $\overline{x}_k$ tak, že průměr $E^H = (1/n) \sum_{j=1}^{n} x_j^H$ je co nejblíže k t^*.

Řešení je nalezeno jakmile máme $S(j) \in \{-1, 1\}$ pro všechny j, kde S je množina, taková, že

- $S(j) = -1$, pokud $x_j^H = \underline{x}_j$,
- $S(j) = 1$, pokud $x_j^H = \overline{x}_j$.

Prvky $S(j)$ jsou určovány během dvou kroků:

1. Pro $j = 1$ až n
   - Pokud $|\underline{x}_j - t^*| > |\overline{x}_j - t^*|$, potom $S(j) = -1$
   - Pokud $|\underline{x}_j - t^*| = |\overline{x}_j - t^*|$, potom $S(j) = 0$

- Pokud $|\underline{x}_j - t^*| < |\overline{x}_j - t^*|$, potom $S(j) = +1$

2. Pokud $S(k) = 0$ pro nějaké k, potom

   - Vypočítáme $s = \sum_{S(j)=-1} \underline{x}_j + \sum_{S(j)=+1} \overline{x}_j$;
   - Pokud $|t^* - (\underline{x}_k + s)/n| < |t^* - (\overline{x}_k + s)/n|$, potom $S(k) = -1$ jinak $S(k) = +1$.

   Po prvním kroku máme nejvýše jednu hodnotu k takovou, že $S(k) = 0$.
   Řešení x^H je

   - $x_j^H = \underline{x}_j$ pokud $S(j) = -1, j = 1, ..., n$
   - $x_j^H = \overline{x}_j$ pokud $S(j) = 1, j = 1, ..., n$

Z odhadu „mezery“ (6.19) máme, že $\overline{V}_H \leq \overline{V} \leq F(t^*) - (t^* - \widehat{E})^2 \leq F(t^*)$, a $F(t^*) - \overline{V}_H$ nám dává velikost chyby řešení, které jsme našli. Pokud $F(t^*) = \overline{V}_H$, potom můžeme být jistí, že optimální řešení bylo nalezeno.

Algoritmus

1. Vypočítáme $\underline{E} = (1/n) \sum_{i=1}^{n} \underline{x}_i$, $\overline{E} = (1/n) \sum_{i=1}^{n} \overline{x}_i$
2. Vyřešíme $F(t^*) = \min_{t \in [\underline{E}, \overline{E}]} F(t)$
3. Pro $j = 1, ..., n$

   - $|\underline{x}_j - t^*| > |\overline{x}_j - t^*|$, potom $x_j^H = \underline{x}_j; S(j) = -1$
   - $|\underline{x}_j - t^*| < |\overline{x}_j - t^*|$, potom $x_j^H = \overline{x}_j; S(j) = +1$
   - $|\underline{x}_j - t^*| = |\overline{x}_j - t^*|$, potom $S(j) = 0$

4. Pokud k existuje a $S(k) = 0$, potom

   - Vypočítáme $s = \sum_{S(j)=-1} \underline{x}_j + \sum_{S(j)=+1} \overline{x}_j$
   - Pokud $|t^* - (\underline{x}_k + s)/n| < |t^* - (\overline{x}_k + s)/n|$, potom $x_j^H = \underline{x}_j; S(k) = -1$
   - Pokud $|t^* - (\underline{x}_k + s)/n| > |t^* - (\overline{x}_k + s)/n|$, potom $x_j^H = \overline{x}_j; S(k) = +1$

5. Obdržíme řešení x^H a vypočítáme heuristickou hodnotu rozptylu

$$\overline{V}_H = \frac{1}{n} \cdot \sum_{i=1}^{n} (x_i^H - \frac{1}{n} \cdot \sum_{j=1}^{n} x_j^H)^2$$

Složitost algoritmu je $O(n)$ s výjimkou druhého kroku. Ke každé vyhodnocení funkce $F(t)$ je zapotřebí $O(n)$ operaci, které se opakují P krát. Můžeme předpokládat, že P nezávisí na n, čímž dostaneme celkovou složitost $O(n)$.

6.3 Ostatní statistiky

6.3.1 Kovariance

Mějme množinu dat (x_i, y_i), $i = 1, ..., n$, pak výběrový koeficient kovariance vypočítáme jako

$$C = \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu_x) \cdot (y_i - \mu_y),$$

kde

$$\mu_x = \frac{1}{n} \sum_{i=1}^{n} x_i;\ \mu_y = \frac{1}{n} \sum_{i=1}^{n} y_i.$$

Kovariance[1] je používána například při výpočtu korelace mezi X a Y. Pokud vezmeme v úvahu neurčitost intervalu, potom po každém měření nemáme přesné hodnoty $x_1, ..., x_n, y_1, ..., y_n$, místo toho máme intervaly $[\underline{x}_1, \overline{x}_1], ..., [\underline{x}_n, \overline{x}_n]$, $[\underline{y}_1, \overline{y}_1], ..., [\underline{y}_n, \overline{y}_n]$. Dostaneme pak různé hodnoty výběrové kovariance a záleží na tom, jaké jsou aktuální hodnoty $x_1, ..., x_n, y_1, ..., y_n$ z vnitřku těchto intervalů. Abychom mohli vzít v úvahu neurčitost intervalu, musíme být schopní popsat interval $[\underline{C}, \overline{C}]$ všech možných hodnot kovariance C.

Tím jsme dospěli k následujícímu problému: máme dané intervaly $[\underline{x}_i, \overline{x}_i]$, $[\underline{y}_i, \overline{y}_i]$, $i = 1, ..., n$, a chceme určit dolní a horní meze $\underline{C}$ a $\overline{C}$ intervalu všech možných hodnot výběrového koeficientu kovariance. Ukažme, že tento problém je NP-složitý (viz [6], [7], [8]).

Věta 2. *Problém výpočtu $\overline{C}$ ze vstupních intervalů $[\underline{x}_i, \overline{x}_i]$, $[\underline{y}_i, \overline{y}_i]$ je NP-složitý.*

[1] V dalším textu budeme pro jednoduchost používat slovo kovariance (korelace) i pro výběrový koeficient kovarianci (korelaci).

Důkaz.

1. Stejným způsobem jako v důkazu Věty 1 zredukujeme tzv. podmnožinový problém na problém výpočtu $\overline{C}$.

Pro každý případ problému o podmnožině máme: pro daných n kladných čísel chceme zjistit, zda existují $\eta_i \in \{+1, -1\}$, pro které je suma $\sum_{i=1}^{n} \eta_i \cdot s_i$ rovna 0.

Ukážeme, že problém může být zredukován na problém výpočtu $\overline{C}$, tj., že každý případ $(s_1, ..., s_n)$ problému P_0 můžeme dát do souvislosti s případem založeným na řešení výpočetního problému pro $\overline{C}$. Můžeme lehce určit zda požadované η_i existují.

