Univerzita Palackého v Olomouci
Přírodovědecká fakulta
Katedra geoinformatiky

Veronika NEVTÍPILOVÁ

Testování neuronových sítí pro prostorovou interpolaci v softwaru GRASS GIS

Bakalářská práce

Vedoucí práce: Mgr. Justyna PASTWA

Olomouc 2013

Čestné prohlášení

Prohlašuji, že jsem bakalářskou práci bakalářského studia oboru Geoinformatika a geografie vypracovala samostatně pod vedením Mgr. Justyny Pastwy.

Všechny použité materiály a zdroje jsou citovány s ohledem na vědeckou etiku, autorská práva a zákony na ochranu duševního vlastnictví.

Všechna poskytnutá i vytvořená digitální data nebudu bez souhlasu školy poskytovat.

V Olomouci 22. května 2013 Veronika NEVTÍPILOVÁ

Děkuji vedoucí práce Mgr. Justyně Pastwě za dobré vedení, podporu, připomínky a rady při vypracování práce. Dále děkuji Mgr. Janu Cahovi a Ing. Michalu Mrnuštíkovi za odborné konzultace a rady.

Obsah

ÚVOD

Prostorová interpolace je poměrně často využívanou metodou pro práci s prostorovými daty. V současné době existuje mnoho interpolačních metod, z nichž každá má své uplatnění. Míra přesnosti těchto metod je však omezená, a proto se pro prostorovou interpolaci hledají nové postupy a metody. Jedním z těchto postupů je i využití neuronových sítí.

Samotný princip neuronových sítí je známý už velmi dlouho, první umělý neuron byl sestrojen v roce 1943 (Volná, 2008). Jejich využití v oblasti geoinformatiky se však začalo rozvíjet teprve v posledních letech. Z dostupné literatury je zřejmé, že neuronové sítě mají při použití k prostorové interpolaci velmi dobré výsledky srovnatelné s jinými interpolačními metodami, v některých případech i lepší (Snell, 2000; Bhaskaran, 2010; Chowdhury, 2010). Používání neuronových sítí pro účely prostorové interpolace zatím není příliš rozšířenou záležitostí mezi běžnými uživateli GIS, neboť většina dostupného GIS softwaru nemá v sobě neuronové sítě implementovány. Software GRASS GIS je jedním z mála, pro který existuje modul umožňující práci s neuronovými sítěmi, konkrétně s modelem vícevrstvého perceptronu. Tato práce se zabývá testováním tohoto modulu a jeho porovnáním s dalšími interpolačními metodami. Cílem je zjistit, zda je kvalita výsledné interpolace srovnatelná s klasickými metodami a jestli je možné používat neuronové sítě v tomto modulu k běžné interpolaci.

V kapitole 1 jsou stanoveny cíle práce. Kapitola 2 stručně shrnuje použité metody a postup práce. V kapitole 3 je stručně a jednoduše popsán teoretický základ používaných interpolačních metod - tedy neuronových sítí, IDW a krigingu, přičemž neuronovým sítím je zde věnována větší pozornost, neboť jsou hlavním tématem práce. Tato kapitola se také věnuje implementaci neuronových sítí ve dvou softwarech použitých v této práci a stručně hodnotí a srovnává příklady z literatury o použití neuronových sítí k prostorové interpolaci. Kapitola 4 popisuje vlastní práci - tvorbu dat, volbu nejlepší neuronové sítě, postup kroků, použité příkazy a nastavení při vlastní interpolaci v softwarech GRASS GIS a R Project. Dále je v ní popsáno následné zpracování dat. Výsledky práce jsou shrnuty v kapitole 5. Tato kapitola předkládá a hodnotí výstupy z předchozí interpolace, porovnává použité metody a hodnotí jejich kvalitu. Srovnány jsou zde neuronové sítě s ostatními vybranými metodami a také neuronové sítě z programů GRASS GIS a R Project mezi sebou.

1 CÍLE PRÁCE

Cílem bakalářské práce je zhodnocení kvality prostorové interpolace pomocí neuronových sítí. Teoretická část práce obsahuje rešerši literatury. Budou vybrány konkrétní příklady použití neuronových sítí k prostorové interpolaci, dále zde bude stručně popsán teoretický základ použitých metod a krátce zhodnocena implementace neuronových sítí do softwarů zpracovávající prostorová data, především bude zdůrazněn software GRASS GIS, který bude využit v praktické části práce.

V praktické části práce budou pomocí prostorové interpolace otestovány tři parametrově odlišné neuronové sítě s učícím algoritmem backpropagation (algoritmus zpětného šíření chyby) a výsledky interpolace pomocí těchto sítí budou srovnány s výsledky interpolace pomocí metod IDW a kriging. Nakonec bude zhodnocena kvalita interpolace pomocí neuronové sítě a bude srovnána interpolace pomocí neuronových sítí v programu GRASS GIS s vybraným programem, ve kterém lze tuto interpolaci také provádět.

Údaje o vytvořených datových sadách budou vyplněny v metainformačním systému Micka Katedry geoinformatiky.

2 POUŽITÉ METODY A POSTUPY ZPRACOVÁNÍ

2.1 Data

V práci byla použita umělá data náhodně generovaná v programu R Project pomocí funkce *grf*. Data byla generována ve třech variantách, které simulovaly různě členitý terén. V každé variantě bylo vygenerováno 1024 bodů v pravidelné mřížce. Každý bod obsahoval souřadnici x a y a výšku z. Podrobněji viz 4.1.

2.2 Programy

V této práci byly zejména použity programy GRASS GISS 6.4 (GRASS Development Team, 2012) a statistický open source software R Project (R Core Team, 2012) a jeho nadstavba s grafickým uživatelským rozhraním RStudio. Interpolace byla prováděna v softwaru GRASS GIS, kde hlavními použitými moduly byly *v.surf.idw* a *ann.** a dále v programu R Project pomocí balíčků pro neuronové sítě *nnet* a *neuralnet*. Pro kriging a IDW byly využity balíčky *gstat*, *geoR*, *automap* a *epiR*. Výpočet RMSE a dalších statistik probíhal v softwaru R Project.

2.3 Postup a metody zpracování

Návrh topologie neuronových sítí

Nejdříve bylo vytvořeno několik neuronových sítí s rozdílným počtem neuronů ve skrytých vrstvách. Poté byly natrénovány pomocí algoritmu backpropagation a otestovány na tréninkovém datasetu. Tímto testem bylo spočítání střední kvadratické chyby (RMSE). Z tohoto testu byl určen interval počtu neuronů, při kterém sítě vykazovaly nejlepší výsledky. Sítě s počtem neuronů v tomto intervalu byly znovu natrénovány na datech, která představovala různě členitý terén. Pro každou z navržených sítí byla vypočítána RMSE pro každý dataset a byla vybrána síť, která vykazovala nejmenší průměrnou RMSE na testovacích datech.

Vlastní interpolace

Na datasetu, který nebyl součástí výběru vhodné sítě, byla provedena interpolace pomocí metod IDW, kriging a neuronové sítě, která vyšla z předchozího testu jako nejlepší, v programech GRASS GIS a R Project. Interpolace proběhla jednou pro každou členitost povrchu. V softwaru GRASS GIS bylo vytvořeno celkem dvanáct rastrů (tři pomocí neuronové sítě natrénované na rastrových datech, šest pomocí neuronové sítě natrénované na vektorových datech a tři pomocí IDW), v softwaru R Project bylo vytvořeno dvanáct rastrů (šest pomocí neuronových sítí ze dvou balíčků, šest pomocí metod IDW a kriging).

Zhodnocení výsledků interpolace

Pro všechny výsledné interpolované povrchy byla spočítána střední kvadratická chyba (RMSE) podle vzorce

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(z_{di} - z_{ri})^2} \tag{1}$$

kde n je počet počet bodů, pro které je RMSE počítána, z_{di} je hodnota v bodě i z interpolovaného povrchu, z_{ri} je původní hodnota v tomto bodě.

Aby bylo možné vizuálně porovnávat rozdíly mezi metodami, byly od sebe odečteny výsledné rastry a získány hodnoty rozdílů mezi nimi. Vždy byl odečítán rastr vytvořený metodami IDW a kriging od rastru vytvořeného pomocí neuronových sítí.

Porovnání klasických metod s neuronovými sítěmi

Srovnána byla hodnota RMSE pro všechny členitosti povrchu a také časová náročnost jednotlivých metod. Porovnání zahrnovalo i celkové možnosti nastavení a práce s jednotlivými metodami v obou softwarech. Nakonec byly srovnány metody v programech GRASS GIS a R Project mezi sebou v rámci programu a možnosti neuronových sítí mezi oběma programy.

3 METODY PROSTOROVÉ INTERPOLACE

Prostorová interpolace je proces, při kterém jsou ze známých hodnot určitého jevu odhadovány hodnoty v místech, kde nebyly naměřeny. Toto platí pouze pro spojité jevy, nespojité jevy nemá smysl interpolovat. Podstatu většiny metod prostorové interpolace shrnuje Toblerův zákon: "Všechna místa jsou spolu související, ale bližší místa spolu souvisejí více než ta vzdálenější" (Longley et al., 2005).

Podle Longley et al. (2005) interpolace nachází uplatnění například v:

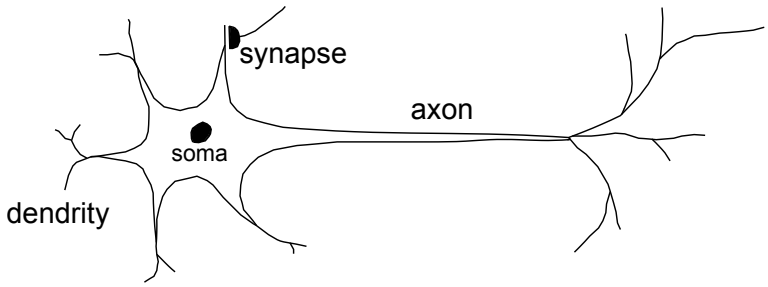
- odhadování množství srážek nebo teploty v místech kde neproběhlo přímé měření,
- tvorbě digitálních modelů terénu (povrchu),
- převzorkování rastrů.

3.1 Neuronové sítě

Biologický neuron a jeho zjednodušené vlastnosti posloužily jako základní jednotka umělých neuronových sítí. Ty byly vytvořeny jako zjednodušený matematický model napodobujících fungování lidského mozku (Rumelhart et al., 1994).

3.1.1 Biologický neuron

Biologický neuron je základní stavební a funkční prvek nervové soustavy. Je to specializovaná buňka sloužící k přenosu signálů a informací nutných pro zajištění životních funkcí organismu. Skládá se z těla (somatu), kde dochází ke zpracování informací, a vstupních a výstupních výběžků (dendritů a axonu) viz obrázek 3.1. Dendrity slouží ke vstupu informací, axon předává zpracované informace dendritům další buňky. K samotnému přenosu informace mezi jednotlivými neurony slouží synapse - speciální rozhraní v místech styku výběžků neuronů.



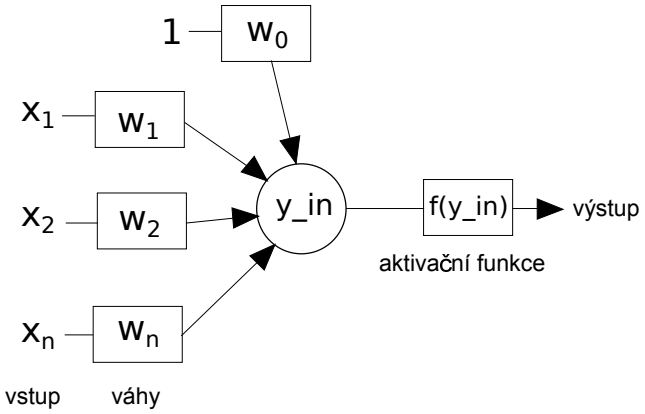
Obrázek 3.1: Biologický neuron (Zdroj: http://www-cs-faculty.stanford.edu/ eroberts/courses/soco/projects/neural-networks/Neuron/index.html)

Při průchodu signálu synapsemi dochází ke změně a zakódování synaptické propustnosti, a tak vzniká paměťová stopa. Při procesu učení vznikají nové paměťové stopy a při zapomínání se synaptická spojení přerušují a paměťové stopy mizí.

Přenos signálu mezi neurony zajišťuje membrána obalující neuron, která je za určitých podmínek schopná generovat elektrické impulsy. Impuls je mezi neurony přenášen přes synaptická spojení mající určitou propustnost, která určuje míru podráždění sousedních neuronů. Pokud míra podráždění neuronu přesáhne určitou hranici, tzv. práh, je vygenerován nový impuls a tak je zajištěno šíření informace (Volná, 2008).

