Katedra
aplikované matematiky

Počítačová cvičení
Škola matematického modelování
2019

Petr Beremlijski, Rajko Ćosić, Lukáš Malý,
Marie Sadowská, Adéla Vrtková

Katedra aplikované matematiky
Fakulta elektrotechniky a informatiky
VŠB - Technická univerzita Ostrava
2019

Předmluva

Milí studenti,
rád bych Vás jménem Katedry aplikované matematiky a Jednoty českých matematiků a fyziků přivítal na již 15. ročníku našeho semináře ŠKOMAM (ŠKOla MAtematického Modelování). Matematika je z mnoha důvodů krásná disciplína a přestože je vytvářená a vybrušovaná už několik tisíc let (a to co vzniklo, je prostě úžasné), je pořád v obrovském rozmachu. Je totiž nesmírně užitečná.

Posledních pár desetiletí žijeme v době počítačů a dnes už i superpočítačů (jeden z největších ve střední Evropě je i u nás na Vysoké škole báňské – Technické univerzitě Ostrava – můžete se podívat na stránky Národního superpočítačového centra IT4Innovations: https://www.it4i.cz), což přináší dříve netušené možnosti rozvoje vlastně všech oblastí lidské činnosti. A ukazuje se, že chceme-li opravdu využít všech možností, které přináší intenzivní rozvoj výpočetní techniky, je třeba vyřešit spoustu nově vznikajících problémů, upravit stávající (a nebo vymyslet zcela nové) algoritmy, které efektivně využijí architekturu superpočítačů, a použijí nejnovější poznatky z matematiky nebo znalosti získané díky spojení různých oborů matematiky. A zde je obrovský prostor a spousta výzev pro matematiku a matematiky.

V reakci na tuto potřebu doby u nás na katedře vzniknul před více než 15 lety studijní obor „Počítačová matematika“ (o pár let později přejmenovaný na „Výpočetní matematika“) spojující studium aplikované matematiky se základy informatiky. Tento obor (ve všech stupních VŠ vzdělání, tj. bakalářském, magisterském a doktorském) již absolvovalo více než 250 studentů. Že se věc daří, dokládají nejen úspěchy našich studentů v celostátních soutěžích, ale především ta skutečnost, že je o naše absolventy velký zájem. Minulý rok jsme náš obor podstatně upravili a přetavili ho do studijního programu „Výpočetní a aplikovaná matematika“. Budeme moc rádi, pokud si ho vyberete. Jsme přesvědčeni, že tím neuděláte chybu!

Následující dny trávené u nás na ŠKOMAMu můžete brát jako malou ochutnávku toho, co by Vás u nás čekalo. Poznáte některé z členů naší katedry, prohlédnete si posluchárny, ale především se budete bavit matematikou.

A já vám za nás všechny přeju, ať si to tu náramně užijete.

V Ostravě 18.1.2019 Jirka Bouchala

Úvod

Tento text je určen pro účastníky semináře Škola matematického modelování (http://skomam.vsb.cz) a slouží jako pomůcka k úlohám, které řeší studenti v průběhu tohoto semináře. Tento seminář, pro který používáme zkratku ŠKOMAM, organizuje Katedra aplikované matematiky (http://am.vsb.cz) Fakulty elektrotechniky a informatiky Vysoké školy báňské – Technické univerzity Ostrava jednou ročně již od roku 2005. V tomto roce probíhá již 15. ročník tohoto semináře.

Pro počítačové řešení úloh jsme v prvních 13 ročnících používali komerční systém Matlab. V minulém ročníku jsme se rozhodli udělat změnu a nově jsme zvolili programový balík R. Software R je zejména prostředí určené pro statistickou analýzu dat a jejich grafické zobrazení. Jazyk R ale umožňuje navíc i manipulaci s daty, numerické výpočty a grafické výstupy. Obsahuje také (podobně jako Matlab) řadu dalších knihoven s mnoha připravenými funkcemi. Jeho hlavní výhodou je jeho volná dostupnost pro výukové i vědecké účely. Podrobný popis tohoto jazyka je k dispozici v [1]. Programový balík R je volně k dispozici na webové adrese https://www.r-project.org/. Pro práci s balíkem R doporučujeme prostředí RStudio, které je volně k dispozici na webové adrese https://www.rstudio.com/products/rstudio/download/.

Tento seminář je pořádán s finanční podporou Fakulty elektrotechniky a informatiky (http://www.fei.vsb.cz), statutárního města Ostravy a projektu Math Exercises for You (http://math4u.vsb.cz) podpořeného programem Erasmus+. Nad akcí převzala záštitu Jednota českých matematiků a fyziků (http://jcmf.vsb.cz).

Spolufinancováno z programu Evropské unie Erasmus+

Cvičení 1: R – nejen nástroj pro statistickou analýzu dat

Abychom se mohli věnovat pokročilejším matematickým úlohám, potřebujeme vhodné prostředí, které nám jejich řešení umožní. Software R je zejména statistický software určený pro statistickou analýzu a grafickou reprezentaci dat, nicméně zvládne řešení i jiných úloh. Podstatnou výhodou softwaru R je jeho volná šiřitelnost. Podrobně se tomuto pracovnímu prostředí věnuje přiložený dokument „Jak pracovat s jazykem R“ [1]. My si zde uvedeme pouze stručný přehled Rkových proměnných a příkazů, které budeme potřebovat.