Uvažujme případ odpovídající intervalům $[\underline{x}_i, \overline{x}_i] = [\underline{y}_i, \overline{y}_i] = [-s_i, s_i]$. Chceme ukázat, že pro odpovídající problém $\overline{C} = C_0$, kde jsme označili $C_0 \stackrel{def}{=} \frac{1}{n} \cdot \sum_{i=1}^{n} s_i^2$, platí $\overline{C} = C_0$ tehdy a jen tehdy, když existují η_i, pro která $\sum \eta_i \cdot s_i = 0$.

2. Nejprve ukážeme, že pro všechny případy platí $\overline{C} \leq C_0$.

Je známo, že výběrová kovariance C je ohraničená součinem $\sigma_x \cdot \sigma_y$, kde $\sigma_x = \sqrt{\sigma_x^2}$ a $\sigma_y = \sqrt{\sigma_y^2}$ jsou konečné výběrové standardní odchylky veličin x a y. V důkazu Věty 1 jsme dokázali, že rozptyl σ_x^2 hodnot $x_1, ..., x_n$ splňuje nerovnost $\sigma_x^2 \leq C_0$, stejným způsobem rozptyl σ_y^2 hodnot $y_1, ..., y_n$ splňuje nerovnost $\sigma_y^2 \leq C_0$. Odtud $C \leq \sigma_x \cdot \sigma_y \leq \sqrt{C_0} \cdot \sqrt{C_0} = C_0$. Každá možná hodnota kovariance je menší nebo rovna C_0, proto také největší z těchto hodnot, tj. $\overline{C}$, nemůže překročit C_0, tj. $\overline{C} \leq C_0$.

3. Dále ukážeme, že pokud $\overline{C} = C_0$, potom požadovaná η_i existují.

Pokud $\overline{C} = C$, tak to znamená, že pro odpovídající hodnoty x_i a y_i je kovariance C rovna C_0, tj.

$$C = C_0 = \frac{1}{n} \cdot \sum_{i=1}^{n} s_i^2.$$

Na druhé straně jsme ukázali, že ve všech případech (včetně tohoto případu) platí $C \leq \sigma_x \cdot \sigma_y \leq \sqrt{C_0} \cdot \sqrt{C_0} = C_0$. Pokud $\sigma_x < \sqrt{C_0}$, potom budeme mít $C < C_0$. Takže pokud $C = C_0$, máme $\sigma_x = \sigma_y = \sqrt{C_0}$, tj. $\sigma_x^2 = \sigma_y^2 = C_0$. V důkazu Věty 1 jsme ukázali, že v tomto případě η_i existují.

4. K dokončení důkazu Věty 2 musíme ukázat, že pokud existují požadované hodnoty η_i, potom $\overline{C} = C_0$.

V tomto případě pro $x_i = y_i = \eta_i \cdot s_i$ máme $\mu_x = \mu_y = 0$ a $x_i \cdot y_i = s_i^2$, proto

$$C = \frac{1}{n} \cdot \sum_{i=1}^{n} (x_i - \mu_x) \cdot (y_i - \mu_y) = \frac{1}{n} \cdot \sum_{i=1}^{n} s_i^2 = C_0.$$

Důkaz je dokončen.

Věta 3. *Problém výpočtu $\underline{C}$ ze vstupních intervalů $[\underline{x}_i, \overline{x}_i]$, $[\underline{y}_i, \overline{y}_i]$ je NP-složitý.*

Důkaz.

Tento důkaz je stejný jako důkaz Věty 2 s tím rozdílem, že v tomto případě použijeme druhou část nerovnosti $|C| \leq \sigma_x \cdot \sigma_y$, tj. $C \geq -\sigma_x \cdot \sigma_y$ a v poslední části důkazu položíme $y_i = -x_i$.

6.3.2 Korelace

Jak jsme se již zmínili dříve, výběrový koeficient kovariance C spočtený z množiny dat $(x_1, y_1), ..., (x_n, y_n)$ tvoří základ výběrového koeficientu korelace

$$\rho = \frac{C}{\sigma_x \cdot \sigma_y}, \tag{6.20}$$

kde $\sigma_x = \sqrt{\sigma_x^2}$ je směrodatná odchylka hodnot $x_1, ..., x_n$ a $\sigma_y = \sqrt{\sigma_y^2}$ je směrodatná odchylka hodnot $y_1, ..., y_n$.

Když máme jenom intervaly $[\underline{x}_1, \overline{x}_1]$,...,$[\underline{x}_n, \overline{x}_n]$, $[\underline{y}_1, \overline{y}_1]$,...,$[\underline{y}_n, \overline{y}_n]$, máme interval $[\underline{\rho}, \overline{\rho}]$ možných hodnot korelace. Není překvapení, že stejně jako u kovariance je problém výpočtu koncových hodnot intervalu $[\underline{\rho}, \overline{\rho}]$ také NP-složitý problém (viz [6], [7], [8]).

Věta 4. *Problém výpočtu $\overline{\rho}$ ze vstupních intervalů $[\underline{x}_i, \overline{x}_i]$, $[\underline{y}_i, \overline{y}_i]$ je NP-složitý.*

Důkaz.

1. Stejným způsobem jako v důkazu Věty 1 a Věty 2, zredukujeme podmnožinový problém na problém výpočtu $\overline{\rho}$.

Připomeňme, že pro každý případ problému o podmnožině máme: pro daných m kladných čísel $s_1, ..., s_m$ chceme zjistit, zda existují $\eta_i \in \{+1, -1\}$ takové, pro které je suma $\sum_{i=1}^{n} \eta_i \cdot s_i$ rovna 0.

Ukážeme, že podmnožinový problém může být zredukován na problém výpočtu $\overline{\rho}$, tj., že každý případ $(s_1, ..., s_m)$ problému o podmnožině P_0 můžeme dát do souvislosti s případem založeným na řešení výpočetního problému pro $\overline{\rho}$. Určíme zda požadované η_i existují.

Uvažujme případ odpovídající následujícím intervalům:

- $n = m + 2$ (v důkazech Věty 1 a Věty 2 jsme měli $n = m$);
- $[\underline{x}_i, \overline{x}_i] = [-s_i, s_i]$ a $\mathbf{y}_i = [0, 0]$ pro $i = 1, ..., m$;
- $\mathbf{x}_{m+1} = \mathbf{y}_{m+2} = [1, 1]; \mathbf{x}_{m+2} = \mathbf{y}_{m+1} = [-1, -1]$.

Jako v důkazu Věty 1 definujeme C_1 jako

$$C_1 = \sum_{i=1}^{m} s_i^2. \tag{6.21}$$

Dokážeme, že platí $\overline{\rho} = -\sqrt{\frac{2}{C_1+2}}$ tehdy a jen tehdy, pokud existují η_i, pro které $\sum \eta_i \cdot s_i = 0$.

2. Korelační koeficient je definován jako $\rho = C/\sqrt{\sigma_x^2} \cdot \sqrt{\sigma_y^2}$. Abychom našli rozsah ρ, musíme nejdříve najít rozsah C, σ_x^2 a σ_y^2.