3.1.2 Formální neuron

Formální neuron (dále jen neuron), vycházející z biologického neuronu, tvoří základ matematického modelu neuronové sítě. Skládá se z n reálných vstupů, které nahrazují dendrity a tvoří vstupní vektor $\mathbf{x} = (x_1, \ldots, x_n)$. Každý ze vstupů je ohodnocen reálnou synaptickou váhou, která může nabývat kladných i záporných hodnot. Synaptické váhy tvoří vektor $\mathbf{w} = (w_1, \ldots, w_n)$. Dalším vstupem neuronu je vstup $x_0 = 1$ ohodnocený váhou w_0, který představuje prahovou hodnotu (Volná, 2008). Model jednoduchého formálního neuronu viz obrázek 3.2.



Obrázek 3.2: Formální neuron (Zdroj: http://www.root.cz/clanky/biologicke-algoritmy-4-neuronove-site/)

Suma všech vážených vstupů y_in udává vnitřní potenciál neuronu

$$y\_in = \sum_{i=0}^{n} w_i x_i \tag{2}$$

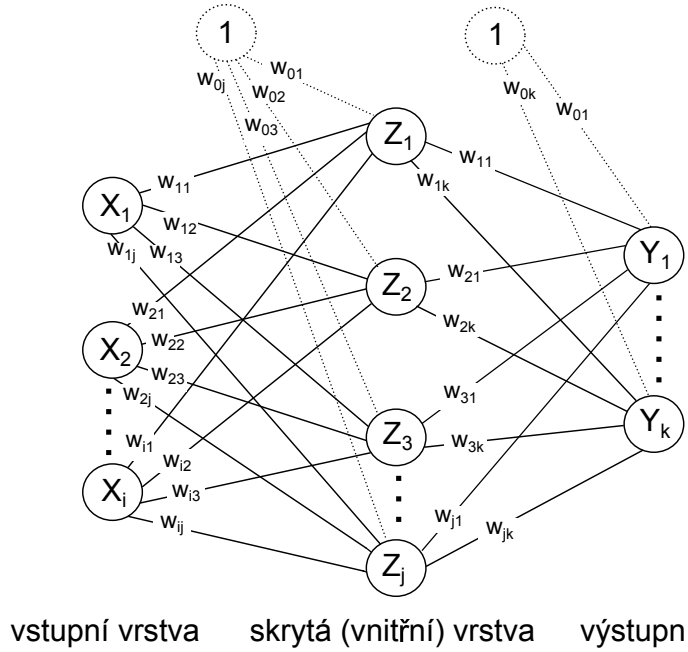
Pokud hodnota vnitřního potenciálu dosáhne prahové hodnoty $x_0 w_0$, pak vzniká výstup (stav) y neuronu. Tato výstupní hodnota je modifikována *aktivační (přenosovou)* funkcí f: $y = f(y\_in)$. Nejjednodušším typem aktivační funkce je ostrá nelinearita v tomto tvaru (Volná, 2008):

$$f(y\_in) = \begin{cases} 1 \text{ pokud} & y\_in \geq 0 \\ 0 \text{ pokud} & y\_in < 0 \end{cases} \tag{3}$$

Tento jednoduchý model neuronu bývá označován jako perceptron (Voženílek, 2011).

3.1.3 Neuronová síť

Neuronová síť vzniká je propojením neuronů. Způsob propojení je takový, že výstup z jednoho neuronu je vstupem dalších neuronů (Volná, 2008). Neurony v síti jsou organizovány do vrstev (viz obrázek 3.3). Každá síť obsahuje vstupní a výstupní vrstvu a libovolný počet skrytých vrstev (Voženílek, 2011). Počet neuronů v každé vrstvě a způsob jejich propojení dává dohromady *architekturu sítě*. Aktivační funkce (viz. část 3.1.2) je ve většině případů pro všechny neurony v síti stejná (Volná, 2008).



Obrázek 3.3: Příklad neuronové sítě (volně převzato z (Volná, 2008))

Důležitou vlastností neuronové sítě je učení. Při učení se mění váhy mezi neurony. Váhy v síti jsou posilovány nebo zeslabovány, podle toho jestli vedou ke správné nebo špatné odpovědi. Učením se neuronová síť nastavuje tak, aby dávala co nejpřesnější výsledky. Existují různé typy učení. Hlavními jsou učení s učitelem (supervised) a bez učitele (unsupervised)(Voženílek, 2011), Volná (2008) uvádí jako další typ učení posilováním (reinforcement learning).

Učení s učitelem Pro učení existuje trénovací množina, která se skládá z dvojic vzor-výstup. Vzor slouží jako vstup neuronové sítě. V síti se náhodně nastaví váhy a spočítá se výstup sítě. Tento výstup je porovnán s požadovaným výstupem v trénovací množině a je stanovena chyba. Poté proběhne úprava vah tak, aby se v příštím výpočtu zmenšila chyba. Toto je opakováno, dokud není dosaženo stanovené minimální chyby (Voženílek, 2011).

Učení bez učitele Vstupem do neuronové sítě je sada vzorů. Síť sama buď třídí vzory do shluků, nebo si přizpůsobuje topologii podle vlastností vstupních vzorů. Váhy jsou nastavovány tak, aby síť poskytovala stejné výsledky, pokud jsou vstupní vektory stejné nebo podobné (Voženílek, 2011).

Učení posilováním Někdy je také nazýváno učení odměnou a trestem. Požadovaná hodnota výstupu není známá, ale informace o vhodnosti výsledků jsou získávány z okolního prostředí, čímž je zajištěna určitá zpětná vazba. Tento typ učení je využíván především v oblasti robotiky a multiagentních systémů (Volná, 2008).

3.1.4 Model vícevrstvé dopředné sítě (vícevrstvého perceptronu)

Co je to perceptron bylo vysvětleno v části 3.1.2. Vícevrstvý perceptron je potom vícevrstvá neuronová síť, ve které je každý neuron modelován jako perceptron. Aktivační funkcí neuronů ve vícevrstvém perceptronu je diferencovatelná spojitá funkce, nejpoužívanější je sigmoidní funkce (Tarassenko, 1998):

$$f(x) = \frac{1}{1 + e^x} \tag{4}$$

Ve vícevrstvých sítích dochází k tzv. úplnému propojení neuronů - každý neuron ve vrstvě je spojen se všemi neurony z vyšší (následující) vrstvy. Podle Volná (2008) se síť na obrázku 3.3 označuje jako třívrstvá (vstupní, vnitřní, výstupní vrstva), Tarassenko (1998) takovouto síť chápe jako dvouvrstvou - jako vrstvy označuje váhově ohodnocená spojení místo neuronů, protože vstupy sítě netvoří neurony ale pouze vektor hodnot.

Nejčastějším typem sítí jsou sítě dopředného typu (feedforward). Šíření signálu v nich probíhá následujícím způsobem. Vstupní vektor x_i je vynásoben vektorem vah w_i a přenesen k neuronům ve vnitřní (skryté) vrstvě. V těchto neuronech proběhne zpracování vstupních hodnot aktivační funkcí, výsledný vektor z_j je vynásoben vahami w_j a slouží jako vstup do další (v tomto případě výstupní) vrstvy. Výstup y_k z této vrstvy je zároveň výstupem celé sítě.

3.1.5 Algoritmus backpropagation

Nejčastěji používaný algoritmus pro učení vícevrstvých neuronových sítí je backpropagation. Jedná se o učení s učitelem. Podle Volná (2008) probíhá ve třech fázích. První z nich je dopředné šíření signálu (viz 3.1.3).

Druhou fází je zpětné šíření (back propagation). Pro každý neuron Y_k ve výstupní vrstvě je spočítáno δ_k, cože je část chyby, která se zpětně šíří do neuronů v předcházející vrstvě podle vzorce

$$\delta_k = (t_k - y_k)\ f'(y\_in_k) \tag{5}$$

kde t_k je očekávaný výstup z neuronu, y_k je vypočtený výstup a y_in_k je vnitřní potenciál neuronu Y_k. Poté je ke každému neuronu Z_j ve vnitřní vrstvě přiřazena suma δ_k vstupů z výstupní vrstvy

$$\delta\_in_j = \sum_{k=0}^{m} \delta_k w_{jk} \tag{6}$$

Pomocí této sumy a derivace vnitřního potenciálu z_in_j neuronu Z_j je spočítána částečná chyba δ_j

$$\delta_j = \delta\_in_j \; f'(z\_in_j) \tag{7}$$

Vypočítané δ_j slouží k úpravě vah mezi vstupní a vnitřní vrstvou sítě: $\Delta v_{ij} = \alpha \delta_j x_i$ kde α je koeficient učení a x_i je vstupní hodnota sítě. Pomocí δ_k jsou upravovány váhy mezi vnitřní a výstupní vrstvou sítě: $\Delta w_{jk} = \alpha \delta_k z_j$ kde z_j je výstupní hodnota neuronu Z_j.

Třetí fází je aktualizace vah na spojení neuronů. Novou váhu označíme jako n a starou jako s. Pak $v_{ij}(n) = v_{ij}(s) + \Delta v_{ij}$ a $w_{jk}(n) = w_{jk}(s) + \Delta w_{jk}$

3.1.6 Implementace v GIS

GRASS GIS

V softwaru GRASS GIS se k práci s neuronovými sítěmi využívá pět skriptů *ann.**. Tyto skripty jsou napsány v programovacím jazyce Python a používají k práci s neuronovými sítěmi knihovnu FANN (Fast Artificail Neural Network) [1]. Skripty jsou navrženy tak, aby pracovaly s rastrovými daty a umožňují používat vícevrstvý perceptron s trénovacím algoritmem backpropagation. Autorem toho to modulu je Pawel Netzel [2].

Skript *ann.data.rast.py* využívá rastrová data k určení vstupu a požadovaný výstupu sítě, dále potřebuje určitý počet náhodně rozmístěných kontrolních vektorových bodů. Z těchto dat pak vytváří soubor s příponou .dat, který slouží k učení neuronové sítě. Parametry sítě se nastavují pomocí skriptu *ann.create.py*. Zde je možno zadat počet vstupních a výstupních neuronů a počet neuronů ve skrytých vrstvách, dále stupeň propojení neuronů a koeficient učení.

K učení vytvořené sítě se používá skript *ann.learn.py*. Tento skript využívá učící algoritmus backpropagation a je zde možnost zadat požadovanou chybu a maximální počet opakování. Délka učení závisí na počtu skrytých vrstev a neuronů v nich a na množství kontrolních bodů, ze kterých je vytvořen soubor *.dat, z kterého se síť učí.

K samotnému výpočtu pomocí natrénované sítě slouží skript *ann.run.rast.py*. Uživatel zadá vstupní souřadnice v rastrovém formátu a název natrénované sítě. Rychlost výpočtu je ovlivněna rozlišením rastru, pro který je výpočet prováděn.

[1] `http://leenissen.dk/fann/wp/`

[2] `http://www.meteo.uni.wroc.pl/index.php?option=com_content&view=article&id=31&Itemid=25`

R Project

V softwaru R Project slouží k trénování a výpočtům sítí balíčky *nnet* a *neuralnet*.

Balíček nnet (Venables a Ripley, 2002), umožňuje trénování dopředné sítě s učícím algoritmem backpropagation a jednou skrytou vrstvou. K dispozici je nastavení základních parametrů: vstup a požadovaný výstup, počet neuronů ve skryté vrstvě, číslo k náhodné inicializaci vah, maximální počet opakování. Trénink sítě je poměrně rychlý a výsledky jsou srovnatelné s ostatními metodami.

Balíček neuralnet (Fritsch et al., 2012) se v mnohých ohledech podobá balíčku *nnet*, ale možností nastavení parametrů je více. Oproti předchozímu balíčku je zde možno nastavit více skrytých vrstev s libovolným počtem neuronů v každé z nich, dále je zde na výběr několik druhů učících algoritmů a druh aktivační funkce. Trénování sítě probíhá pomaleji než s balíčkem *nnet*, ale natrénované sítě většinou vykazují při testování menší chybu než sítě z balíčku *nnet*.

Výpočet pomocí natrénované sítě probíhá u obou balíčků velmi podobně. Je zadána natrénovaná síť a vstupní data, pro které budou výsledky spočítány. Samotný výpočet trvá velmi krátce u obou balíčků.