Prostředí

- *help, demo, methods, getAnywhere, library, ls, remove, class, dim, length, setwd*

Proměnné

- Skaláry
- Vektory
- Matice
- Datové rámce

Příkazy

- Skalární funkce - *sin, cos, tan, exp, log, abs, sqrt, round*
- Vektorové funkce a generování vektorů - *max, min, mean, median, sort, seq, rep*
- Maticové funkce a generování matic - *matrix, det, ones, zeros, diag*
- Změna typu objektu - *as.numeric, as.character, as.factor, as.matrix, as.data.frame*
- Importování dat - *read.csv, read.csv2*
- Skalární operace - $+,\ -,\ *,\ /, \hat{}$
- Maticové a vektorové operace - $+,\ -,\ \%*\%,\ t$ (transponování), *solve* ($solve(A, v) = x \Leftrightarrow Ax = v$)
  Operace „po prvcích“ - $*, \hat{}, /$
- Základní grafika (vykreslení grafů) - *plot, lines, points, dev.set, dev.next, dev.off*
- Pokročilá grafika - *plot, ggplot*
- Řídící příkazy - *if, ifelse* (podmíněné příkazy), *for, while* (příkazy cyklu se známým počtem opakování a podmínkou na začátku)

- Relace a logické operace - <, >, <=, >=, ==, ! =, &, |, !
- Skripty a funkce - *function*

Úkol 1.1 Legenda říká, že když byly vymyšleny šachy, tak se místnímu vládci (někde v Asii) tato hra tak zalíbila, že se rozhodl odměnit jejich vynálezce a za odměnu mu nabídl cokoliv, co si bude přát. Vynálezce mu na to odpověděl, že si nepřeje nic jiného než několik zrnek rýže. A aby se to dobře počítalo, tak že chce za první políčko šachovnice dostat jedno zrnko rýže, za druhé dvě zrnka rýže, za třetí čtyři zrnka, za čtvrté osm zrnek a tak dále. Tedy ať za každé další pole šachovnice dostane dvojnásobný počet zrnek rýže ve srovnání s polem předchozím. Kolik kilogramů rýže žádal, jestliže 30 000 zrnek rýže váží 1 kilogram? ▲

Úkol 1.2 Na spořící účet do banky jste učinili vklad 5000 Kč a chcete jej nechat zhodnocovat do doby, kdy budete mít našetřeno 200 000 Kč. Roční úroková sazba je v době vkladu 15%, ale je známo, že každý rok klesá o 0,1 procentního bodu. Zjistěte, kolik let je potřeba za těchto podmínek nechat vklad zhodnocovat, abyste dosáhli požadované cílové částky. ▲

Úkol 1.3 Sestrojte grafy následujících funkcí:

a) $f(x) = x^2$,

b) $f(x) = x^2 \sin\left(\frac{1}{x^2}\right)$,

c) $f(x) = |x|$. ▲

Úkol 1.4 Pomocí funkce *seq* si zadefinujte libovolný vektor (např. posloupnost čísel od 0 do 30 s krokem 3).

a) Vyberte ty prvky, které jsou větší než 5.

b) Vyberte ty prvky, které jsou větší než 5 a zároveň menší než 12.

c) Vyberte ty prvky, které jsou menší než 5 nebo větší než 12. ▲

Úkol 1.5 Pomocí funkce *matrix* vytvořte matice se 4 řádky a 3 sloupci, které budou obsahovat čísla od 1 do 12 řazena následovně

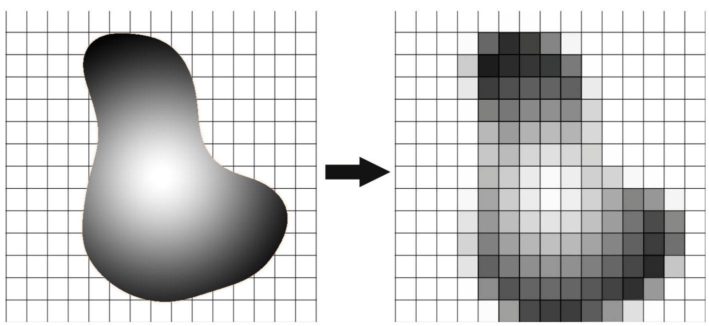
$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 5 & 9 \\ 2 & 6 & 10 \\ 3 & 7 & 11 \\ 4 & 8 & 12 \end{bmatrix}.$$

Aplikujte na tyto matice funkce *as.numeric*, *as.character* a *as.data.frame* a prozkoumejte výstup. ▲

V dobách analogových fotoaparátů se vyfocený obraz v každém okamžiku uchovával skutečně jako obraz. Ať už na negativu filmu, nebo po vyvolání na fotografickém papíře či diapozitivu. Dnes je tomu jinak. Při zmáčknutí spouště digitálního fotoaparátu se aktuální scéna před objektivem zachytí pomocí snímacího čipu a uloží jako soubor čísel na paměťovou kartu. I kdybychom tuto kartu rozebrali, obrázky na ní neuvidíme. K jejich zobrazení potřebujeme opět nějaké digitální zařízení, které umí obraz uložený v jedničkách a nulách převést do viditelné formy.

Matice obrazu

Pro jednoduchost se nejprve zabývejme černobílými obrázky - přesněji obrázky ve stupních šedé. V okamžiku exponování dojde k zachycení snímané scény na čip, který je tvořen soustavou miniaturních fotodiod uspořádaných do řádků a sloupců. V závislosti na osvětlení příslušné části čipu každá z diod vyprodukuje určitý náboj, který je změřen, a jeho hodnota je zaznamenána ve formě čísla. V případě JPEG obrázku se jedná o celé číslo od 0 do 255,[1] přičemž hodnota 0 odpovídá černé a hodnota 255 bílé. Proces rozdělení spojitého obrazu na diskrétní hodnoty v jednotlivých oddělených bodech se nazývá `kvantování` a `vzorkování`.