3. Výpočet rozptylu σ_y^2 je nejlehčí, protože u y_i nemáme žádnou intervalovou neurčitost. Pro y_i máme $E_y = 0$, a proto

$$\sigma_y^2 = \frac{1}{n} \cdot \sum_{i=1}^{n} y_i^2 - (\mu_y)^2 = \frac{2}{n} = \frac{2}{m+2}. \tag{6.22}$$

4. Abychom nalezli kovarianci, využijeme známý vzorec

$$C = \frac{1}{n} \cdot \sum_{i=1}^{n} x_i \cdot y_i - E_x \cdot E_y. \tag{6.23}$$

Protože $E_y = 0$, takže druhá suma v tomto vzorci je rovna 0, takže C se rovná první sumě. V první sumě prvních m členů se rovná 0, protože pro $i = 1, ..., m$ máme $y_i = 0$. Nenulové členy jsou pro $i = m + 1$ a $i = m + 2$, takže máme

$$C = -\frac{2}{n} = -\frac{2}{m+2}. \tag{6.24}$$

5. Do vzorce pro korelaci (6.20) dosadíme vzorce (6.22) a (6.24), tím dostaneme

$$\rho = -\frac{\frac{2}{m+2}}{\sqrt{\frac{2}{m+2}} \cdot \sqrt{\sigma_x^2}} = -\sqrt{\frac{2}{(m+2) \cdot \sigma_x^2}}. \tag{6.25}$$

Korelace ρ dosáhne svého maxima $\overline{\rho}$ jen tehdy, pokud σ_x^2 nabývá své největší hodnoty $\overline{\sigma}_x^2$, odkud:

$$\overline{\rho} = -\sqrt{\frac{2}{(m+2)\cdot\overline{\sigma}_x^2}}. \tag{6.26}$$

Pokud známe $\overline{\rho}$, potom můžeme vyjádřit $\overline{\sigma}_x^2$ jako

$$\overline{\sigma}_x^2 = \frac{2}{(m+2)(\overline{\rho})^2}. \tag{6.27}$$

Požadovaná hodnota $\overline{\rho} = -\sqrt{\frac{2}{C_1+2}}$ odpovídá $\overline{\sigma}_x^2 = \frac{C_1+2}{m+2}$.

Abychom dokončili důkaz, musíme ukázat, že platí $\overline{\sigma}_x^2 = \frac{C_1+2}{m+2}$ jen tehdy, pokud existují η_i, pro které $\sum \eta_i \cdot s_i = 0$.

6. Do vyjádření σ_x^2 dosadíme vyjádření E_x:

$$\sigma_x^2 = \tfrac{1}{n} \cdot \sum_{i=1}^{n} x_i^2 - (\tfrac{1}{n} \cdot \sum_{i=1}^{n} x_i)^2.$$

Zjednodušíme vyjádření tím, že nahradíme hodnoty $n = m+2$, $x_{m+1} = 1$ a $x_{m+2} = -1$. Máme

$$\sum_{i=1}^{n} x_i = \sum_{i=1}^{m} x_i + x_{m+1} + x_{m+2} = \sum_{i=1}^{m} x_i$$

a

$$\sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{m} x_i^2 + x_{m+1}^2 + x_{m+2}^2 = \sum_{i=1}^{m} x_i^2 + 2.$$

Proto

$$\sigma_x^2 = \tfrac{1}{m+2} \cdot \sum_{i=1}^{m} x_i^2 + \tfrac{2}{m+2} - \tfrac{1}{(m+2)^2} \cdot (\sum_{i=1}^{m} x_i)^2.$$

Stejně jako v důkazu Věty 1 můžeme ukázat, že vždy $\sigma_x^2 \leq \frac{C_1+2}{m+2}$ a že $\overline{\sigma}_x^2 = \frac{C_1+2}{m+2}$ jenom pokud existují η_i pro které $\sum \eta_i \cdot s_i = 0$.

Věta je dokázána.

Věta 5. *Problém výpočtu $\underline{\rho}$ ze vstupních intervalů $[\underline{x}_i, \overline{x}_i]$, $[\underline{y}_i, \overline{y}_i]$ je NP-složitý.*

Důkaz. Tento důkaz je podobný důkazu Věty 3 s tím rozdílem, že položíme $y_{m+1} = 1$ a $y_{m+2} = -1$. V tomto případě

$$C = \tfrac{2}{m+2},$$

a proto

$$\rho = \sqrt{\tfrac{2}{(m+2)\cdot\sigma_x^2}},$$

odtud je jasné, že největší možná hodnota σ_x^2 odpovídá nejmenší možné hodnotě ρ.

Kapitola 7

Výpočetní výsledky

Popsané algoritmy jsme implementovali v systému Mathematica. Jako vzorek dat nám poslouží denní minimální a maximální cena akcií nadnárodní společnosti Hewlett Packard. Počítač, na kterém jsme prováděli testování, má následující charakteristiky: procesor o taktu 2.00 GHz a operační paměť 768 MB RAM.

Nejdřív se podíváme na výpočet dolní meze rozptylu. Tabulka 7.1 zobrazuje výsledky našeho testování. V této tabulce n značí velikost testovaného souboru dat, $\underline{V}_1$ výsledek získaný pomocí $O(n^2)$ algoritmu a t_1 je čas v sekundách, který byl potřeba k výpočtu na daném počítači; $\underline{V}_2$ výsledek získaný pomocí $O(n \cdot log(n))$ algoritmu a t_2 je odpovídající čas; $\underline{V}_3$ výsledek získaný pomocí $O(n)$ algoritmu a t_3 je odpovídající čas.

Výsledky získané třemi algoritmy jsou totožné, ale jak můžeme vidět čas výpočtu pomocí $O(n)$ algoritmu je znatelně větší v porovnání se zbývajícími dvěma algoritmy, takže použití prvního algoritmu na velký objem dat nepřipadá v úvahu, kdežto $O(n \cdot log(n)$ a $O(n)$ algoritmy nám dají výsledek v poměrně krátkém čase i pro velký objem dat.

n	$\underline{V}_1$	$\underline{V}_2$	$\underline{V}_3$	t_1	t_2	t_3
100	4.8159	4.8159	4.8159	0.0785	0.078	0.047
1000	28.2004	28.2004	28.2004	2.2820	0.422	0.141
5000	762.1010	762.1010	762.1010	38.0940	2.625	1.375
10000	633.6220	633.6220	633.6220	143.4530	5.532	4.188

Tabulka 7.1: Doby výpočtů a dosažené dolní meze pro jednotlivé algoritmy.