3.1.7 Příklady využití k interpolaci

Neuronové sítě se již delší dobu osvědčují při prostorové interpolaci ať už v běžných případech, kdy dávají často lepší výsledky než klasické metody, nebo tam, kde klasické metody nelze z určitých důvodů použít. Konkrétními příklady využití neuronových sítí k interpolaci jsou práce:

- Srovnání ordinary krigingu a neuronových sítí pro prostorové mapování kontaminace podzemní vody arsenem (Chowdhury, 2010),
- Nový přístup k odvozování polí teploty a salinity v Indickém oceáně za použití neuronových sítí (Bhaskaran, 2010),
- Prostorová interpolace povrchové teploty vzduchu pomocí neuronových sítí: Hodnocení jejich využití při vytváření podrobnějších obecných cirkulačních modelů (Snell, 2000).

Tyto příklady mají několik společných rysů. Ve všech byla k interpolaci použita vícevrstvá dopředná síť. K natrénování sítě byl použit algoritmus backpropagation. V případě, že byly výsledky interpolace pomocí neuronových sítí srovnávány s výsledky klasických interpolačních metod, bylo zjištěno, že neuronové sítě poskytly ve většině případů přesnější a lepší výsledky než klasické metody a nezáleželo na tom, jakým způsobem srovnávání probíhalo.

Ačkoliv byl ve všech případech použit stejný typ sítě, jednotlivé modely se od sebe lišily. Bhaskaran (2010) a Chowdhury (2010) vytvořili ve svém modelu sítě dvě skryté

vrstvy, Snell (2000) použil jen jednu. Lišil se i počet neuronů ve skrytých vrstvách a počet vstupních parametrů sítě. Zatímco Chowdhury (2010) použil jen dva vstupní parametry (zeměpisnou šířku a délku), Bhaskaran (2010) používá tři a Snell (2000) mnohem více.

Tyto rozdíly v nastavení dokazují, že není možné určit jeden univerzální typ sítě, ale je třeba vybírat typy sítí a jejich nastavení podle konkrétního problému.

3.2 Inverse distance weighting (IDW)

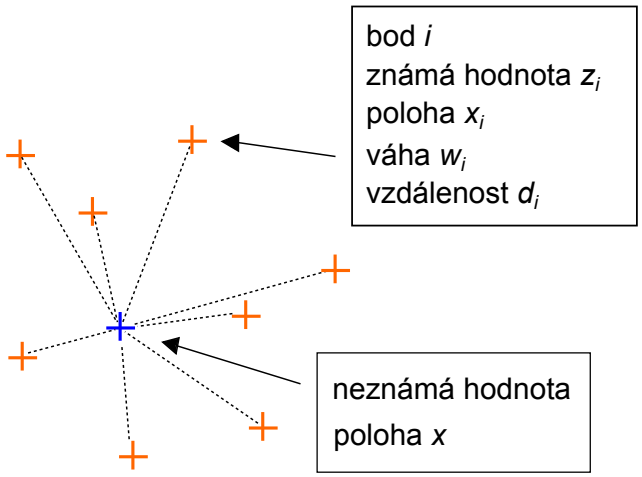
IDW, neboli metoda inverzních vzdáleností, je jednou z nejpoužívanějších interpolačních metod. Je velmi jednoduché ji naprogramovat, porozumět jí a používat ji. IDW patří mezi exaktní interpolační metody, které při výpočtu nemění naměřené známé hodnoty (Longley et al., 2005).

3.2.1 Inverse distance weighting model

Metoda IDW odhaduje neznámé hodnoty jako vážený průměr z okolních známých hodnot, přičemž bližší hodnoty mají vyšší váhu než ty vzdálenější. Hodnota v neznámém bodě se vypočítá pomocí vzorce

$$z(x) = \frac{\sum_{i=0}^{n} w_i z_i}{\sum_{i=0}^{n} w_i} \tag{8}$$

kde $z(x)$ je neznámá hodnota v bodě x, z_i jsou známé hodnoty a w_i jsou jejich váhy (Longley et al., 2005).



Obrázek 3.4: IDW (Zdroj: Longley et al. (2005))

Existuje několik způsobů jak určit váhu bodů se známými hodnotami. Nejčastěji používaný je ten, kdy se váha spočítá jako převrácená hodnota vzdálenosti od bodu s neznámou hodnotou a vzdálenost d je umocněna parametrem p (Horák, 2011).

$$w_i = \frac{1}{d_i^p} \tag{9}$$

Doporučuje se nastavení p v rozmezí 1 - 3. Pokud je $p < 1$ pak je výsledek interpolace méně zahlazený, pokud je $p > 1$ pak je to naopak (Neteler a Mitasova, 2008).

Výběr bodů se známými hodnotami, ze kterých bude spočítána neznámá hodnota může probíhat několika způsoby. Nejčastěji se určuje počet nejbližších bodů, které vstoupí do výpočtu, další možností je určení prahové vzdálenosti, kdy body za ní už nemají na výpočet vliv (Longley et al., 2005).

Metoda IDW se vyznačuje charakteristickými nežádoucími jevy, které vznikají jako důsledek výpočtu váženým průměrem. Způsob výpočtu umožňuje vznik nových hodnot pouze v rozsahu existujících hodnot. Pokud je takto interpolován terén, kde nebyly zaměřeny vrcholy a prohlubně, pak se budou vrcholy jevit jako prohlubně a naopak (Longley et al., 2005). Dalším nežádoucím jevem je tvorba koncentrických izolinií kolem bodů s původními hodnotami tzv. bull's eyes (Horák, 2011).

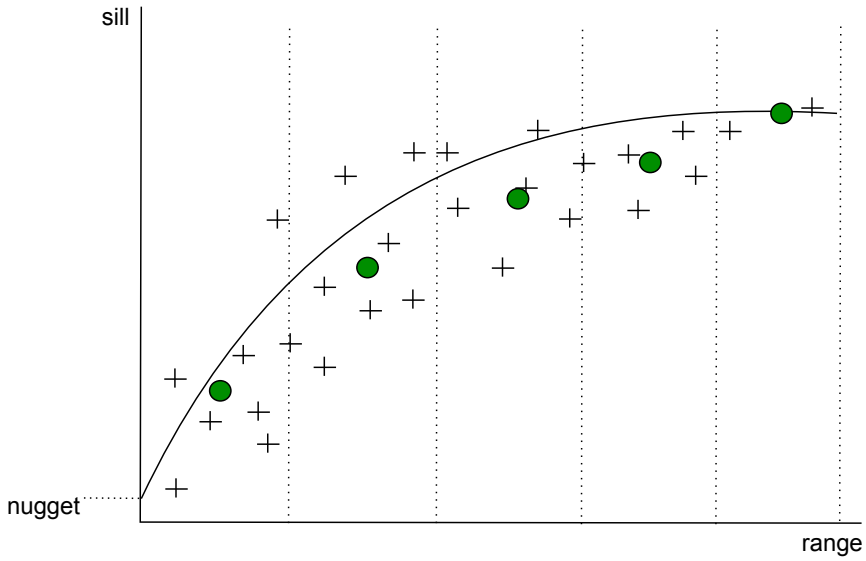
Metoda IDW je snadno dostupná ve většině GIS softwarů, např. ArcGIS, QGIS, GRASS GIS, IDRISI, R.

3.3 Ordinary kriging

Základní myšlenkou krigingu je nalezení určitých obecných vlastností podle naměřených hodnot a aplikování těchto vlastností při výpočtu neznámých hodnot. Z těchto vlastností je nejdůležitější hladkost (smoothness). Teorie říká, že hodnoty v bližších bodech jsou si podobnější než ve vzdálenějších bodech. Rozdíl hodnot z mezi dvěma body je spočítán jako

$$(z(x) - z(x)_i)^2 \tag{10}$$

S narůstající vzdáleností je pravděpodobné, že se bude tento rozdíl zvětšovat až do určité vzdálenosti a pak už se nebude měnit (Longley et al., 2005). Obrázek 3.5 je příklad semivariogramu, který znázorňuje jak se mění rozdíly mezi jednotlivými páry bodů z naměřených hodnot. Hodnota *sill* značí polovinu umocněného rozdílu mezi párem bodů - tedy semivarianci, *range* je vzdálenost, ve které se hodnota *sill* už nemění. Hodnota semivariance není nikdy 0, ani v nulové vzdálenosti. Proto je v semivariogramu parametr *nugget*, který udává rozdíl v hodnotě mezi dvěma body ve stejném místě, nebo ve velmi malé vzdálenosti. Křížky značí hodnoty rozdílu mezi vybranými páry bodů, kolečka jsou průměry z těchto hodnot v určité vzdálenosti. (Longley et al., 2005)



Obrázek 3.5: Příklad semivariogramu (Zdroj: volně převzato z Longley et al. (2005))

3.3.1 Ordinary kriging model

Ordinary kriging je standardní a často používanou verzí krigingu. Hodnota z v bodě s_0 je spočítána pomocí vzorce

$$z(s_0) = \sum_{i=0}^{n} w_i(s_0)\ z(s_i) = \lambda_0^T\ z \tag{11}$$

kde n je počet bodů sloužících k výpočtu, w_i je váha vypočítaná pro bod $z(s_i)$ a bod $z(s_i)$ je bod se známou hodnotou z. Zkráceně tedy λ_0 je vektor vah w_i a z je vektor n bodů o známých hodnotách. Váhy w_i jsou spočítány pomocí soustavy rovnic (Hengl, 2009).

Stejně jako IDW je i kriging známá interpolační metoda dostupná v populárních GIS softwarech.

4 TESTOVÁNÍ INTERPOLAČNÍCH METOD

Tato kapitola se zabývá vlastní interpolací pomocí zvolených metod v programech GRASS GIS 6.4-svn a R Project.

4.1 Tvorba dat

Pro účely testování interpolace pomocí neuronových sítí byly v softwaru R Project vytvořeny tři datasety, které simulovaly různou členitost povrchu. Nejčlenitější povrch byl označen jako *členitost 1*, středně členitý povrch jako *členitost 2* a nejméně členitý povrch jako *členitost 3*. Datasety byly náhodně generovány pomocí funkce *grf* (gaussian random fields) z balíčku *geoR*, která vytváří body a náhodně jim přiřazuje hodnoty. Tyto hodnoty jsou ovlivňovány dalšími parametry funkce.

```
grf(pocetBodu, grid = "reg", cov.pars = c(sill, range), nug = nugget,
 cov.model = covModel, aniso.pars = c(anisotropyDirection,
 anisotropyRatio), xlims = xlims, ylims = ylims)
```

Parametr *grid="reg"* určuje, že body budou vygenerovány v pravidelné mřížce. Parametr *cov.model* udává typ variogramu, v tomto případě byl použit sférický variogram. Hodnoty *xlims* a *ylims* byly zadány v intervalu $0 - 1$. Pro každý dataset bylo vygenerováno 1024 bodů v pravidelné síti. Tyto body měly tři atributy: souřadnice x a y v rozsahu $0 - 1$ (ovlivněno parametry *xlims* a *ylims*) a hodnotu z, která představovala výšku. Hodnoty parametrů ovlivňujících členitost, které byly použity při tvorbě datasetů viz tab. 4.1

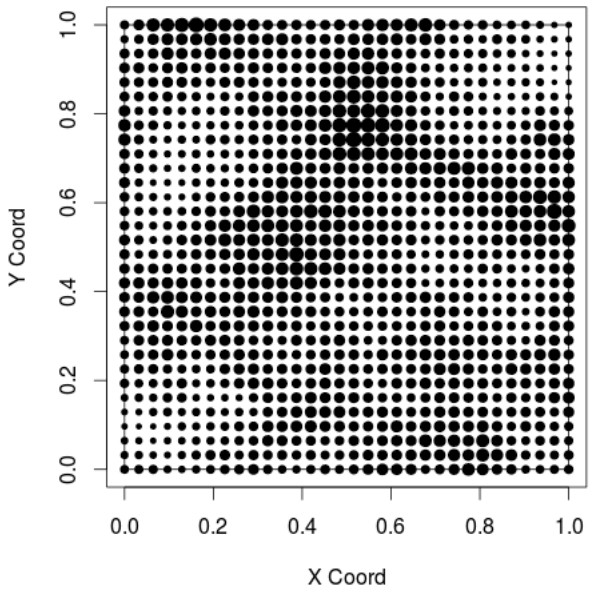
Tabulka 4.1: Hodnoty parametrů pro všechny členitosti povrchu

	sill	range	nugget	anisotropy ratio
členitost 1	0.12	0.3	0.00001	0.8
členitost 2	0.08	0.5	0.00001	0.8
členitost 3	0.01	1.2	0.00001	0.3

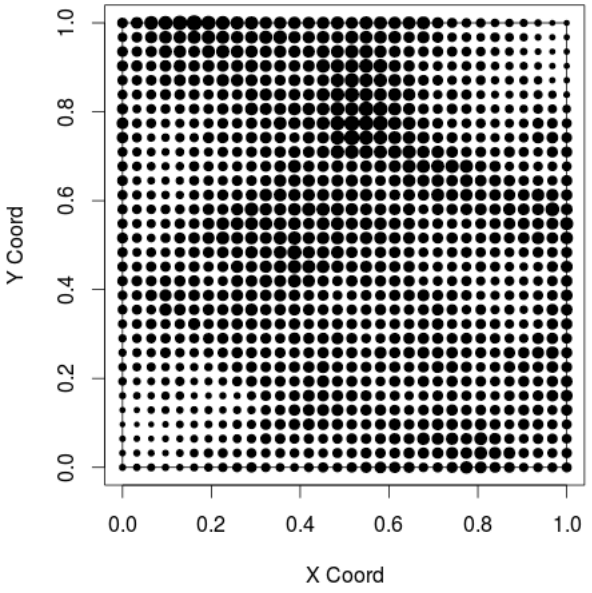
Rozsah hodnot z se pro každou členitost lišil. Data s nižší členitostí měla rozsah hodnot nižší.