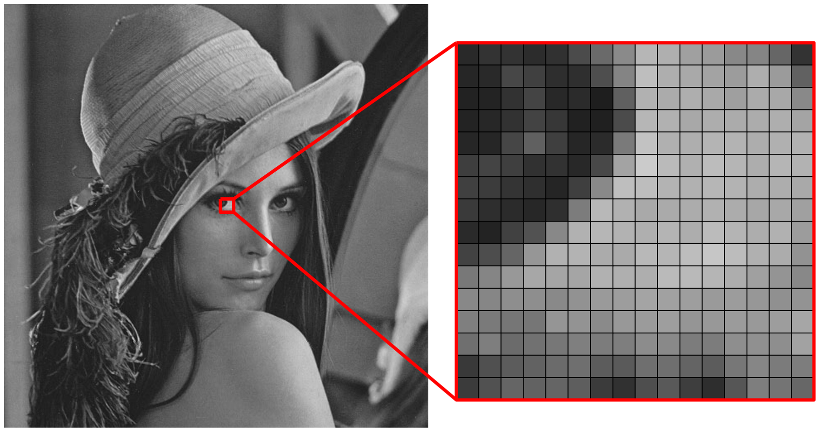
Obrázek 1: Příklad kvantování a vzorkování

Pomineme-li ve skutečnosti binární uložení obrazu, můžeme si jeho digitální podobu představit jako obecně obdélníkovou tabulku čísel, ve které každá hodnota reprezentuje jas určité malé plošky (pixelu) v zachyceném obraze. Tuto tabulku budeme označovat pojmem `matice obrazu`:

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix}, \quad \text{kde } a_{i,j} \in \{0, 1, 2, ..., 255\}.$$

Příklad části matice obrazu můžeme vidět na obr. 2.

[1]Tyto hodnoty odpovídají osmibitové reprezentaci čísla.

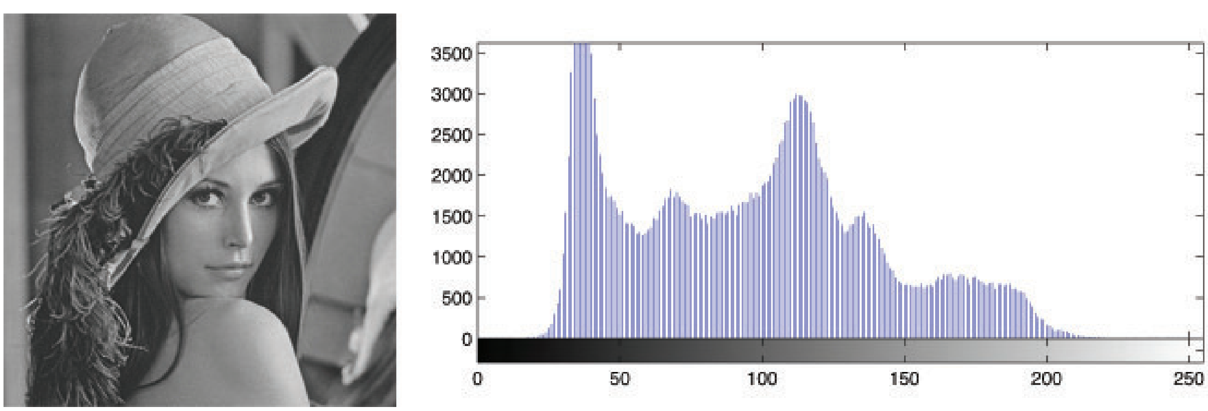


41	51	51	44	49	75	106	139	182	175	160	155	137	134	102	51
38	41	71	64	46	47	78	133	190	174	169	163	156	174	155	97
34	35	50	68	40	43	30	96	177	171	174	169	159	171	168	137
35	39	50	67	55	33	32	75	182	177	168	173	171	172	174	164
38	38	70	96	63	38	43	122	194	176	174	180	179	173	176	177
62	68	63	49	50	40	72	163	203	186	176	188	181	173	181	178
64	60	52	47	37	63	138	190	189	179	177	182	179	173	173	169
58	46	43	39	47	124	184	177	169	170	172	177	172	167	168	160
42	35	65	82	136	177	182	178	175	180	173	187	180	173	177	156
66	77	101	150	177	179	171	170	184	181	189	198	188	176	172	146
120	133	152	167	165	165	177	175	174	186	185	181	173	160	148	136
137	134	142	142	137	146	154	157	164	161	158	154	156	151	145	147
127	122	118	118	149	148	138	152	155	158	149	152	155	153	157	166
111	126	107	113	119	127	124	126	140	137	124	124	143	144	142	162
57	80	92	98	107	111	92	90	111	111	100	92	119	134	123	125
59	81	101	98	101	94	59	49	78	89	62	46	86	126	116	103