Dále se podíváme na výpočet horní meze rozptylu. Všechny algoritmy nejdříve aplikujeme na stejná data jako jsme použili pro výpočet dolní meze intervalu pro rozptyl. Velikost testovaných souborů dat bude $n = 20, 40, 60, 80, 100$. Tato data jsou úzké intervaly až na pár výjimek. V Tabulce 7.2 máme uvedené výsledky testování. $\overline{V}_{opt}$ značí optimální řešení, které jsme nalezli pomocí příkazu `Maximize`, který je součástí jádra programu Mathematica, $\overline{V}_1$ - výsledek $O(n)$ algoritmu, $\overline{V}_2$ - výsledek $O(n \cdot log(n))$ algoritmu, $\overline{V}_3$ - výsledek $O(n)$ algoritmu, $\overline{V}_H$ - výsledek heuristického algoritmu, t_{opt}, t_1, t_2, t_3, t_h - časy v sekundách jednotlivých postupů.

n	$\overline{V}_{opt}$	$\overline{V}_1$	$\overline{V}_2$	$\overline{V}_3$	$\overline{V}_H$
20	1.65592	1.66187	1.66187	1.66187	1.66187
40	4.33791	4.33791	4.33791	4.33791	4.33791
60	11.30600	11.30600	11.30600	11.30600	11.30600
80	9.50121	9.50121	9.50121	9.50121	9.50121
100	8.15335	8.15335	8.15335	8.15335	8.15335
n	t_{opt}	t_1	t_2	t_3	t_H
20	4.906	0.078	0.047	0.016	0.031
40	56.312	0.094	0.047	0.032	0.047
60	208.422	0.078	0.047	0.032	0.047
80	493.125	0.109	0.062	0.032	0.062
100	1061.390	0.219	0.063	0.047	0.078

Tabulka 7.2: Doby výpočtů a dosažené horní meze pro jednotlivé algoritmy.

Je vidět, že optimální řešení ve většině případů se shoduje s výsledky algoritmů i přesto, že v testovaných datech je několik intervalů, které nesplňují podmínku o úzkých intervalech. Povzbuzeni slibnými výsledky pro menší počet intervalů, rozhodli jsme se zkusit nalézt $\overline{V}$ i pro mnohem rozsáhlejší data, která byla užita pro výpočet $\underline{V}$. K výpočtu jsme užili heuristický algoritmus a $O(n)$ algoritmus. Výsledky a časy jednotlivých algoritmů jsou uvedeny v Tabulce 7.3. Pro $n = 100$, 1000 a 5000 jsme dostali tytéž výsledky, pro $n = 10000$ se získané výsledky malinko liší; spočtený rozptyl je menší pro $O(n)$ algoritmus.

Na závěr zkusíme aplikovat heuristický algoritmus na obecný případ a vý-

n	$\overline{V}_{O(n)}$	$\overline{V}_H$	$t_{O(n)}$	t_H
100	8.15335	8.15335	0.140	0.063
1000	38.96910	38.96910	0.141	1.594
5000	855.44400	855.44400	0.765	40.046
10000	708.57300	708.58100	2.469	144.219

Tabulka 7.3: Dosažené horní meze a časy pro jednotlivé algoritmy.

n	$\overline{V}_{opt}$	$\overline{V}_H$	P	t_{opt}	t_H
20	917.739	969.92	5.69	8.219	0.094
40	1044.680	1189.48	13.86	257.750	0.078
60	951.783	1069.95	12.42	248.657	0.078
80	1090.573	1291.01	18.41	643.078	0.140
100	1158.200	1306.56	12.81	1268.920	0.157

Tabulka 7.4: Dosažené horní meze pro jednotlivé algoritmy, absolutní procentuální chyby výsledků v porovnání s optimálním řešením a časy pro jednotlivé postupy.

sledky porovnáme s optimálním řešením. Testována data $[\underline{x}_i, \overline{x}_i]$ náhodně vygenerujeme následujícím způsobem:

Funkce $Random()$ generuje náhodná čísla z intervalu $[0, 1]$. Zvolíme reálné konstanty x_{min}, x_{max}, d_{min}, d_{max}, kde $x_{min} < x_{max}$ a $d_{min} < d_{max}$. V našem příkladě jsme zvolili $x_{min} = 80$, $x_{max} = 100$, $d_{min} = 5$, $d_{max} = 50$. Dále spočteme hranice intervalů pomocí následujícího cyklu, kde n je počet generovaných intervalů:

Pro $i = 1, ..., n$ nechť:

$$\widehat{x}_i = x_{min} + Random() * (x_{max} - x_{min})$$
$$\delta x_i = d_{min} + Random() * (d_{max} - d_{min})$$
$$\underline{x}_i = \widehat{x}_i - \delta x_i$$
$$\overline{x}_i = \widehat{x}_i + \delta x_i$$

Tabulka 7.4 ukazuje výsledky algoritmu, kde P značí absolutní procentuální chybu výsledků v porovnání s optimálním řešením; t_{opt}, t_H - časy v sekundách jednotlivých postupů.

Vidíme, že v případě obecných intervalů již výsledky nejsou zdaleka tak přesné jako v případě intervalů speciálního typu uvažovaných výše.

Kapitola 8

Závěr

Statistická analýza intervalových dat se stává velice důležitou částí moderní matematiky. Proto je velice vhodné použití co nejefektivnějších statistických postupů. Většina tradičních statistických vzorců $y = f(x_1, ..., x_n)$, tj. vzorce pro výpočet výběrové střední hodnoty či výběrového rozptylu atd. jsou založený na zjednodušeném předpokladu, že známe přesné hodnoty $x_1, ..., x_n$ veličiny, která nás zajímá. Ve skutečnosti tyto hodnoty pocházejí z měření nebo odhadů, což nám neposkytuje absolutně přesná data. Proto je žádoucí, abychom zjistili jak nepřesnost dat ovlivňuje výsledek statistické analýzy.

V mnoha případech ze skutečného života známe jenom horní mez Δ pro nepřesnost měření. V tomto případě máme výsledek měření $\widetilde{x}$ požadované veličiny. Jediná informace, kterou máme o aktuální (neznámé) hodnotě x veličiny je, že tato hodnota musí patřit do intervalu $[\widetilde{x} - \Delta, \widetilde{x} + \Delta]$. Proto je žádoucí upravit tradiční statistické vzorce $f(x_1, ..., x_n)$ pro případ, kdy vstupní hodnoty jsou intervalového typu, tj. známe n intervalů $\mathbf{x}_1, ..., \mathbf{x}_n$ a chceme vypočítat rozsah možných hodnot požadované charakteristiky $f(x_1, ..., x_n)$, kde $x_i \in \mathbf{x}_i$.

V obecném případě je tento výpočetní problém NP-složitý. Existují praktické situace, kdy je možné navrhnout efektivní algoritmy pro výpočet těchto charakteristik.

V této diplomové práci jsme popsali a opravili některé známe postupy a tak jsme získali algoritmy s výpočetní složitosti $O(n \cdot log(n))$, což je srovnatelné se složitostí třídění daných hodnot, v některých situacích dokonce máme postupy s lineární složitosti $O(n)$ - nejmenší možná výpočetní složitost pro jakýkoliv algo-

ritmus zpracovávající n vstupních hodnot $x_1, ..., x_n$. Dále máme efektivní algoritmus pro výpočet rozptylu v některých praktických situacích a rychlý heuristický algoritmus pro obecný případ, který nám poskytuje výsledek s malou chybou.

Pro dolní mez $\underline{V}$ rozptylu V máme dva algoritmy o složitosti $O(n)$ a $O(n \cdot log(n))$ pro obecný případ.