- *členitost 1*: -0.7396643 – 1.090838
- *členitost 2*: -0.534922 – 0.8930179228
- *členitost 3*: -0.2012766 – 0.1411988275

Na obrázcích 4.1, 4.2 a 4.3 je znázorněno rozložení bodů v jednotlivých datasetech. Větší průměr kolečka značí vyšší hodnotu z.

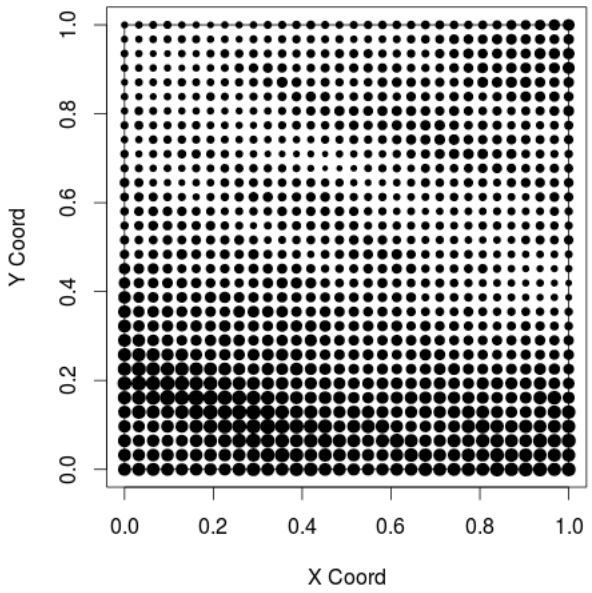


Obrázek 4.1: Rozložení bodů pro *členitost 1*



Obrázek 4.2: Rozložení bodů pro *členitost 2*

Poté byla pomocí funkce *sample* vytvořena z každého datasetu trénovací a testovací data. Trénovací data obsahovala 724 náhodně vybraných bodů a byla použita pro naučení neuronových sítí a pro vlastní interpolaci. Testovací data obsahovala zbylých 300 bodů a byla použita k výpočtu RMSE.



Obrázek 4.3: Rozložení bodů pro *členitost 3*

4.2 Výběr nejlepšího nastavení sítě

Pro účely testování bylo žádoucí vybrat takové nastavení sítě, které by poskytovalo co nejlepší výsledky. Výběr nastavení probíhal hlavně v programu R Project, díky jeho lepším výpočetním možnostem a rychlejšímu trénování neuronových sítí. Výběr nejlepšího nastavení probíhal metodou pokus – omyl (angl. test and trial). Výsledky následujícího testování nelze brát jako jediné správné a platné. Pokud by bylo testování prováděno vícekrát, mohlo by být dosaženo odlišných výsledků.

Jako první byl vytvořen jeden dataset způsobem popsaným v 4.1. Na tomto datasetu byla natrénována síť z balíčku *nnet*. Při každém tréninku sítě se měnil počet neuronů v intervalu $1 - 60$. Pro každou natrénovanou síť byla spočítána RMSE. Po seřazení výsledků podle nejmenší hodnoty RMSE byl vybrán interval $15 - 30$ neuronů, který byl použit v dalším testování. Stejný postup byl použit u sítě z balíčku *neuralnet*. Pro další testování byl opět zvolen interval počtu neuronů $15 - 30$. Tyto testy byly základem dalšího testování i pro neuronové sítě v programu GRASS GIS.

4.2.1 Výběr nastavení pro balíček nnet

Byl vytvořen skript, který postupně vytvářel datasety způsobem popsaným v 4.1. Pro každou členitost povrchu bylo vytvořeno deset datasetů a tyto datasety byly rozděleny na trénovací a testovací data. Dále byla natrénována neuronová síť na trénovacích datech ze všech datasetů a spočítána RMSE na testovacích datech. Tento postup proběhl celkem patnáctkrát, při každém průchodu se měnil počet neuronů ve

skryté vrstvě v intervalu 15−30. Pro každou členitost byla spočítána průměrná RMSE ze všech datasetů. Nakonec byl spočítán průměr ze všech členitostí pro každé nastavení sítě. Výsledky byly seřazeny v tabulce 4.2. Jako nejlepší vyšla síť s 28 neurony ve skryté vrstvě.

Tabulka 4.2: Průměrné hodnoty RMSE při testování nastavení sítě (nnet)

počet neuronů	členitost 1	členitost 2	členitost 3	průměr
28	0.148758	0.096397	0.017947	0.087701
24	0.151135	0.096007	0.018192	0.088444
25	0.149176	0.099806	0.018237	0.089073
26	0.154605	0.095057	0.017955	0.089206
22	0.152875	0.097339	0.018894	0.089703
30	0.157484	0.095408	0.018022	0.090304
19	0.154560	0.097975	0.018826	0.090454
20	0.157060	0.097911	0.018621	0.091197
18	0.156477	0.098647	0.019657	0.091594
29	0.163756	0.095861	0.018037	0.092551
21	0.161621	0.097511	0.019381	0.092838
16	0.161058	0.099874	0.019209	0.093380
23	0.167908	0.095541	0.019640	0.094363
27	0.158991	0.106556	0.018620	0.094722
17	0.166058	0.100206	0.019149	0.095138
15	0.164467	0.103390	0.019708	0.095855

4.2.2 Výběr nastavení pro balíček neuralnet

Výběr nastavení pro sítě z tohoto balíčku byl obdobný jako v balíčku *nnet*. Rozdíl byl v tom, že balíček *neuralnet* umožňuje trénovaní sítí s více skrytými vrstvami. Postupně byly vyzkoušeny sítě s jednou až čtyřmi skrytými vrstvami. Počet neuronů v první skryté vrstvě se vždy pohyboval v intervalu $15 - 30$. Počet neuronů v dalších skrytých vrstvách už byl pevně nastaven a byl vybírán z řady $5, 10, 15, 20, 25$. Pro každé testované nastavení byla opět spočítána průměrná RMSE ze všech datasetů. Jako nejlepší byla vybrána síť se čtyřmi skrytými vrstvami a počty neuronů $24, 15, 10, 5$. Výsledky z testování této sítě viz. tabulka 4.3.

4.2.3 Výběr nastavení pro sítě v programu GRASS GIS

Základem pro výběr nastavení byly předchozí testy v programu R Project. Další zdrojem byl poster autora modulu *ann.** (Netzel, 2011). Jako první byla vyzkoušena síť, která vyšla jako nejlepší pro balíček *neuralnet*, protože modul *ann.** také umožňuje trénovat sítě s více skrytými vrstvami. Také v posteru byla k interpolaci využívána síť se čtyřmi skrytými vrstvami. Avšak čas učení této sítě byl velmi dlouhý a ačkoliv

Tabulka 4.3: Průměrné hodnoty RMSE při testování nastavení sítě (neuralnet)

počet neuronů	členitost 1	členitost 2	členitost 3	průměr
24	0.149159	0.098706	0.029940	0.092602
16	0.153596	0.098004	0.029210	0.093603
15	0.148941	0.100402	0.032237	0.093860
22	0.150838	0.097616	0.033587	0.094014
20	0.149615	0.098070	0.035774	0.094486
30	0.150110	0.098537	0.035594	0.094747
17	0.152036	0.100431	0.031792	0.094753
23	0.152735	0.099836	0.031717	0.094763
27	0.155015	0.096519	0.033971	0.095169
29	0.150820	0.098389	0.036693	0.095301
26	0.152207	0.099028	0.034765	0.095333
28	0.151465	0.098248	0.038276	0.095996
21	0.153177	0.099907	0.036171	0.096418
19	0.150610	0.102501	0.036541	0.096551
18	0.152551	0.101085	0.038194	0.097277
25	0.156908	0.101286	0.033683	0.097292

chyba na trénovacích datech z počátku klesala, po určitém počtu iterací začala znovu stoupat a výsledná RMSE byla příliš velká.

Dále byly zkoušeny sítě s nižším počtem skrytých vrstev a ačkoliv učení v tomto případě probíhalo velmi rychle, výsledná RMSE byla opět nepřiměřeně vysoká, její hodnoty byly o několik řádů vyšší než bylo očekáváno. Jako optimální se ukázalo použít síť se třemi skrytými vrstvami.

Pro tuto síť byly nejdříve vyzkoušeny počty neuronů v intervalu $15 - 30$ ve všech skrytých vrstvách. Tento počet byl však nedostatečný, postupně tedy byly počty neuronů zvyšovány a byla sledováno klesání RMSE. Toto pokračovalo, dokud RMSE nezačala stoupat. Tímto postupem bylo nakonec dosaženo sítě se třemi skrytými vrstvami a počtem neuronů $32, 38, 27$.

Pro účely testování dalších možností modulu *ann.** (viz. 4.3.2) byla obdobným způsobem vytvořena nová síť se třemi skrytými vrstvami a počtem neuronů $20, 25, 17$

4.3 Interpolace v softwaru GRASS GIS 6.4

Pro výpočty pomocí neuronových sítí slouží v tomto programu modul *ann.**. Tento modul není součástí instalace programu GRASS GIS a není uložen v oficiálních repozitářích dostupných modulů pro GRASS GIS. Bylo tedy nutné ho stáhnout zvlášť z adresy `http://www.wgug.org/index.php?option=com_content&view=article&id=56&Itemid=9` a ručně přidat mezi normálně nainstalované skripty.

Před trénováním neuronové sítě bylo nutné normalizovat trénovací data ze všech

datasetů podle vzorce

$$z'_i = \frac{z_i - min_z}{max_z - min_z} \tag{12}$$

kde z_i je původní hodnota, min_z je nejnižší a max_z nejvyšší hodnota z v jednom datasetu. Metody IDW a kriging normalizaci dat nepotřebovaly.

4.3.1 Neuronové sítě z rastrových dat

Dříve, než bylo možné provádět interpolaci, bylo nutné připravit data. Modul *ann.** vyžaduje vstupní data v rastrovém formátu. Původní data (4.1) byla vytvořena jako vektorové body. Aby bylo možné provádět alespoň nějaké srovnání metod, byly jako vstupní rastrová data použity rastry, které vznikly interpolací pomocí kringingu v softwaru R Project. Tímto bylo zajištěno, že interpolace v modulu *ann.** proběhne na datech s téměř stejnými nebo podobnými vlastnostmi jako interpolace pomocí ostatních metod. Rastry vytvořené metodou kriging byly vybrány proto, že vykazovaly nejnižší RMSE.

U vybraných rastrových data byla nejprve v programu R Project normalizována hodnota z a pak byly tyto rastry importovány do programu GRASS GIS pomocí příkazu *r.in.arc*. Podle těchto rastrů byla příkazem *g.region* zadáno rozlišení rastru 400×400 pixelů. Dalším krokem bylo rozdělení importovaného rastru na tři rastry z nichž každý obsahoval pouze jednu hodnotu x nebo y nebo z. Toto rozdělení zajistil příkaz *r.mapcalc*.

Skript *ann.data.rast.py* vytváří soubor s příponou .dat, který slouží k učení neuronové sítě. K vytvoření tohoto souboru bylo nutné kromě tří rastrů obsahujících hodnoty x, y, z vytvořit také určitý počet náhodně rozložených vektorových bodů, kterým byla přiřazena hodnota rastru z. Tyto body byly vytvořeny příkazem *v.random*. Byly vytvořen čtyři vektorové bodové vrstvy pro každou členitost povrchu s počty bodů 500, 1000, 2000, 3000. Příkazem *v.what.rast* jim byly přiřazeny hodnoty příslušného rastru z.

```
ann.data.rast.py in = x,y out = z~output = pt3000xy vector = body3000
```

K vytvoření a uložení nastavení sítě slouží skript *ann.create.py*. Bylo třeba zadat počet vstupních a výstupních neuronů (parametry *in = 2* a *out = 1* - v tomto pořadí). Do parametru *hidd* byly zadány počty neuronů, které v testech vyšly jako nejlepší. Nakonec byla síť pojmenována. Aktivační funkce výstupního neuronu byla nastavena jako lineární.

```
ann.create.py -l in = 2 hidd = 32, 38, 27 out = 1 net = sit_n
 learn_rate = 0.7
```

K učení sítě byl využit skript *ann.learn.py*. Byla zadáno jméno vytvořená síť, soubor s příponou .dat, který sloužil jako trénovací data, maximální počet iterací, požadovaná chyba na trénovacích datech a název souboru, do kterého byla naučená síť uložena.

```
ann.learn.py net = sit_n data = pt3000xy max_iter = 50000 error = 0.001
 output = sit_nl
```

Průběh učení bylo možno sledovat v terminálu, kde se po každé 1000 iteraci vypisovala aktuální chyba.