Obrázek 2: Část matice obrazu

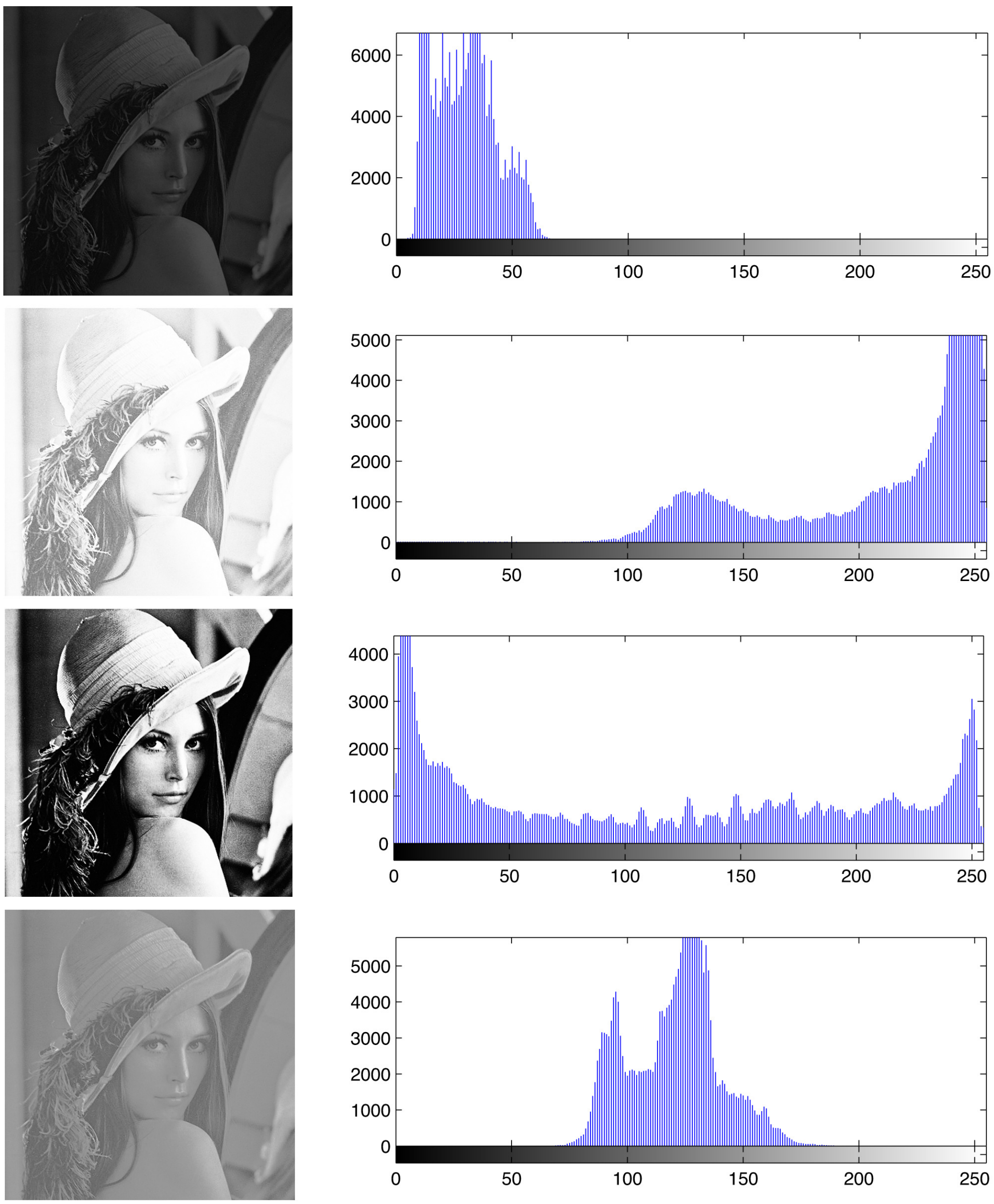
Histogram

Máme-li obrázek ve stupních šedé (číslech od 0 do 255), může nás zajímat, kolikrát se v obrázku jednotlivé stupně (čísla) vyskytují. Graf četnosti výskytu jednotlivých hodnot v obrázku se nazývá `histogram`.



Obrázek 3: Obrázek a jeho histogram

Přestože obrázek není svým histogramem jednoznačně definován, můžeme z histogramu o původním obrázku hodně vyčíst. Tmavé obrázky mají v histogramu velké hodnoty nahromaděné v levé části grafu. Naopak světlé obrázky mají v histogramu velké hodnoty v pravé části grafu. Histogram obrázků s nízkým kontrastem vypadá jako jeden relativně úzký „kopec“, zatímco obrázky s vysokým kontrastem mají většinou dva „kopce“, každý na opačné straně grafu.



Obrázek 4: Příklad různých obrázků a jejich histogramů

Jednoduché úpravy

Jak jsme si řekli, obrázky jsou v počítači uloženy jako matice čísel, takže místo manipulace s barevnými ploškami nám stačí provádět operace s čísly. Nyní si v jednoduchosti popíšeme některé základní operace s maticemi obrázků. Předpokládejme, že máme obrázek ve stupních šedé, který má 512×512 pixelů. Matice takového obrazu, kterou označíme I_{orig}, má 512 řádků a 512 sloupců. Dohromady je to 262 144 čísel, takže je jasné, že nebudeme počítat s každou hodnotou ručně, ale nastavíme nějaké pravidlo, které počítači řekne, co má s jednotlivými hodnotami udělat.

- `Ořez obrázku` provedeme jednoduše tak, že vybereme jen omezený rozsah řádků a sloupců původní matice. Například pro výřez pravé horní čtvrtiny obrázku I_{orig} vezmeme pouze 1. až 256. řádek a 257. až 512. sloupec. Dostaneme tam matici I_{crop}, která má 256 řádků a 256 sloupců.

- `Zesvětlení obrázku` provedeme například tak, že k hodnotě každého pixelu v matici I_{orig} přičteme nějaké kladné číslo, například 100. Touto operací se však může stát, že se některé hodnoty ocitnou mimo rozsah 0 až 255. Opravíme to například tak, že všechny hodnoty, které vyjdou větší než 255, zmenšíme právě na 255. Ve všech bodech, kde došlo k takovém ořezu hodnot ovšem ztrácíme obrazovou informaci.