Problém výpočtu horní meze $\overline{V}$ je obecně NP-složitý. K dispozicí máme $O(n^2)$ algoritmus pro případ úzkých intervalů (žádné dva intervaly nemají společný průnik), případ „tzv. širších intervalů“ (kdy pro nějaké K, žádná skupina o K intervalech nemá společný bod) a případ soukromí (kdy každé dva intervaly se buď shodují nebo se neprotínají). V diplomové práci uvádíme algoritmy s lineární složitosti pro případ úzkých intervalů a pro případ soukromí, dále $O(n \cdot log(n))$ algoritmus pro „tzv. širší intervaly“. Pro případ jednoho měřícího přistroje máme algoritmus s lineární složitostí a $O(n^m)$ složitý algoritmus pro m měřících přístrojů. Na závěr uvádíme heuristický algoritmus se složitosti $O(n)$, kterým nám poskytuje rychlý výsledek, ale s malou chybou.

Postupy jsme aplikovali na reálná finanční data velkého rozsahu a ilustrovali na nich výpočetní složitost jednotlivých algoritmů.

Literatura

[1] Aló R., Beheshti M., Xiang G.: *Computing Variance under Interval Uncertainty: A New Algorithm and its Potential Application to Privacy in Statistical Databases*. Proceedings of the International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems IPMU'06, Paris, France, July 2-7, 2006, str. 810-816.

[2] Antoch J., Brzezina M., Miele R.: *A note on variability of interval data*. Computational Statistics, 2010, vol. 25, issue 1, str. 143-153.

[3] Cormen Th. H., Leiserson C. E., Rivest R. L. a Stein C.: *Introduction to Algorithms*. MIT Press, Cambridge, MA, 2001.

[4] Dantsin E., Kreinovich V., Wolpert A., Xiang G.: *Population Variance under Interval Uncertainty: A New Algorithm*. Reliable Computing, Vol. 12, No. 4. (8 August 2006), str. 273-280.

[5] Granvilliers L., KreinovichV., Müller N.: *Novel Approaches to Numerical Software with Result Verification*. In: R. Alt, A. Frommer, R. B. Kearfott, W. Luther(eds.), Numerical Software with Result Verification, (International Dagstuhl Seminar, Dagstuhl Castle, Germany, Jan. 19-24, 2003), Springer Lectures Notes in Computer Science, 2004, Vol. 2991, str. 274-305.

[6] Ferson S., Ginzburg L., Kreinovich V., Aviles M.: *Exact Bounds on Sample Variance of Interval Data*. Abstracts of 2002 SIAM Workshop on Validated Computing, Toronto, Canada, May 23-25, 2002, str. 67-69.

[7] Ferson S., Ginzburg L., Kreinovich V., Longpré L. a Aviles M.: *Computing Variance for Interval Data is NP-Hard*. ACM SIGACT News, 2002, Vol. 33, No. 2, str. 108-118.

[8] Ferson S., Ginzburg L., Kreinovich V., Longpré L. a Aviles M.: *Exact Bounds on Finite Populations of Interval Data*. Reliable Computing, 2005, Vol. 11, No. 3, str. 207–233.

[9] Kreinovich V., Longpre L.: *Computational Complexity and Feasibility Of Data Processing and Interval Computations, with Extension to Cases When We Have Partial Information about Probabilities*. In: V. Brattka, M. Schroeder, K. Weihrauch, N. Zhong, Proceedings of the Conference on Computability and Complexity in Analysis CCA'2003, Cincinnati, Ohio, USA, Aug. 28-30, 2003, str. 19-54.

[10] Kreinovich V., Nguyen H. T., Wu B.: *On-Line Algorithms for Computing Mean and Variance of Interval Data, and Their Use in Intelligent Systems*. Information Sciences, 2007, Vol. 177, No. 16, str. 3228-3238.

[11] Kreinovich V., Xiang G., Starks S. A., Longpré L., Ceberio M., Araiza R., Beck J., Kandathi R., Nayak A., Torres R., Hajagos J.: *Towards combining probabilistic and interval uncertainty in engineering calculations: algorithms for computing statistics under interval uncertainty, and their computational complexity*. Reliable Computing, 2006, Vol. 12, No. 6, str. 471-501.

[12] Spadoni M., Stefanini L.: *Computing the variance of interval and fuzzy data*. Fuzzy Sets and Systems, Volume 165 Issue 1, February, 2011.

[13] Töpfer P.: *Algoritmy a programovací techniky*. Prometheus, Praha 1995.

[14] Xiang G.: *Fast algorithm for computing the upper endpoint of sample variance for interval data: case of sufficiently accurate measurements*. Reliable Computing, 2006, Vol. 12, No. 1, str. 59-64.

[15] Xiang G., Ceberio M., Kreinovich V.: *Computing Population Variance and Entropy under Interval Uncertainty: Linear-Time Algorithms*. University of Texas at El Paso, Department of Computter Science, Technical Report UTEP-CS-06-28b, November 2006.

Přílohy

Prvních 100 testovaných intervalů:

{{40.2,40.67},{40.12,40.73},{40.85,41.55},{40.53,41.18},{40.66,41.28},{40.52,41.04},
{41.06,41.8},{40.33,41.35},{40.21,40.78},{40.2,40.88},{40.5,41.08},{40.96,41.48},
{40.8,41.48},{37.6,42.01},{42.11,42.83},{42.33,43.28},{42.26,43.14},{41.51,42.28},
{41.35,41.84},{41.65,42.1},{41.28,42.13},{40.72,41.83},{40.1,41.32},{40.34,41.19},
{41.28,42.15},{41.36,41.87},{41.26,41.9},{41.76,42.4},{41.95,42.71},{41.94,43.05},
{42.06,43.45},{43.12,43.86},{42.78,43.52},{42.84,43.84},{42.95,43.83},{42.3,42.8},
{42.08,43.25},{42.57,43.72},{47.27,48.55},{48.35,48.8},{48.27,48.79},{48.36,49.12},
{47.54,48.18},{48.06,48.68},{48.24,48.81},{48.15,49.39},{47.99,48.94},{47.76,48.24},
{47.61,48.42},{47.08,47.53},{46.47,47.48},{46.43,47.05},{45.9,46.6},{45.13,45.72},
{45.36,46.69},{46.58,46.98},{46.57,47.28},{46.88,47.83},{46.66,47.59},{46.83,47.64},
{45.76,46.79},{46.08,46.48},{46.08,46.42},{45.61,46.4},{45.31,45.84},{45.27,45.71},
{45.2,46.06},{44.57,45.05},{44.71,45.39},{44.18,44.96},{43.4,44.22},{43.01,43.77},
{42.22,43.49},{41.84,42.38},{42.16,42.46},{42.25,42.62},{41.94,42.49},{41.47,41.98},
{41.29,41.92},{41.4,41.82},{41.74,42.1},{41.72,42.1},{41.68,42.19},{41.,42.01},
{41.22,41.61},{41.27,41.86},{41.53,42.18},{42.19,42.66},{42.2,42.78},{42.16,42.72},
{42.09,43.21},{42.51,43.25},{42.52,43.27},{42.42,43.17},{42.33,42.94},{41.8,42.43},
{42.08,42.76},{43.03,43.4},{43.48,44.35},{43.11,44.44}}

Algoritmus se složitosti $O(n^2)$ pro výpočet $\underline{V}$:

Vstupem je soubor *test.xls*, který obsahuje testované intervaly a výstupem programu je $\underline{V}$.