Samotná interpolace byla provedena pomocí skriptu *ann.run.rast.py*. V příkazu byla zadána vstupní data (souřadnice x a y), název souboru s naučenou sítí a název výsledného rastru. Interpolace proběhla celkem dvanáctkrát, pro výsledné hodnocení byly vybrány ty rastry, které měly nejnižší RMSE.

```
ann.run.rast.py in = x,y net = sit_nl output = sit3000b
```

4.3.2 Neuronové sítě z vektorových dat

Protožeby nebylo možné srovnávat hodnotu RMSE u povrchů, které vytvořila neuronová síť z rastrů vytvořených pomocí krigingu, byl vytvořen skript v Jave, který převáděl vektorové body z formátu csv do souborů s příponou .dat, které využívá neuronová síť k učení. Takto bylo možné vynechat skript *ann.data.rast.py*, který vytváří soubor .dat z rastrových dat.

Do souborů s příponou .dat byly převedeny soubory s trénovacími daty (*train.set*). Na tomto souboru byla nejprve pomocí skriptu *ann.learn.py* natrénována neuronová síť se třemi skrytými vrstvami a počtem neuronů 32, 38, 27. Učící koeficient se měnil podle členitosti dat. Na data s nižší členitostí bylo nutné použít menší učící koeficient.

Jako druhá možnost byla skriptem *ann.learn.py* natrénována síť s počtem neuronů 20, 25, 17. Učící koeficient se v tomto případě neměnil a byl nastaven na hodnotu 0.4. Tato síť byla výrazně lepší než předchozí. Síť se naučila ve velmi krátkém čase a výsledná RMSE byla srovnatelná s ostatními metodami.

4.3.3 IDW

Pro interpolaci metodou IDW byla použita data vytvořená v části 4.1. Tato data byla importována příkazem *v.in.ascii*. Před interpolací bylo nutné upravit pomocí příkazu *g.region* rozlišení rastru na 400×400 pixelů. Nastavení parametrů *npoints* (počet bodů) a *power* bylo zvoleno stejné jako při interpolaci v R Project.

```
v.surf.idw input = train.set output = idw_rastr column =
z~npoints = 18 power = 1.0
```

4.3.4 Kriging

Software GRASS GIS ve verzi 6.4 nemá vlastní modul pro kriging, ale využívá spojení s R Project a provádí kriging pomocí jeho nástrojů. Proto byla interpolace pomocí krigingu provedena pouze jednou v části 4.4.3)

Výsledné zpracování

Všechny interpolované rastry s rozlišením 400×400 pixelů byly exportovány z programu GRASS GIS příkazem *r.out.xyz*. Takto bylo vytvořeno patnáct ascii souborů s příponou .txt, které obsahovaly 160 000 řádků a 3 sloupce s hodnotami x, y a z. Tyto soubory byly poté importovány do programu R Project, kde z nich byly opět vytvořeny matice s rozlišením 400×400. Tyto matice pak byly použity k dalším testům. Před dalším zpracováním bylo nutné převést hodnoty z z datasetů interpolovaných pomocí neuronových sítí zpět na normální hodnoty pomocí vzorce

$$z_i = z_i' \, (max_z - min_z) + min_z \tag{13}$$

kde z_i' je interpolovaná hodnota a max_z a min_z jsou maximální a minimální hodnota z původních trénovacích dat.

4.4 Interpolace v softwaru R Project

Software R Project byl vybrán pro srovnání interpolace pomocí neuronových sítí, protože nabízí uživateli hned dva balíčky, které umožňují s neuronovými sítěmi pracovat, aniž by sám uživatel musel programovat, dále má velké výpočetní možnosti a výhodou je i open source licence programu.

Nejprve byly vytvořeny tři datasety (viz. 4.1). Tyto datasety byly rozděleny na trénovací a testovací data a pojmenovány *train.set* a *test.set*. Z testovacích dat byly vytvořeny soubory dat obsahující pouze souřadnice, byly pojmenovány *test.coord*. Tato data byla použita pro výpočet RMSE. Dále byla vytvořena matice se 160 000 řádky a 2 sloupci, která obsahovala souřadnice x a y. Byla označena jako *grid*. Do této matice byla spočítána interpolace.

Před trénováním neuronové sítě bylo nutné normalizovat trénovací data ze všech datasetů podle vzorce 12.

4.4.1 Neuronové sítě

Použité balíčky pro trénování neuronových sítí mají velmi podobnou strukturu a způsob použití. Balíček *neuralnet* nabízí více možností při nastavování parametrů, balíček *nnet* je sice jednodušší ale výsledky výpočtů jsou srovnatelné s balíčkem *neuralnet*.

Balíček nnet Tento balíček umožňuje použít sítě pouze s jednou skrytou vrstvou a učícím algoritmem backpropagation. Učící algoritmus je pevně daný a nelze jej měnit. Funkce *nnet* provádí trénink sítě.

```
sit_nnet <- nnet(z~x+y, data = train.set, size = 28, linout = TRUE,
 rang = 0.7, maxit = 100000, trace = FALSE, abstol = 0.0001)
```

První parametr funkce *nnet* se nazývá *formula* a rozlišuje vstupní data (za vlnovkou) a očekávaný výstup sítě (před vlnovkou). Údaje získává z parametru *data*, bylo tedy nutné označit stejnými znaky zápis ve formuli a názvy sloupců v souboru s trénovacími daty. Parametr *size* určuje počet neuronů ve skryté vrstvě. Na základě předchozích testů byla zvolena hodnota 28. Pomocí parametru *linout = TRUE* byla nastavena aktivační funkce výstupního neuronu jako lineární. *Rang* obsahuje číslo, podle kterého se náhodně inicializují váhy před prvním průběhem sítě. Parametr *maxit* udává maximální počet iterací - průběhů sítě při učení. Pomocí funkce *predict* je prováděn výpočet.

```
vystup <- predict(sit_nnet, grid, type = "raw")
vysledek <- cbind(grid, vystup)
```

Pro výpočet je třeba zadat proměnnou, ve které je uloženo nastavení naučené sítě, v tomto případě *sit_nnet*, dále vstupy sítě (souřadnice), pro které se výpočet provádí, a posledním parametrem je *type*, který udává, jaký výstup bude spočítán. Hodnota *raw* znamená, že výstup bude přesně to číslo, které bylo vypočteno. Další možností je mít klasifikované výstupy. Výstupem funkce *predict* byl tedy při tomto nastavení vektor čísel, ke kterému bylo třeba přiřadit souřadnice a teprve poté bylo možné s tímto výsledkem dále pracovat.

Balíček neuralnet Tento balíček umožňuje trénovat sítě s více skrytými vrstvami a různými učícími algoritmy. V této práci byl použit algoritmus RPROP (resilient backpropagation), který je nastaven jako výchozí. Při použití klasického algoritmu backpropagation síť vykazovala chyby, které se nepodařilo odstranit. Proto bylo použito implicitní nastavení učícího algoritmu.

```
sit_neuralnet <- neuralnet(z~x+y, data = train.set,
 hidden = c(24, 15, 10, 5),threshold = 0.05, stepmax = 100000,
 rep = 1, err.fct = "sse", linear.output = TRUE)
```

Parametry funkce *neuralnet* jsou velmi podobné těm z funkce *nnet*. Parametry *formula* a *data* fungují stejně jako u předchozí funkce. K definování skrytých vrstev a počtu neuronů v nich se používá parametr *hidden*. *Threshold* udává požadovanou chybu, které by měla síť při učení dosáhnout. Maximální počet iterací byl nastaven na 100000 (*stepmax*), aktivační funkce výstupních neuronů byla nastvena jako lienární (*linear.output = TRUE*). K výpočtu pomocí natrénované sítě slouží funkce *compute*.

```
vystup <- compute(sit_neuralnet, grid, rep=1)
vysledek <- cbind(grid, vystup$net.result)
```

Parametry nutné pro výpočet jsou dva. Prvním z nich je proměnná, která uchovává nastavení sítě - *sit_neuralnet*, a dále nové souřadnice, pro které výpočet probíhá. Výstupem funkce *compute* byla matice s mnoha sloupci, která zahrnovala průběžné hodnoty na jednotlivých neuronech a teprve poslední sloupec obsahoval výsledné spočítané hodnoty. Tyto hodnoty bylo opět nutné přiřadit k souřadnicím.

4.4.2 IDW

V softwaru R Project implementuje tuto metodu několik balíčků, v této práci byl použit balíček *gstat* s funkcí *idw*.

```
idw_result <- idw(z~x+y, locations = train.set, newdata = grid,
 nmax = 18, idp = 1.0)
```

Funkce *idw* opět využívá parametr *formula* k rozlišení souřadnic a hodnot, které budou interpolovány. Parametr *locations* obsahuje data, ze kterých probíhá interpolace a parametr *newdata* udává nové souřadnice. Počet bodů, které jsou použity k interpolaci, se nastavuje pomocí parametru *nmax* (byla zvolena hodnota 18). Číslo *p* (power) je zadáno v parametru *idp*.

Výstupem z funkce *idw* byl objekt *SpatialPointsDataFrame*, který bylo nutno převést na objekt *DataFrame*, pracovně označený jako *a* se kterým bylo možné dále pracovat. Tento objekt obsahoval čtyři sloupce, v prvních dvou byly souřadnice, další obsahoval interpolované hodnoty a v posledním byly hodnoty rozptylu. Pro další práci byly důležité pouze první tři sloupce, proto byly z výsledku vybrány a uloženy do nové proměnné *idw_vysledek*.

```
a <- as.data.frame(idw_result)
idw_vysledek <- as.data.frame(a[1:3])
```

4.4.3 Kriging

Pro metodu kriging existuje v R Project celá řada balíčků a funkcí. Pro účely této práce byl zvolen balíček *automap* a v něm funkce *autoKrige*, protože provádí tzv. ordinary kriging. Další použitou funkcí byla *autofitVariogram*, která automaticky přizpůsobuje variogram.

```
variogram <- autofitVariogram(z~1, train.set, model = cModel)
kriging_result <- autoKrige(z~1, train.set, grid, model = cModel)
```

Typ variogramu, podle kterého byla provedena interpolace, byl zvolen jako sférický, neboť data, ze kterých interpolace probíhala, byla vytvořena právě s tímto variogramem (viz. 4.1). Dalšími parametry, které bylo nutno zadat, byla původní data (train.set) a nové souřadnice (grid).

Výstupem z funkce *autoKrige* byl objekt *SpatialPointsDataFrame*, který obsahoval jak výsledky, tak údaje o nastavení variogramu a další věci, které neměly být využity v této práci. Proto byla do přechodné proměnné *b* uložena pouze část s výsledky, která měla pět sloupců a to souřadnice, výsledek interpolace, rozptyl a směrodatnou odchylku. Z proměnné *b* byly do konečného výsledku uloženy pouze souřadnice a výsledky interpolace.

```
b <- kriging_result$krige_output
kriging_vysledek <- cbind(coordinates(b), b$var1.pred)
```

Výsledné zpracování

Výsledkem po průběhu všech metod byly vždy tři nové datasety z každé metody, které měly 160 000 záznamů a obsahovaly souřadnice x a y a interpolovanou hodnotu z. Hodnoty v rastrech interpolovaných pomocí neuronových sítí byly opět převedeny na normální hodnoty podle vzorce 13.

Všechny nové datasety byly dále převedeny na matice s velikostí 400×400 záznamů a poté uloženy jako rastr s rozlišením 400×400 pixelů. Uloženy byly pomocí funkce *epi.asc* z balíčku *epiR* ve formátu ascii rastr s příponou .txt.

4.5 Výpočet RMSE a vizuální srovnání

Pro potřeby hodnocení kvality interpolace a srovnání metod bylo třeba spočítat RMSE. Výpočet probíhal v programu R podle vzorce 1.