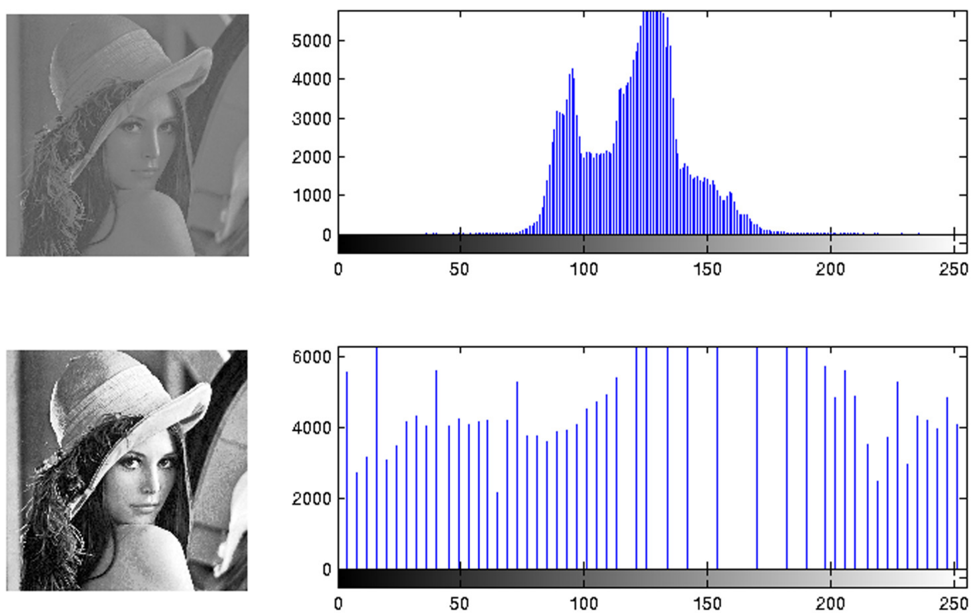
- `Ztmavení obrázku` můžeme provést obdobně jako zesvětlení, ale musíme zkontrolovat, zda některé hodnoty nevyšly záporné. Pokud ano, nastavíme je na 0. Další možností je vynásobit všechny hodnoty matice obrazu nějakým číslem z intervalu $(0, 1)$. Zde nehrozí, že bychom se dostali mimo rozsah 0 až 255, ale může se stát, že výsledkem nebudou jen celá čísla, a tak je potřeba každou výslednou hodnotu zaokrouhlit na nejbližší celé číslo.

- `Negativ obrázku` je obraz, který má obrácenou reprezentaci čísel. V klasickém obrázku odpovídá 0 černé a 255 bílé hodnoty mezi těmito čísly odpovídají různým stupňům šedé od tmavé po světlou. V negativním zobrazení je to naopak. Abychom nemuseli přeprogramovat zobrazovací zařízení k obrácení stupnice, můžeme obrátit stupnici přímo v našem obrazu. Hodnoty 0 změníme na 255 a naopak. Všechny hodnoty tedy vypočítáme tak, že původní hodnotu odečteme od čísla 255 a dostaneme právě negativ původního obrázku, který již zobrazíme klasicky.

- `Prahování` je jednoduchá operace, kde zvolíme $t \in (0, 255)$ (jednoduchý práh). Jednotlivé prvky matice I_{prah} nastavíme na 0, pokud příslušná hodnota matice I_{orig} je menší než t, a na 255, pokud je příslušná hodnota matice I_{orig} je větší nebo rovna t.

- `Vyrovnání histogramu` je metoda, která slouží k automatickému upravení kontrastu tak, aby byla četnost jednotlivých hodnot rozdělena přibližně rovnoměrně. Tato metoda je automatická a nezohledňuje, o jaký obrázek se jedná, takže výsledek může působit poněkud uměle.

Obrázek 5: Negativ obrázku (vlevo), obrázek po prahovaní s parametrem $t = 50$ (uprostřed) a $t = 150$ (vpravo)



Obrázek 6: Příklad vyrovnání histogramu obrázku s nízkým kontrastem

Barevné obrázky

Pro reprezentaci barevného obrazu se používají matice rovnou tři - jedna pro každou barevnou složku `RGB`.[2] Výsledný obraz pak vznikne složením těchto tří obrazů. Hodnota každého pixelu obrazu se tedy skládá ze tří čísel, a tak se tato ploška zobrazuje v jedné z 256^3 možných barev.

Pro jiný formát souboru může matice obrazu obsahovat i jiná čísla než celá čísla od 0 do 255. Můžeme například použít reálná čísla z intervalu $(0, 1)$, kdy řekneme, že

[2]RGB je systém rozložení barevného obrazu na červený (R), zelený (G) a modrý (B) kanál.

černá je 0 a bílá odpovídá hodnotě 1. Pokud uložíme hodnoty každého pixelu ve formátu 14-bitových čísel, dostaneme až 16384^3 barev. Běžné zobrazovací zařízení však neumí takový rozsah barev zobrazit, a proto ve většině případů stačí ukládat obrázky do osmi bitů.

Obrázek 7: JPEG obrázek složený z RGB kanálů

Úkol 2.1 Načtěte svůj obrázek do R pomocí příkazu *image.load*. Zobrazte tento obrázek pomocí funkce *plot* a zobrazte jeho jednotlivé složky R, G a B. Převeďte tento obrázek do stupňů šedé pomocí funkce *grayscale* a uložte pomocí funkce *image.save*. ▲

Úkol 2.2 Proveďte libovolný výřez vašeho černobílého obrázku obsahující 512×512 pixelů a uložte tento oříznutý obrázek. Zobrazte histogram tohoto obrázku pomocí funkce *hist*. ▲