$a = \texttt{Partition}[\texttt{Flatten}[\texttt{Import}[\texttt{"test".xls}]], 2];$

$b = \texttt{Flatten}[a];$

$c = \texttt{Sort}[b, \texttt{LessEqual}];$

$u = \texttt{Total}[\texttt{Part}[a, \texttt{All}, \{1, 2\}]]/\texttt{Length}[a];$

$n = \{\};$

$\texttt{Map}[\texttt{If}[u[[1]] \leq c[[\#]] \leq u[[2]] \| u[[1]] \leq c[[\#+1]] <= u[[2]] \| u[[1]] >= c[[\#]] \&\& u[[2]] \leq c[[\#+1]], \texttt{AppendTo}[n, \#]]\&, \texttt{Range}[\texttt{Length}[c]-1]; i = \texttt{NestList}[\#\&, \{\}, \texttt{Length}[n]-1];$

$\texttt{For}[k = 1, k \leq \texttt{Length}[n], k\texttt{++}, \texttt{Map}[\texttt{If}[a[[\#, 1]] \geq c[[n[[k]]+1]], \texttt{AppendTo}[i[[k]], \#]]\&, \texttt{Range}[\texttt{Length}[a]]]];$

$j = \texttt{NestList}[\#\&, \{\}, \texttt{Length}[n] - 1];$

$\texttt{For}[k = 1, k \leq \texttt{Length}[n], k\texttt{++}, \texttt{Map}[\texttt{If}[a[[\#, 2]] \leq c[[n[[k]]]], \texttt{AppendTo}[j[[k]], \#]]\&, \texttt{Range}[\texttt{Length}[a]]]];$

$\texttt{Sk} = \texttt{Map}[\texttt{Total}[a[[\#, 1]]]\&, i] + \texttt{Map}[\texttt{Total}[a[[\#, 2]]]\&, j];$

$\texttt{Nk} = \texttt{Map}[\texttt{Length}[i[[\#]]] + \texttt{Length}[j[[\#]]]\&, \texttt{Range}[\texttt{Length}[n]]];$

$\texttt{Ek} = \texttt{Map}[\texttt{If}[\texttt{Nk}[[\#]]\texttt{==}0, 0, \texttt{Sk}[[\#]]/\texttt{Nk}[[\#]]]\&, \texttt{Range}[\texttt{Length}[n]]];$

$K = \texttt{Select}[\texttt{Map}[\texttt{If}[c[[n[[\#]]]] \leq \texttt{Ek}[[\#]] \leq c[[n[[\#]] + 1]], \#]\&, \texttt{Range}[\texttt{Length}[\texttt{Ek}]]], \# \geq 1\&];$

$V = \texttt{Min}[1/\texttt{Length}[a] * (\texttt{Total}[\texttt{Map}[(a[[i[[\#]], 1]] - \texttt{Ek}[[\#]])\texttt{\^{}}2\&, K], \{-1\}] + \texttt{Total}[\texttt{Map}[(a[[j[[\#]], 2]] - \texttt{Ek}[[\#]])\texttt{\^{}}2\&, K], \{-1\}])];$

$\texttt{V0} = \texttt{Select}[\texttt{Map}[\texttt{If}[\texttt{Ek}[[\#]]\texttt{==}0, V = 0]\&, \texttt{Range}[\texttt{Length}[\texttt{Ek}]]], \#\texttt{==}0\&];$

$\texttt{Min}[V, \texttt{V0}]$

Algoritmus se složitosti $O(n \cdot log(n))$ pro výpočet $\underline{V}$:

Vstupem je soubor *test.xls*, který obsahuje testované intervaly a výstupem programu je $\underline{V}$.

$a =$ Partition[Flatten[Import["test".xls]], 2];

$b =$ Flatten[a];

$c =$ Sort[b, LessEqual];

suma1 $= 0$; suma2 $= 0$; suma3 $= 0$; suma4 $= 0$;

kminus $= 1$; kplus $=$ Length[c]; kmid $=$ IntegerPart[(kminus $+$ kplus)$/2$];

Map[If[$a[[\#, 2]] \leq c[[$kmid$]]$, suma1 $= (c[[$kmid$]] - a[[\#, 2]]) +$ suma1]&,

Range[Length[a]]];

Map[If[$a[[\#, 1]] >= c[[$kmid $+ 1]]$, suma2 $= (a[[\#, 1]] - c[[$kmid$]]) +$ suma2]&,

Range[Length[a]]];

Map[If[$a[[\#, 2]] \leq c[[$kmid$]]$, suma3 $= (c[[$kmid $+ 1]] - a[[\#, 2]]) +$ suma3]&,

Range[Length[a]]];

Map[If[$a[[\#, 1]] >= c[[$kmid $+ 1]]$, suma4 $= (a[[\#, 1]] - c[[$kmid $+ 1]]) +$ suma4]&,

Range[Length[a]]];

While[suma1 $>$ suma2$\|$suma3 $<$ suma4, suma1 $= 0$; suma2 $= 0$; suma3 $= 0$; suma4 $=$

0; Map[If[$a[[\#, 2]] \leq c[[$kmid$]]$, suma1 $= (c[[$kmid$]] - a[[\#, 2]]) +$ suma1]&,

Range[Length[a]]];

Map[If[$a[[\#, 1]] >= c[[$kmid $+ 1]]$, suma2 $= (a[[\#, 1]] - c[[$kmid$]]) +$ suma2]&,

Range[Length[a]]];

Map[If[$a[[\#, 2]] \leq c[[$kmid$]]$, suma3 $= (c[[$kmid $+ 1]] - a[[\#, 2]]) +$ suma3]&,

Range[Length[a]]];

Map[If[$a[[\#, 1]] >= c[[$kmid $+ 1]]$, suma4 $= (a[[\#, 1]] - c[[$kmid $+ 1]]) +$ suma4]&,

Range[Length[a]]]; If[suma1 $\leq$ suma2$\|$suma3 $<$ suma4, kminus $=$ kmid $+ 1$];

If[suma1 $>$ suma2$\|$suma3 $\geq$ suma4, kplus $=$ kmid-1]; kmid $=$ IntegerPart[(kminus$+$

kplus)$/2$]]

suma1k $= 0$; suma2k $= 0$;

$i =$ Length[Select[Map[If[$a[[\#, 1]] \geq c[[$kmid$+1]]$, suma1k $= a[[\#, 1]]+$suma1k]&,

Range[Length[a]]], InexactNumberQ[$\#$]&]];