Při výpočtu RMSE pro výstup z programu GRASS GIS u povrchů interpolovaných pomocí neuronových sítí z rastrových dat byla nejprve vytvořena příkazem *v.random* pro každou členitost vektorová vrstva náhodně umístěných bodů. Těmto bodům byla přiřazeny hodnoty z původního rastru a poté i hodnoty z výsledného rastru. Tyto body byly příkazem *v.out.ogr* exportovány ve formátu csv. Poté byly importovány do programu R, kde byly hodnoty z převedeny podle vzorce 12 a byla spočítána RMSE.

Při výpočtu RMSE pro výstup z programu GRASS GIS u povrchů interpolovaných pomocí neuronových sítí z vektorových dat byly do programu GRASS GIS importovány body z datasetů *test.set*. Příkazem *v.what.rast* k nim byly přiřazeny odpovídající hodnoty výsledného rastru. Tyto body byly převedeny zpět do R, kde byly hodnoty z výseldných rastrů převedeny vzorcem 13 a poté byla spočítána RMSE. Postup při výpočtu pro rastry vytvořené metodou IDW byl stejný, pouze hodnoty z výsledných rastrů nebyly převáděny.

Výpočet RMSE v programu R Project probíhal pro všechny metody stejným způsobem. Nejprve byla znovu provedena interpolace pomocí zvolených metod, ale tentokrát výpočet probíhal pro souřadnice z testovacích datasetů (*test.coord*). Hodnoty interpolované neuronovými sítěmi byly opět převedeny podle vzorce 13. K těmto výsledkům pak byly přiřazeny původní hodnoty z testovacího datasetu a poté byla spočítána RMSE.

Dalším způsobem srovnávání bylo vizuální porovnání výsledných rastrů odečtením těchto rastrů. Vždy byly odčítány rastry vytvořené metodami IDW a kriging od rastrů vytvořených pomocí neuronových sítí. Z toho to bylo možné pozorovat rozdíly mezi interpolačními metodami.

5 VÝSLEDKY

Tato kapitola shrnuje výsledky, kterých bylo dosaženo metodami a postupy v kapitole 4. Hlavními výsledky jsou:

- zhodnocení kvality interpolace pomocí modulu *ann.** v programu GRASS GIS 6.4-svn,
- srovnání modulu *ann.** s interpolačními metodami IDW a kriging,
- porovnání interpolace pomocí neuronových sítí v programu GRASS GIS a R Project.

5.1 Hodnocení kvality interpolace

Nejprve byla hodnocena kvalita interpolace pro neuronovou síť, která se trénovala z rastrových dat (viz. 4.3.1). Hodnoty RMSE se s klesající členitostí dat (klesajícím rozsahem hodnot z) snižovaly. Čas učení se s počtem použitých vektorových bodů většinou prodlužoval, na datech s nižší členitostí byl kratší. Čím více náhodných vektorových bodů bylo použito, tím méně iterací bylo třeba k natrénování sítě. V tabulkách 5.1, 5.2, 5.3 byly shrnuty údaje o trénování sítí. K hodnocení byly vybrány ty rastry, které měly nejnižší RMSE, tedy většinou ty, které vytvořily sítě natrénované z dat s počtem bodů 3000. Pouze v případě *členitosti 2* to bylo 2000 bodů.

Tabulka 5.1: Údaje o trénování sítě pro *členitost 1*

počet bodů	čas učení v minutách	počet iterací	RMSE
500	12	13292	0.099482
1000	22	11675	0.106778
2000	25	7901	0.088307
3000	39	7618	0.064648

Tabulka 5.2: Údaje o trénování sítě pro *členitost 2*

počet bodů	čas učení v minutách	počet iterací	RMSE
500	10	10729	0.068894
1000	18	10014	0.056571
2000	25	6998	0.039702
3000	23	3903	0.042661

Rozsah hodnot z interpolovaného rastru pro *členitost 1* byl nižší než u původního rastru (viz tabulka 5.4). Při výpočtu tedy neuronová síť neinterpolovala krajní hodnoty. Toto mohlo být způsobeno špatným rozmístěním náhodných vektorových bodů

Tabulka 5.3: Údaje o trénování sítě pro *členitost 3*

počet bodů	čas učení v minutách	počet iterací	RMSE
500	4	4863	0.010265
1000	13	9534	0.009793
2000	14	3403	0.009272
3000	11	1717	0.008466

(viz 4.3.1). Hodnota RMSE byla v tomto případě 0.0646. V případě *členitosti 2* je zde opět vidět, že neuronová síť vynechala krajní hodnoty, v tomto případě je však rozdíl v menší, náhodné vektorové body byly pravděpodobně rozmístěny lépe. Hodnota RMSE byla 0.0427. Pro *členitost 3* bylo chování sítě podobné jako v předchozích dvou případech. Hodnota RMSE byla 0.0085. Pokud byla porovnána procentuální hodnota

Tabulka 5.4: Rozsah hodnot z původních a interpolovaných dat

	rozsah původních dat	rozsah interpolovaného rastru
členitost 1	-0.6924255 - 1.034648	-0.6198587 - 0.8887726
členitost 2	-0.5086115 - 0.863945	-0.4962887 - 0.7969054
členitost 3	-0.1734903 - 0.124689	-0.1563402 - 0.1131727

RMSE podle rozsahu dat, bylo zjištěno, že neuronová síť interpolovala povrchy s chybou lišící se pouze o 1 % (viz tabulka 5.5).

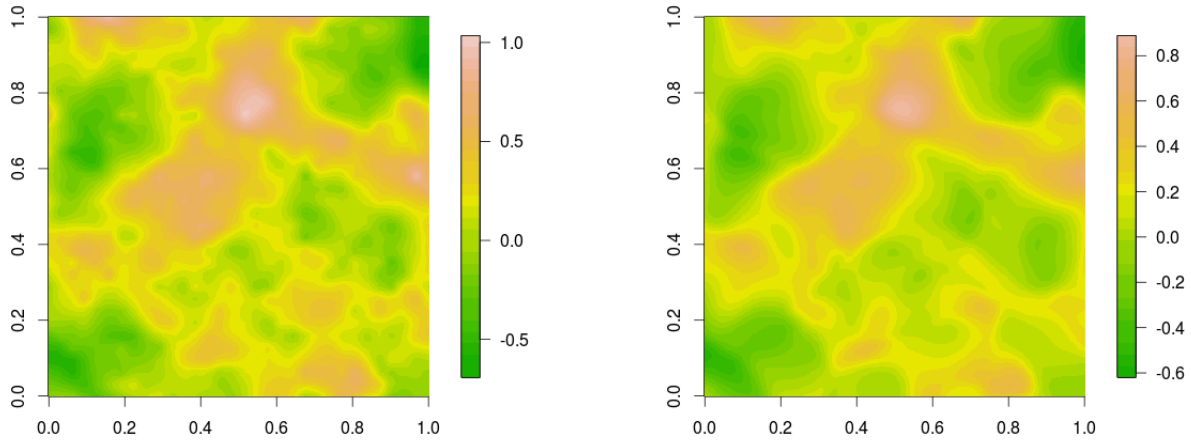
Tabulka 5.5: Procentuální srovnání RMSE pro všechny členitosti

	procentuální hodnota RMSE
členitost 1	4.2852
členitost 2	3.2989
členitost 3	3.1412

Kvalita interpolace byla dále hodnocena vizuálním porovnáním výsledných rastrů. Neuronová síť z modul *ann.**, která byla natrénována pomocí rastrových dat (viz. 4.3.1) se dokázala poměrně dobře přizpůsobit a výsledný rastr se velmi podobá rastru původnímu. Obrázek 5.1 znázorňuje původní a interpolovaný rastr pro *členitost 1.* Původní i výsledné rastry pro *členitost 2* a *členitost 3* lze nalézt v příloze 1.

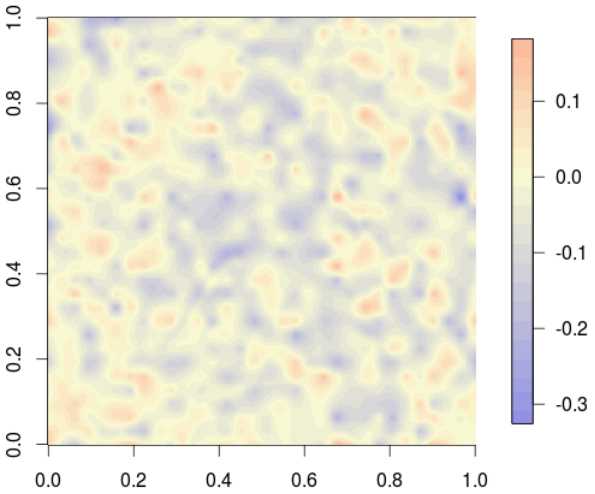
Obrázek 5.2 ukazuje rozdíly v hodnotách z mezi novým a původním rastrem. Od nově interpolovaného rastru byl odečten původní rastr. Rastr vytvořený neuronovou sítí měl hodnoty z většinou nižší než původní rastr. Průměrná hodnota rozdílu byla -0.03145 , nejvyšší hodnoty rozdílů byly -0.32560 a 0.18220.

Neuronová síť z modulu *ann.**, která byla natrénována pomocí vektorových dat (viz 4.3.2) měla podobné chování, jako síť trénovaná z rastrových dat. Při trénování



(a) původní rastr (b) interpolovaný rastr

Obrázek 5.1: Porovnání původního rastru s rastrem interpolovaným pomocí neuronové sítě (*členitost 1*)



Obrázek 5.2: Rozdíl hodnot z mezi novým a původním rastrem (*členitost 1*)

sítě z vektorových dat bylo nutné změnit koeficient učení. Pokud byl použit příliš vysoký, chyba na trénovacích datech buď neklesla ve stanoveném počtu iterací, nebo opakovaně klesala a stoupala aniž by klesla pod stanovené minimum, dalším nežádoucím případem byl náhlý obrovský nárůst chyby na nepřiměřeně velké hodnoty. Tabulka 5.6 shrnuje údaje o trénování neuronové sítě s počtem neuronů 32, 38, 27 na všech členitostech. Údaje o trénování neuronové sítě s počtem neuronů 20, 25, 17 shrnuje tabulka 5.7.

Při interpolaci neuronová síť opět vynechávala krajní hodnoty. V případě *členitosti 2* byla dolní hranice intervalu hodnot z nižší než u původních dat. Rozsah původních a nově interpolovaných dat lze nalézt v tabulce 5.8, která uvádí tyto hodnoty pro rastr vytvořený sítí s počtem neuronů $20, 25, 17$. Hodnoty RMSE byly pro *členitost 1* 0.1407, pro *členitost 2* 0.0982, pro *členitost 3* 0.0252.

Hodnoty RMSE pro rastry vytvořené neuronovou sítí natrénovanou z vektorových

Tabulka 5.6: údaje o trénování sítě (32, 38, 27) pro všechny členitosti

	čas učení v minutách	počet iterací	učící koeficient	RMSE
členitost 1	42	32494	0.7	0.212881
členitost 2	47	36136	0.4	0.126754
členitost 3	14	3403	0.1	0.017628

Tabulka 5.7: Údaje o trénování sítě (20, 25, 17) pro všechny členitosti

	čas učení v minutách	počet iterací	učící koeficient	RMSE
členitost 1	7	10504	0.4	0.140651
členitost 2	3	5021	0.4	0.098192
členitost 3	2	3104	0.4	0.025206

Tabulka 5.8: Rozsah hodnot z původních a interpolovaných dat u sítě (20, 25, 17)

	rozsah původních dat	rozsah interpolovaného rastru
členitost 1	-0.7396643 - 1.090838	-0.6786178 - 0.9450067
členitost 2	-0.5203077 - 0.893018	-0.6193843 - 0.6952054
členitost 3	-0.2012766 - 0.141199	-0.1532334 - 0.0899525

dat byly vyšší než u předchozí sítě, ale procentuálně se příliš nelišily. V tabulce 5.9 je zaznamenána procentuální hodnota RMSE. Pro *členitost 3* je vyšší, neboť zvolené parametry sítě nebyly úplně vhodné pro tuto členitost dat.