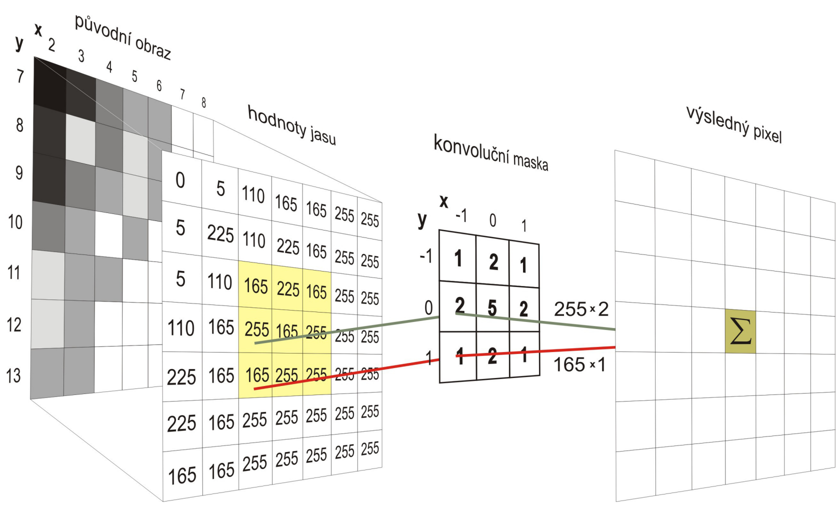
Úkol 2.3 Proveďte zesvětlení a ztmavení vašeho obrázku a vytvořte jeho negativ. Proveďte prahování s prahem $10, 100, 150, 200$. ▲

Úkol 2.4 Proveďte zvýšení kontrastu vašeho obrázku pomocí vyrovnání histogramu. ▲

Na závěr této části poznamenejme, že mnohem více lze o problematice zpracování obrazu nalézt v textu [2].

Konvoluce

V této části cvičení si zkusíme aplikovat na obrázky `diskrétní konvoluci`. Diskrétní konvoluce je zobrazení, do kterého vstupuje obrázek a takzvaná `konvoluční maska`. Konvoluční maska specifikuje, co konkrétně konvoluce s obrázkem provede. Protože se v počítačích obrázky vyskytují nejčastěji ve formě matice, jejíž každá hodnota odpovídá jasu v daném pixelu, nepřekvapí nás, když i konvoluční maska bude ze stejného světa. Diskrétní konvoluce tedy vezme matici obrázku a každému bodu přiřadí novou hodnotu podle následujícího schématu:



Obrázek 8: Schéma diskrétní konvoluce

Jednoduše řečeno: položíme masku na obrázek tak, aby střed masky byl v bodě, kde konvoluci počítáme, přenásobíme každou hodnotou v masce příslušnou hodnotu obrázku a potom vše sečteme.[3]

[3]Konvoluci vypočteme pomocí předpisu $(f * h)(x, y) = \sum_{r=-k}^{k} \sum_{s=-k}^{k} f(x+r, y+s)\, h(r, s)$, kde funkce f popisuje obrázek a funkce h masku.

Pomocí konvoluce jsme schopni odstranit z obrázků šum nebo zvýraznit hrany, což je hojně využíváno, chceme-li, aby počítač dokázal sám rozpoznat objekty na obrázku (třeba při zpracování dat z průmyslových kamer).

Čím je maska větší, tím větší okolí bodu nám může zasáhnout do výpočtu. Názorně je to vidět v následujícím příkladu, kdy byla nejdříve použita maska

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix},$$

která vlastně novou hodnotu spočítá jako průměr hodnot v okolních bodech. Poté na stejný obrázek aplikujeme masku větší, ale opět s hodnotami, které zprůměrují hodnoty obrázku:

$$\frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Obrázek 9: Lena před aplikací konvoluce (vlevo), po konvoluci s maskou 3×3 (uprostřed) a po konvoluci s maskou 5×5 (vpravo)

Jak si můžeme všimnout, naše maska zprůměrováním hodnot na okolí obraz rozmazala, a to tím více, čím větší okolí bodu se do průměru započítalo. Kdyby byla maska stejně veliká jako obrázek, tak by celý výsledný obrázek byl jen v jedné barvě.

Již jsme si vyzkoušeli, jak se v R pracuje s maticemi obrazu. Nyní si zkusíme naprogramovat diskrétní konvoluci podle následujícího algoritmu.

Algoritmus 1: Diskrétní konvoluce

```
obrazek = nacti_obrazek()
N,M = rozmery(obrazek)
maska = matice(n,n)
for i=1..N
  for j=1..M
    restrikce = obrazek[(i-n/2):(i+n/2),(j-n/2):(j+n/2)]
    novy_obrazek(i,j) = suma_prvku(maska * restrikce)
  end
end
uloz_obrazek(novy_obrazek)
```

Abychom se vyhnuli přečnívání masky „do prázdna“ při počítání hodnot u krajních bodů, nebudeme konvoluci počítat v bodech, ve kterých by maska přečnívala přes okraj obrázku. Vytvoříme si vlastně takový rámeček, na kterém výpočet konvoluce prostě vynecháme.

Úkol 2.5 Naprogramujte diskrétní konvoluci podle návodu výše a na obrázcích Lena.png a bricks.jpg (můžete použít i své obrázky z úvodu cvičení) otestujte následující konvoluční masky. Obrázky načítejte v odstínech šedi.

- $\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ ... odstraní šum
- $\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$ ... Gaussova maska
- $\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$ ... Laplaceova maska
- $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ ... zvýrazní hrany
- $\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$ ... zvýrazní svislé hrany

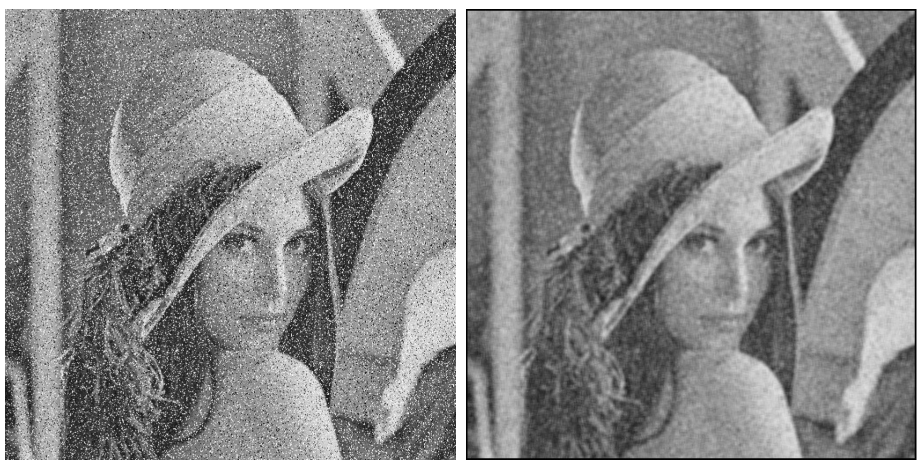
- $\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$ ... zvýrazní vodorovné hrany

U posledních čtyř masek přičtěte ke každému bodu ještě hodnotu 128, ať obrázky nejsou moc tmavé. ▲

Pro ilustraci praktického využití diskrétní konvoluce použijeme na zašuměný obrázek konvoluční masku

$$\frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix},$$

viz obr. 10.