$j =$ Length[Select[Map[If[$a[[\#, 2]] \leq c[[$kmid$]]$, suma2k $= a[[\#, 2]] +$ suma2k]&,

```
Range[Length[a]]], InexactNumberQ[#]&]];
Sk = suma1k + suma2k;
Nk = i + j;
If[Nk == 0, V = 0, Rk = Sk/Nk;
sumaV1 = 0; sumaV2 = 0;
Map[If[a[[#, 1]]>=c[[kmid + 1]], sumaV1 = (a[[#, 1]] - Rk)^2 + sumaV1]&,
Range[Length[a]]];
Map[If[a[[#, 2]]<=c[[kmid]], sumaV2 = (a[[#, 2]] - Rk)^2 + sumaV2]&,
Range[Length[a]]];
V = 1/Length[a] * (sumaV1 + sumaV2)];
V
```

Algoritmus se složitosti $O(n)$ pro výpočet $\underline{V}$:

Vstupem je soubor *test.xls*, který obsahuje testované intervaly a výstupem programu je $\underline{V}$.

```
Ii = Flatten[Import["test.xls"]];
Icko = {{}, {}};
Map[If[EvenQ[#], AppendTo[Icko[[2]], Ii[[#]]], AppendTo[Icko[[1]], Ii[[#]]]]&,
Range[Length[Ii]]]; velikost = Length[Ii]/2;
Pminus = {{}, {}}; Pplus = {{}, {}}; Iminus = {{}, {}}; Iplus = {{}, {}}; nminus = 0;
nplus = 0; Nminus = 0; Nplus = 0; Sminus = 0; Splus = 0;
mplus = Min[Pplus];
While[Length[Flatten[Icko]] > 0, Pminus = {{}, {}}; Pplus = {{}, {}}; nminus = 0;
nplus = 0; eminus = 0; eplus = 0; m = Median[Flatten[Icko]];
Map[If[Icko[[2, #]]<=m, AppendTo[Pminus[[2]], Icko[[2, #]]]]&, Range[Length[Icko[[2]]]]];
Map[If[Icko[[1, #]]<=m, AppendTo[Pminus[[1]], Icko[[1, #]]]]&, Range[Length[Icko[[1]]]]];
Map[If[Icko[[2, #]] > m, AppendTo[Pplus[[2]], Icko[[2, #]]]]&, Range[Length[Icko[[2]]]]];
Map[If[Icko[[1, #]] > m, AppendTo[Pplus[[1]], Icko[[1, #]]]]&, Range[Length[Icko[[1]]]]];
mplus = Min[Pplus]; eminus = Sminus + Total[Pminus[[2]]]; eplus = Splus +
Total[Pplus[[1]]]; nminus = Nminus + Length[Pminus[[2]]];
nplus = Nplus + Length[Pplus[[1]]]; If[(nminus + nplus) == 0, Ee = m,
Ee = (eminus + eplus)/(nminus + nplus)];
If[Ee < m, Iplus = Map[Join[Iplus[[#]], Pplus[[#]]]&, Range[2]]; Splus = eplus;
Icko = Pminus; Nplus = nplus];
If[Ee > mplus, Iminus = Map[Join[Iminus[[#]], Pminus[[#]]]&, Range[2]];
Sminus = eminus; Icko = Pplus; Nminus = nminus];
If[m ≤ Ee ≤ mplus, Iminus = Map[Join[Iminus[[#]], Pminus[[#]]]&, Range[2]]; Iplus =
Map[Join[Iplus[[#]], Pplus[[#]]]&, Range[2]]; Icko = {{}, {}};
Sminus = eminus; Splus = eplus; Nminus = nminus; Nplus = nplus]; ];
x = Join[Iminus[[2]], Iplus[[1]]];
V = Total[Map[(x[[#]] − Ee)^2&, Range[Length[x]]]]/velikost
```

Algoritmus se složitosti $O(n^2)$ pro výpočet $\overline{V}$:

Vstupem je soubor *test.xls*, který obsahuje testované intervaly a výstupem programu je $\overline{V}$.

```
a = Partition[Flatten[Import["test.xls"]], 2];
n = Length[a];
b = Flatten[Join[a, {-Infinity, Infinity}]];
c = Sort[b, LessEqual];
u = Total[Part[a, All, {1, 2}]]/Length[a];
k = {};
Map[If[u[[1]] ≤ c[[#]] ≤ u[[2]]||u[[1]] ≤ c[[#+1]]<=u[[2]]||u[[1]]>=c[[#]]&&u[[2]] ≤
c[[# + 1]], AppendTo[k, #]]&, Range[Length[c]] − 1];

x1 = {};
For[l = 1, l ≤ Length[k], l++, Map[
If[c[[k[[l]]+1]] < ((a[[#, 1]]+a[[#, 2]])/2−(a[[#, 2]]−(a[[#, 1]]+a[[#, 2]])/2)/n),
AppendTo[x1, {a[[#, 2]]}]]||If[c[[k[[l]]]] > ((a[[#, 1]] + a[[#, 2]])/2 + (a[[#, 2]] −
(a[[#, 1]] + a[[#, 2]])/2)/n), AppendTo[x1, {a[[#, 1]]}]]||
If[c[[k[[l]] + 1]] ≥ ((a[[#, 1]] + a[[#, 2]])/2 − (a[[#, 2]] − (a[[#, 1]] + a[[#, 2]])/2)/n)&&
c[[k[[l]]]] ≤ ((a[[#, 1]] + a[[#, 2]])/2 + (a[[#, 2]] − (a[[#, 1]] + a[[#, 2]])/2)/n),
AppendTo[x1, {a[[#, 1]], a[[#, 2]]}]]&,
Range[Length[a]]]];
x = Partition[x1, Length[a]];
y = Map[Tuples[x[[#]]]&, Range[Length[x]]];
z = Total[y, {3}]/Length[a];
rozptyly = {};
For[g = 1, g ≤ Length[z], g++, Map[If[u[[1]]<=z[[g, #]] ≤ u[[2]],
AppendTo[rozptyly, 1/Length[y[[g, #]]] ∗ Total[(y[[g, #]] − z[[g, #]])^2]]]&,
Range[Length[z[[g]]]]]];
HorniV = Max[rozptyly]
```

Algoritmus se složitosti $O(n \cdot log(n))$ pro výpočet $\overline{V}$:

Vstupem je soubor *test.xls*, který obsahuje testované intervaly a výstupem programu je $\overline{V}$.

```
a = Partition[Flatten[Import["test.xls"]], 2];
n = Length[a];
b = Map[Total[a[[#]]]/2&, Range[Length[a]]];
c = NestList[#&, {}, Length[b] - 1];
Map[AppendTo[c[[#]], b[[#]]]&, Range[Length[b]]];
d = Map[AppendTo[c[[#]], #]&, Range[Length[b]]];
e = SortBy[d, First];
index = Flatten[Map[Append[{}, e[[#, 2]]]&, Range[Length[e]]]];
a1 = Flatten[Map[Append[{}, a[[#]]]&, index], 1];
e0 = 0;
m0 = 0;
E0 = Last[Map[e0+=a1[[#, 2]]&, Range[n]]]/n;
M0 = Last[Map[m0+=(a1[[#, 2]])^2&, Range[n]]]/n;
f = NestList[#&, {}, n];
f[[1]] = E0;
Map[AppendTo[f[[# + 1]], f[[#]] + 1/n * a1[[#, 1]] - 1/n * a1[[#, 2]]]&, Range[n]];
t = NestList[#&, {}, n];
t[[1]] = M0;
Map[AppendTo[t[[# + 1]], t[[#]] + 1/n * (a1[[#, 1]])^2 - 1/n * (a1[[#, 2]])^2]&, Range[n]];
f = Flatten[f];
t = Flatten[t];
S = {};
M = {};
Map[AppendTo[S, f[[#]]]&, Range[n + 1]];
Map[AppendTo[M, t[[#]]]&, Range[n + 1]];
V = Max[M - S^2]
```

Algoritmus se složitosti $O(n)$ pro výpočet $\overline{V}$:

Vstupem je soubor *test.xls*, který obsahuje testované intervaly a výstupem programu je $\overline{V}$.

```
a = Partition[Flatten[Import["test.xls"]], 2];
n = Length[a];
Iminus = {}; Iplus = {}; Icko = {}; G = {};
Map[If[a[[#, 1]]==a[[#, 2]], AppendTo[G, #], AppendTo[Icko, #]]&, Range[n]];
I1 = Icko;
Sminus = 0; Splus = 0; e = 0; eminus = 0; eplus = 0; Eminus = 0; Eplus = 0;
Map[e+=a[[#, 1]]&, G];
Pminus = {}; Pplus = {};
b1 = Map[Total[a[[#]]]/2&, Range[n]];
Iplus1 = {}; Iminus2 = {};

While[Length[Icko] > 0, eminus = 0; eplus = 0; Pminus = {}; Pplus = {};
b = Map[Total[a[[#]]]/2&, Icko];
 m = Median[b];
p = Min[Flatten[Position[b, Max[Select[b, #<=m&]]]]];
p1 = Intersection[Icko, Flatten[Position[b1, Max[Select[b1, #<=m&]]]]];
Map[If[b1[[#]] ≤ b[[p]], AppendTo[Pminus, #]]&, Icko];
Map[If[b1[[#]] > b[[p]], AppendTo[Pplus, #]]&, Icko];
p2 = Min[p1];
p3 = Select[p1, # ≠ p2&];
Pminus = Complement[Pminus, p3];
Pplus = Flatten[AppendTo[Pplus, p3]];
eminus = Eminus + Sum[a[[j, 1]], {j, Pminus}];
eplus = Eplus + Sum[a[[i, 2]], {i, Pplus}];
e1 = e + eminus + eplus;

y = (n * (a[[p2, 1]] + a[[p2, 2]]) − a[[p2, 2]] + a[[p2, 1]])/2;
If[Length[Icko]==1, Iplus1 = Join[Iplus, Icko]; Iminus2 = Join[Iminus, Icko];
```

```
Icko = {}, If[y < e1, Iminus = Join[Iminus, Pminus]; Icko = Pplus; Eminus =
eminus,
If[y > e1, Icko = Pminus; Iplus = Join[Iplus, Pplus]; Eplus = eplus]]];
If[y == e1, Iminus = Join[Iminus, Pminus]; Iplus = Join[Iplus, Pplus]; Icko =
{}; x = 1]]
V1 = 0;
If[x==1, E1 = 0;
Map[E1+=a[[#, 1]]&, Iminus];
Map[E1+=a[[#, 2]]&, Iplus];
V1 = 1/n * (Total[(Map[a[[#, 2]]&, Iplus] - E1/n)^2] +
Total[(Map[a[[#, 1]]&, Iminus] - E1/n)^2])];

Iminus1 = Complement[I1, Iplus1];
E2 = 0;
Map[E2+=a[[#, 1]]&, Iminus1];
Map[E2+=a[[#, 2]]&, Iplus1];
V2 = 1/n * (Total[(Map[a[[#, 2]]&, Iplus1] - E2/n)^2] +
Total[(Map[a[[#, 1]]&, Iminus1] - E2/n)^2]);

Iplus2 = Complement[I1, Iminus2];
E3 = 0;
Map[E3+=a[[#, 1]]&, Iminus2];
Map[E3+=a[[#, 2]]&, Iplus2];
V3 = 1/n * (Total[(Map[a[[#, 2]]&, Iplus2] - E3/n)^2] +
Total[(Map[a[[#, 1]]&, Iminus2] - E3/n)^2]);
V = Max[V1, V2, V3]
```

Heuristický algoritmus:

Vstupem je soubor *test.xls*, který obsahuje testované intervaly a výstupem programu je $\overline{V}$.

$a =$ `Partition[Flatten[Import["test.xls"]], 2];`

$n =$ `Length[`a`];`

`SH` $=$ `Total[Part[`a`, All, {1, 2}]]/`n`;`

$t = t$`/.Last[FindMinimum[Sum[Max[(`a`[[`i`, 1]] - `t`)^2, (`a`[[`i`, 2]] - `t`)^2], {`i`, `n`}], {`t`},`
`$MachinePrecision]];`

$x =$ `NestList[#&, {}, `$n - 1$`];`

$S =$ `NestList[#&, {}, `$n - 1$`];`

$i = \{\}; j = \{\}; k = \{\};$

`Map[If[Abs[`a`[[#, 1]] - `t`] > Abs[`a`[[#, 2]] - `t`], `x`[[#]] = `a`[[#, 1]]; `S`[[#]] = -1;`
`AppendTo[`i`, #]]&, Range[`n`]];`

`Map[If[Abs[`a`[[#, 1]] - `t`]<=Abs[`a`[[#, 2]] - `t`], `x`[[#]] = `a`[[#, 2]]; `S`[[#]] = 1;`
`AppendTo[`j`, #]]&, Range[`n`]];`

`Map[If[Abs[`a`[[#, 1]] - `t`]==Abs[`a`[[#, 2]] - `t`], `S`[[#]] = 0; AppendTo[`k`, #]]&, Range[`n`]];`

$s =$ `Total[Map[`a`[[#, 1]]&, `i`]] + Total[Map[`a`[[#, 2]]&, `j`]];`

`Map[If[Abs[`$t - (a[[\#, 1]] + s)/n$`] < Abs[`$t - (a[[\#, 2]] + s)/n$`], `x`[[#]] = `a`[[#, 1]];`
`S[[#]] = -1]&, `k`];`

`Map[If[Abs[`$t - (a[[\#, 1]] + s)/n$`]>=Abs[`$t - (a[[\#, 2]] + s)/n$`], `x`[[#]] = `a`[[#, 2]];`
`S[[#]] = 1]&, `k`];`

$V = 1/n *$ `Sum[(`x`[[`i`]] - 1/`$n *$ `Sum[`x`[[`i`]], {`i`, `n`}])^2, {`i`, `n`}]`