Tabulka 5.9: Procentuální srovnání RMSE pro všechny členitosti (n. síť 20, 25, 17)

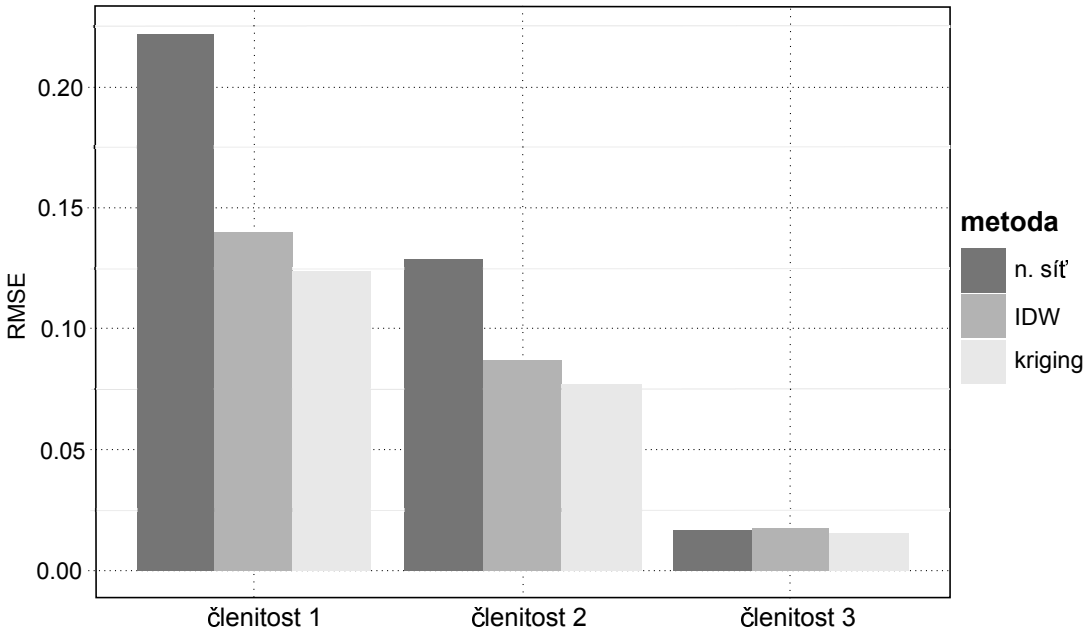
	procentuální hodnota RMSE
členitost 1	8.6658
členitost 2	7.4700
členitost 3	10.3624

5.2 Srovnání modulu *ann.** s metodami IDW a kriging

Pro srovnávání metod byly použity výsledky neuronových sítí, které byly natrénovány pomocí vektorových dat. Pro trénování sítě z rastrových dat totiž sloužil jako původní trénovací rastr ten, který byl vytvořen pomocí krigingu. Metody byly srovnávány podle hodnoty RMSE, dále z hlediska časové náročnosti a uživatelské přívětivosti.

5.2.1 Srovnání podle RMSE

Obrázek 5.3 srovnává hodnoty RMSE výsledných rastrů ze všech metod. Síť použitá v tomto srovnání byla ta s počty neuronů 38, 32, 27. Hodnoty RMSE z rastrů vytvořených pomocí sítě jsou v případě *členitosti 1* a *členitosti 2* vyšší než u ostatních metod. Toto bylo způsobeno špatným nastavením parametrů sítě, síť se pravděpodobně přetrénovala. Naopak pro *členitost 3* bylo toto nastavení v pořádku a hodnoty RMSE byly srovnatelné s ostatními metodami. V tabulce 5.10 jsou zaznamenány hodnoty RMSE s přeností na čtyři desetinná místa.



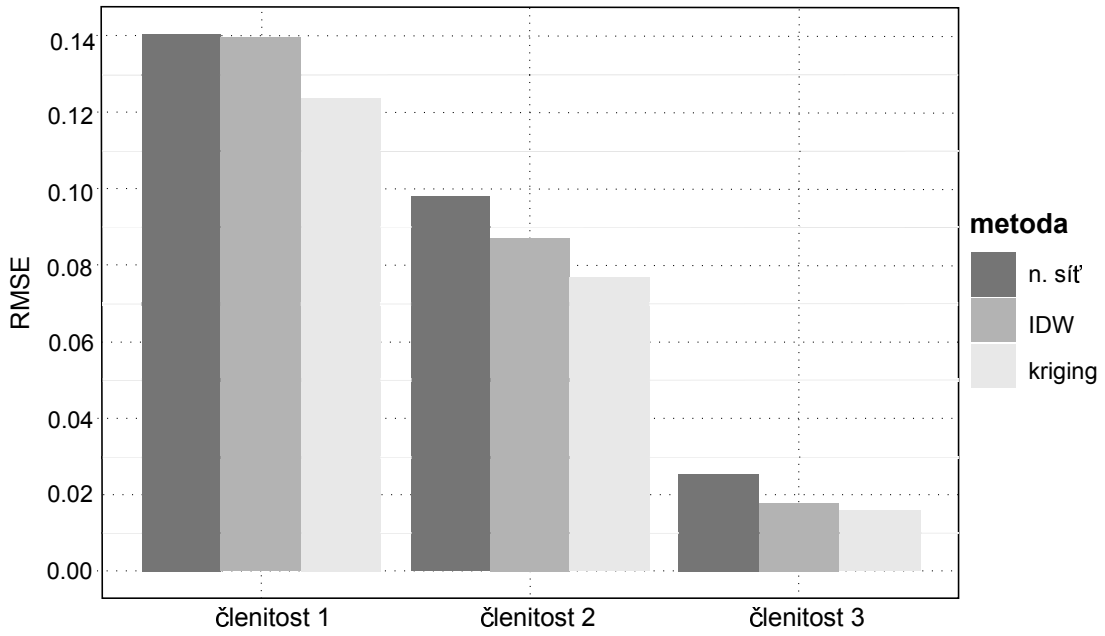
Obrázek 5.3: Srovnání RMSE pro všechny členitosti pro GRASS GIS (síť 32, 38, 27)

Tabulka 5.10: Hodnoty RMSE pro všechny členitosti pro GRASS GIS

	n. síť (32, 38, 27)	IDW	kriging
členitost 1	0.2221	0.1398	0.1240
členitost 2	0.1285	0.0874	0.0770
členitost 3	0.0172	0.0177	0.0158

Obrázek 5.4 opět srovnává všechny metody, síť použitá v tomto srovnání má počty neuronů 20, 25, 17. Hodnoty RMSE byly v tomto případě srovnatelné u všech metod, ale pro *členitost 2* a *3* byla hodnota RMSE u rastru interpolovaného pomocí neuronové sítě vyšší. Vyšší hodnota mohla být způsobena nastavením parametrů sítě, které se nehodilo pro data z této členitosti. Tabulka 5.11 zaznamenává hodnoty RMSE s přesností čtyř desetinných míst.

Na uměle vytvořených datech (viz 4.1) dosahovaly rastry vytvořené pomocí neuronových sítí vyšších hodnot RMSE než metody IDW a kriging, přestože původní



Obrázek 5.4: Srovnání RMSE pro všechny členitosti pro GRASS GIS (síť 20, 25, 17)

Tabulka 5.11: Hodnoty RMSE pro všechny členitosti pro GRASS GIS

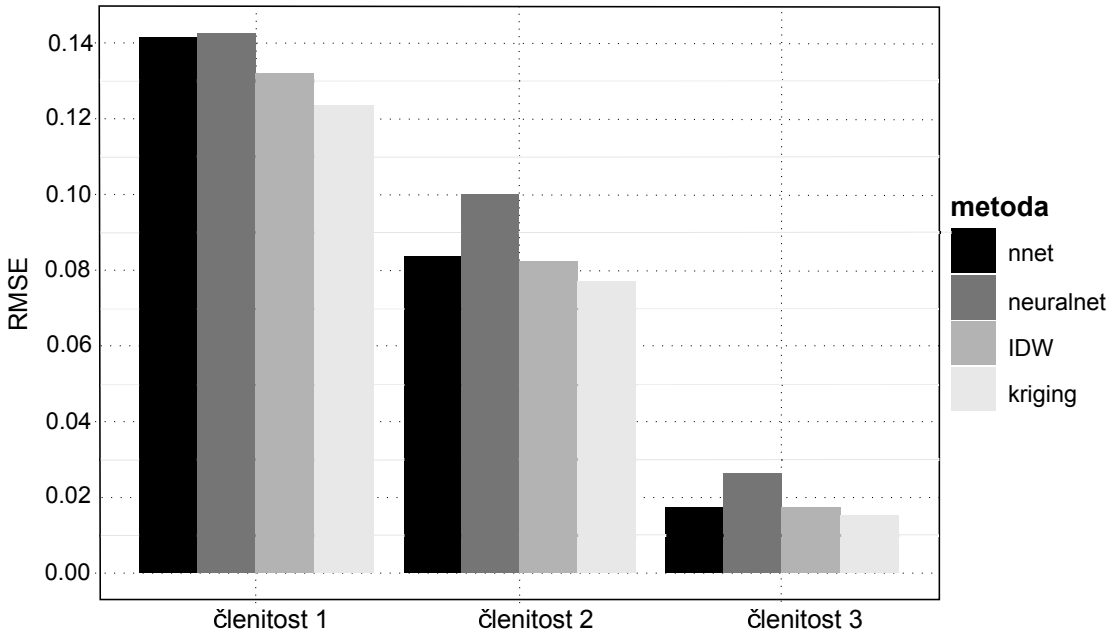
	n. síť (20, 25, 17)	IDW	kriging
členitost 1	0.1407	0.1398	0.1240
členitost 2	0.0982	0.0874	0.0770
členitost 3	0.0252	0.0177	0.0158

předpoklad byl, že neuronové sítě by mohly mít lepší výsledky. Důvodů je několik: Ačkoliv bylo nastavení sítí testováno, je možné, že byly zvoleny nevhodné parametry vzhledem k povaze dat a sítě se nenatrénovaly dobře. Důvodem horších výsledků sítě může být i to, že jsou zadány pouze dva vstupní parametry, ze kterých se síť učí.

Pro srovnání je zde na obrázku 5.5 uvedeno i porovnání metod v programu R Project. Neuronové sítě z obou balíčků měly na všech členitostech vyšší hodnotu RMSE než metody IDW a kriging. Výrazněji vyšší hodnoty u balíčku *neuralnet* v případě *členitosti 2, 3* byly způsobeny přetrénováním sítě. Hodnoty RMSE u neuronové sítě z balíčku *nnet* pro *členitost 2* a *3* se nejvíce podobají hodnotám RMSE u ostatních metod. Tabulka 5.12 ukazuje hodnoty RMSE s přesností na čtyři desetinná místa.

Tabulka 5.12: Hodnoty RMSE pro všechny členitosti pro R Project

	nnet	neuralnet	IDW	kriging
členitost 1	0.1418	0.1427	0.1325	0.1240
členitost 2	0.0843	0.1002	0.0828	0.0770
členitost 3	0.0177	0.0264	0.0176	0.0158



Obrázek 5.5: Srovnání RMSE pro všechny členitosti pro R Project

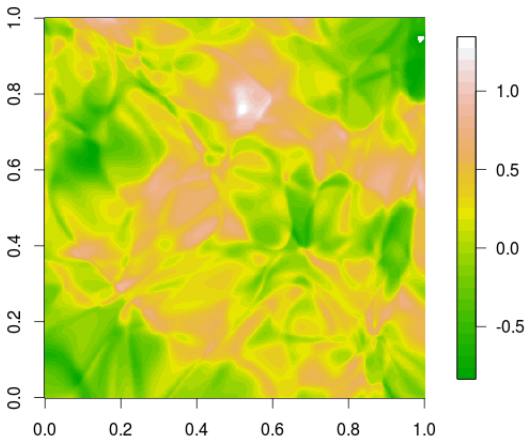
Obrázek 5.6 ukazuje výsledné rastry interpolované pomocí neuronových sítí a metod IDW a kriging v programu GRASS GIS pro *členitost 1.* Výsledné rastry pro tuto i další dvě členitosti lze nalézt v příloze 1.

Obrázek 5.7 ukazuje rozdílové rastry pro *členitost 1.* Dílčí obrázky (a) a (b) ukazují rozdíly mezi rastry vytvořenými neuronovou sítí s počtem neuronů 32, 38, 27 a ostatními metodami. Maximální hodnoty v mezi sítí a metodou IDW byly −0.784100 a 0.707100 a průměrná hodnota rozdílu byla 0.006563. Rastr interpolovaný pomocí IDW měl tedy spíše nižší hodnoty než ten interpolovaný pomocí neuronové sítě. Maximální hodnoty mezi sítí a metodou kriging byly −0.666400 a 0.686100, průměrná hodnota byla 0.005321. Rastr interpolovaný pomocí krigingu měl tedy také spíše nižší hodnoty.