Obrázek 10: Lena před odstraněním šumu (vlevo) a Lena po odstranění šumu (vpravo)

Jak je vidět, použili jsme stejnou masku, na které jsme si ukazovali rozmazání obrázku. Při odstraňování šumu pomocí diskrétní konvoluce se rozmazání obrázku pravděpodobně nevyhneme. Čím větší šum potřebujeme z obrázku odfiltrovat, tím více budeme muset obrázek rozmazat.

Velmi podobný výsledek bychom obdrželi i při použití Gaussovy masky

$$\frac{1}{273}\begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix},$$

viz obr. 11.

Obrázek 11: Lena před odstraněním šumu (vlevo) a Lena po odstranění šumu pomocí Gaussovy masky (vpravo)

Další informace o využití konvoluce pro zpracování obrazu může zvídavý čtenář získat opět v textu [2].

Cvičení 3: Statistika ve Hře o trůny

Hra o trůny – populární fantasy seriál, který jistě všichni alespoň z doslechu znají. Seriál je založen na knižní sáze od spisovatele George R. R. Martina. Příběh je plný bojů znepřátelených rodů o Železný trůn, krvavých bitev a podrazů, ale i strachu z příchodu zimy, která může trvat i celý lidský život. V tomto cvičení použijeme data obsahující údaje o bitvách, které se v příběhu odehrávají a které lze najít na odkaze [3]. Na nich si ukážeme, jak lze z `dat` vytáhnout `informace` a ty posléze třeba spojit do `znalostí`.

Jak jste se dozvěděli na přednášce, data zapisujeme do tzv. `datové matice`, což není nic jiného než tabulka, jejíž počet řádků je určen počtem pozorování a počet sloupců je určen počtem proměnných. Podívejte se na obr. 12, kde je zobrazeno prvních patnáct řádků a prvních pět sloupců datové matice, se kterou budeme dále pracovat. Každý řádek

	name	year	battle_number	attacker_king	defender_king
1	Battle of the Golden Tooth	298	1	Joffrey/Tommen Baratheon	Robb Stark
2	Battle at the Mummer's Ford	298	2	Joffrey/Tommen Baratheon	Robb Stark
3	Battle of Riverrun	298	3	Joffrey/Tommen Baratheon	Robb Stark
4	Battle of the Green Fork	298	4	Robb Stark	Joffrey/Tommen Baratheon
5	Battle of the Whispering Wood	298	5	Robb Stark	Joffrey/Tommen Baratheon
6	Battle of the Camps	298	6	Robb Stark	Joffrey/Tommen Baratheon
7	Sack of Darry	298	7	Joffrey/Tommen Baratheon	Robb Stark
8	Battle of Moat Cailin	299	8	Balon/Euron Greyjoy	Robb Stark
9	Battle of Deepwood Motte	299	9	Balon/Euron Greyjoy	Robb Stark
10	Battle of the Stony Shore	299	10	Balon/Euron Greyjoy	Robb Stark
11	Battle of Torrhen's Square	299	11	Robb Stark	Balon/Euron Greyjoy
12	Battle of Winterfell	299	12	Balon/Euron Greyjoy	Robb Stark
13	Sack of Torrhen's Square	299	13	Balon/Euron Greyjoy	Balon/Euron Greyjoy
14	Sack of Winterfell	299	14	Joffrey/Tommen Baratheon	Robb Stark
15	Battle of Oxcross	299	15	Robb Stark	Joffrey/Tommen Baratheon

Obrázek 12: Část datové matice

obsahuje informace o jedné bitvě, např. její název, rok, jméno útočícího krále, apod. Níže je uveden seznam všech proměnných se stručným popisem:

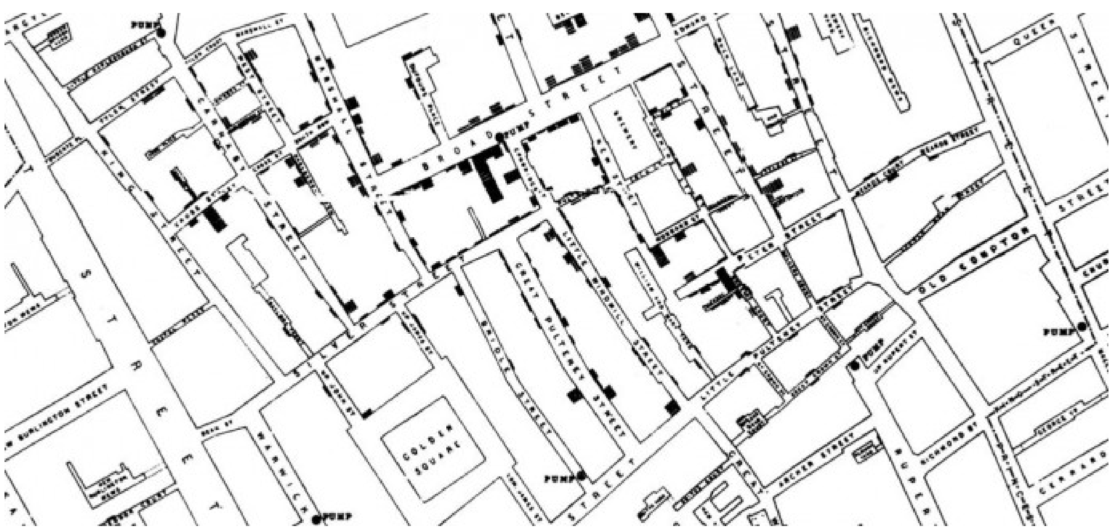
- `name` - název bitvy,
- `year` - rok, ve kterém se bitva odehrála,
- `battle_number` - unikátní ID bitvy,
- `attacker_king` - král útočící armády,
- `defender_king` - král bránící armády,
- `attacker_1` - hlavní útočící rod,
- `defender_1` - hlavní bránící se rod,

- `attacker_outcome` - výsledek bitvy z pohledu útočícího,
- `battle_type` - typ bitvy,
- `attacker_size` - velikost útočící armády,
- `defender_size` - velikost bránící se armády,
- `region` - území, kde se bitva odehrála.

Jednou ze základních úloh statistiky je `sumarizace` dat. Sumarizací můžeme rozumět popis kvantitativní proměnné například pomocí `průměru` a `mediánu` nebo popis kvalitativní proměnné pomocí `absolutních a relativních četností`. Při analýze výše popsané datové sady by nás mohla zajímat průměrná velikost útočící armády nebo třeba četnosti odehraných bitev na jednotlivých územích. Dokážete vymyslet další podobné úlohy?

Další velmi důležitou úlohou statistiky je nejen efektní, ale hlavně efektivní `vizualizace` dat. Jejím účelem je převedení dat a informací do grafické podoby, což umožní proniknout do zkoumaného problému, pochopit pozorované jevy nebo identifikovat hlavní trendy. Právě volba vhodného grafického zobrazení, vhodného grafu či diagramu, je největším úskalím kvalitně provedené vizualizace. Na tomto cvičení si vyzkoušíme pracovat se základními nástroji pro vizualizaci jako je `sloupcový graf` nebo `krabicový graf`, nicméně zájemci o pokročilejší vizualizaci dat se mohou podívat do katalogu grafických nástrojů [4].

Podíváme-li se kousek do historie, pravděpodobně o jednu z prvních vizualizací dat se pokusil již v 19. století anglický lékař, který zmapoval šíření cholery v Londýně a identifikoval tak zdroj nákazy. Tento lékař se jmenoval `John Snow` a na jeho znázornění šíření cholery se můžete podívat na obr. 13. Zájemci se poté mohou podívat na celý článek o tomto vědci vydaný v britském deníku The Guardian [5].



Obrázek 13: Vizualizace šíření cholery z 19. století

V první části cvičení si představíme knihovnu *dplyr*, která poskytuje funkce pro práci s datovou maticí, např. funkce *filter*, *select* nebo *summarise*. V průběhu analýzy totiž výzkumníka mohou zajímat pouze pozorování, která splňují určitou podmínku, nebo jen některé proměnné. Např. v našich datech se můžeme zajímat pouze o ty bitvy, které se odehrály v roce 299, popř. se budeme chtít zaměřit pouze na proměnnou popisující typ bitvy. Funkce knihovny *dplyr* nám umožní s datovou maticí jednoduše pracovat – vybírat řádky, vybírat sloupce, počítat sumární charakteristiky apod. V druhé části cvičení se pak seznámíme s knihovnou *ggplot2*, která obsahuje sofistikovaný aparát pro vizualizaci dat.

Reference

[1] K. Konečná, J. Koláček: Jak pracovat s jazykem R. [cit. 29.11.2018]. Dostupné z: https://www.math.muni.cz/~kolacek/vyuka/vypsyst/navod_R.pdf

[2] E. Sojka, J. Gaura, M. Krumnikl: Matematické základy digitálního zpracování obrazu. [cit. 19.12.2018]. Text vytvořený při realizaci projektu „Matematika pro inženýry 21. století“, Vysoká škola báňská - Technická univerzita Ostrava (2012). Dostupné z: http://mi21.vsb.cz/modul/matematicke-zaklady-digitalniho-zpracovani-obrazu

[3] *Kaggle* [online]. Game of Thrones. [cit. 29.11.2018]. Dostupné z: https://www.kaggle.com/mylesoneill/game-of-thrones

[4] *The Data Visualisation Catalogue* [online]. [cit. 29.11.2018]. Dostupné z: https://datavizcatalogue.com/

[5] *The Guardian* [online]. John Snow's data journalism: the cholera map that changed the world. [cit. 29.11.2018]. Dostupné z: https://www.theguardian.com/news/datablog/2013/mar/15/john-snow-cholera-map

Definice A.1 Nechť jsou dány prvky $a_{1,1}, a_{1,2}, \dots, a_{m,n}$ z dané množiny $\mathcal{F}$. `Matice` typu (m, n) (stručně $m \times n$ matice) je obdélníková tabulka

$$A = \begin{bmatrix} a_{1,1} & \dots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{m,1} & \dots & a_{m,n} \end{bmatrix},$$

která má $m \cdot n$ prvků $a_{i,j}$ uspořádaných do m řádků r_i^A a n sloupců s_j^A, takže

$$A = \begin{bmatrix} r_1^A \\ \vdots \\ r_m^A \end{bmatrix} = \left[s_1^A, \dots, s_n^A\right],$$
$$r_i^A = \left[a_{i,1}, \dots, a_{i,n}\right], \quad s_j^A = \begin{bmatrix} a_{1,j} \\ \vdots \\ a_{m,j} \end{bmatrix}.$$

Stručně píšeme též $A = [a_{i,j}]$.