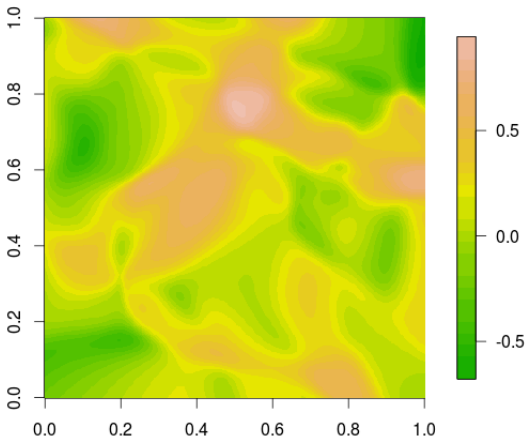
Dílčí obrázky (c) a (d) ukazují rozdíly mezi rastry vytvořenými neuronovou sítí s počtem neuronů 20, 25, 17 a dalšími dvěma metodami. Maximální rozdíly mezi sítí a IDW byly −0.3207000 a 0.3265000, průměrná hodnota rozdílu byla −0.0030340. Maximální rozdíly mezi sítí a krigingem byly −0.272800 a 0.305600, průměrná hodnota rozdílu byla −0.001977. Rastry vytvořené pomocí metod IDW a kriging měly v tomto případě spíše vyšší hodnoty. Rozdíly mezi touto sítí a metodami IDW a kriging byly nižší než v prvním případě. Všechny další rozdílové rastry i statistické tabulky lze nalézt v příloze 2 a 3.

5.2.2 Srovnání podle časové náročnosti

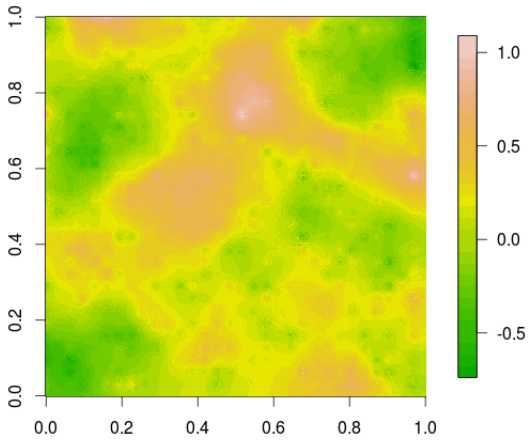
Čas potřebný k provedení interpolace ukazuje tabulka 5.13. Tyto hodnoty zahrnují u sítí čas potřebný k natrénování a délku trvání výpočtu, u ostatních metod se jedná pouze o čas potřebný k výpočtu. U neuronových sítí bylo časově náročné hlavně učení



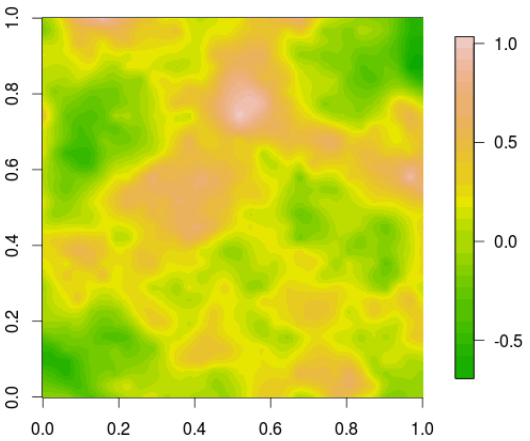
(a) síť 32, 38, 27



(b) síť 20, 25, 17



(c) IDW



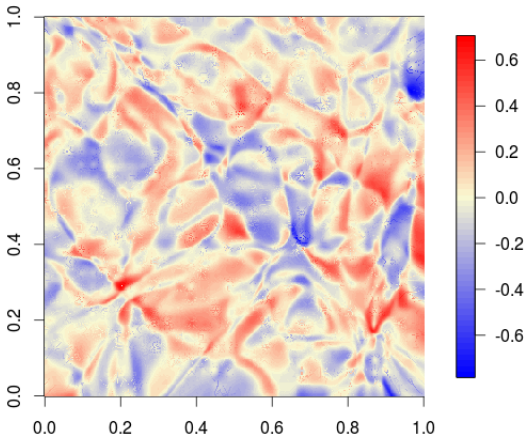
(d) kriging

Obrázek 5.6: Porovnání interpolovaných rastrů pro *členitost 1*

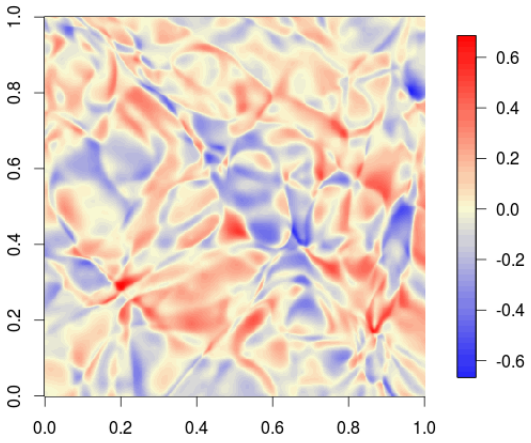
sítě. Pokud byly zvoleny špatné parametry, učení trvalo velmi dlouho - viz první sloupec tabulky. Naopak pokud byly parametry zvoleny lépe, čas učení se výrazně zkrátil. S klesající členitostí dat klesala i doba nutná k natrénování sítě. Samotný výpočet pomocí naučené sítě byl velmi rychlý, netrval déle než pár vteřin. Členitost dat neměla vliv na rychlost výpočtu ani u jedné metody. Nejdelší výpočetní čas měla metoda kriging. Nejrychlejší interpolační metodou byla metoda IDW. Interpolace pomocí neuronových sítí trvala nejdelší dobu, především kvůli dlouhému učení sítě.

Tabulka 5.13: Čas potřebný k provedení interpolace

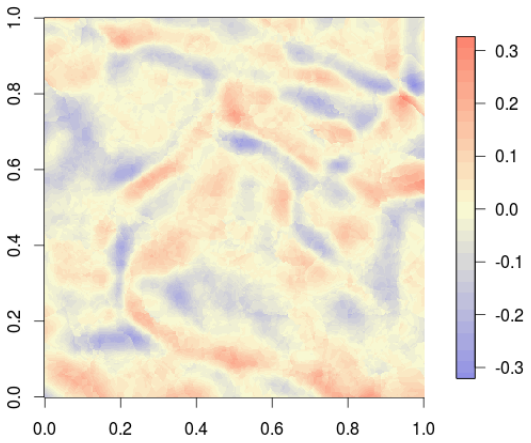
	čas v minutách			
	n. síť (32, 38, 27)	n. síť (20, 25, 17)	kriging	IDW
členitost 1	42	7	3	0.5
členitost 2	47	3	3	0.5
členitost 3	14	2	3	0.5



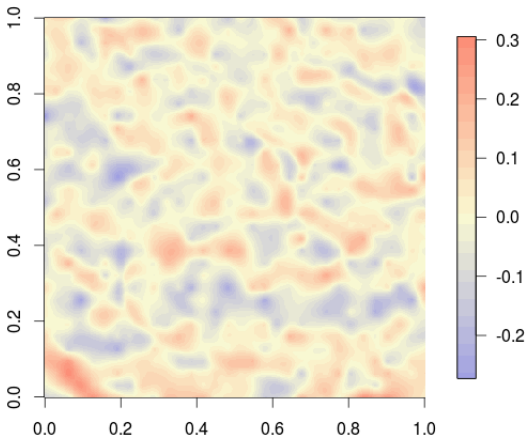
(a) n. síť 32, 38, 27 - kriging



(b)n. síť 32, 38, 27 - IDW



(c) n. síť 20, 25, 17 - kriging



(d) n. síť 20, 25, 17 - IDW

Obrázek 5.7: Rozdíl v hodnotách *z* výsledných rastrů pro *členitost 1*

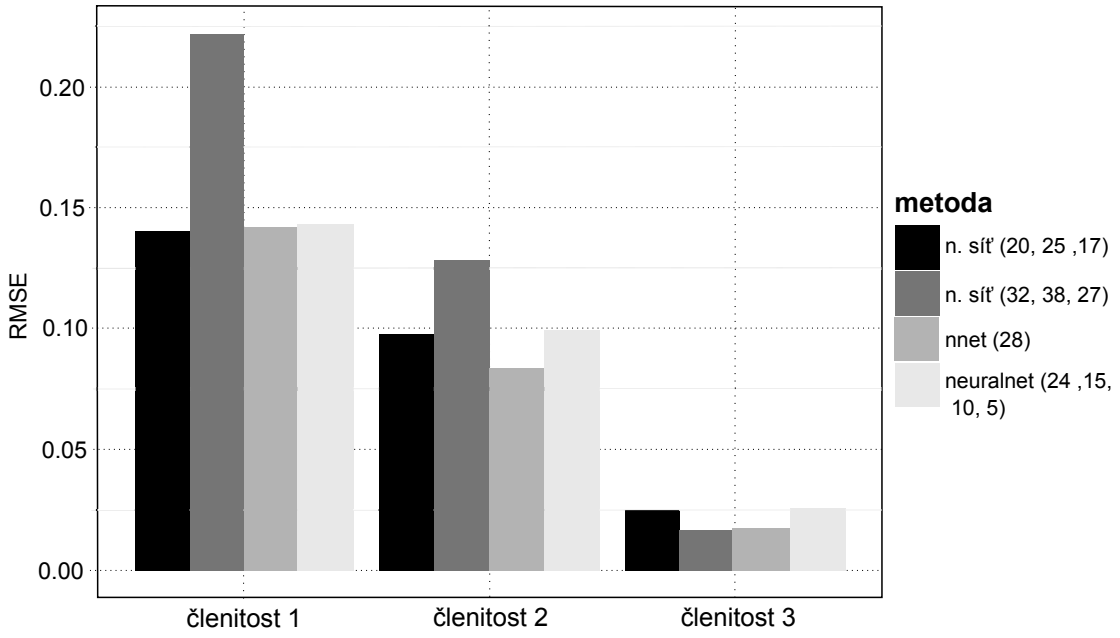
5.2.3 Srovnání podle uživatelské přívětivosti

Toto hodnocení shrnuje možnosti práce s metodou, dostupnost nápovědy, pochopitelnost metody. Metoda IDW a skript *v.surf.idw* je součástí hlavní instalace programu GRASS GIS. Lze s ní pracovat v příkazové řádce i v grafickém rozhraní. Při otevření v grafickém rozhraní je dostupný i manuál, který popisuje jednotlivá nastavení, co a jak ovlivní a na konci je i krátké teoretické shrnutí metody. Metoda kriging nemá v programu GRASS GIS vlastní modul a je prováděna pomocí propojení s programem R Project. Uživatel tak nemá možnost pracovat v grafickém rozhraní, ale pouze v příkazové řádce. Manuál a návod jak pracovat s metodou je dostupný v programu R Project i na internetu a je srozumitelný a podrobný. Modul *ann.** není součástí hlavní instalace, není ani uložen v repozitáři modulů a skriptů dostupných ke stažení příkazem *g.extension*. K nalezení je na stránce `http://grasswiki.osgeo.org/wiki/AddOns`. S modulem *ann.** je možno pracovat v příkazové řádce i v grafickém rozhraní, ale manuál zde není dostupný přímo, je třeba jej otevřít zvlášť. V manuálu je popsáno, co je třeba do kterého parametru zadat, ale ne jak to ovlivní výsledek. Na rozdíl

od metody IDW nebo kriging je použití modulu *ann.** pro uživatele, kteří se nevyznají v problematice neuronových sítí, obtížnější a stručný manuál není v tomto příliš užitečný.

5.3 Srovnání interpolace pomocí neuronových sítí v programech GRASS GIS a R Project

Srovnávání bylo provedeno z několika hledisek. Pokud byly sítě srovnány podle RMSE, není zde patrná výraznější odlišnost. Obrázek 5.8 porovnává hodnoty RMSE pro dvě sítě z programu GRASS GIS a pro síť z balíčku *nnet* a *neuralnet*. Pro *členitost 1* byly hodnoty vyrovnané, kromě sítě (32, 38, 27) z programu GRASS GIS, kde došlo v důsledku nevhodného nastavení parametrů pravděpodobně k přetrénování sítě. K přetrénování této sítě došlo nejspíš i v případě *členitosti 2*. Hodnoty RMSE ostatních sítí si byly opět podobné, nejlépe na tom byla síť z balíčku *nnet*. Pro *členitost 3* byly hodnoty opět vyrovnané, jenom síť (20, 25, 17) a síť z balíčku *neuralnet* jevily známky přetrénování. Nejlepší výsledky pro tuto členitost měla síť (32, 38, 27). Použité neuronové sítě v obou programech, které byly zvoleny jako jedny z nejvhodnějších, jsou si velmi podobné podle hodnoty RMSE. Pokud by byly použity sítě s jinými parametry, není vyloučeno, že by se hodnoty RMSE lišily více nebo méně. V tabulce 5.14 byly zaznamenány hodnoty RMSE z obrázku 5.8.



Obrázek 5.8: Srovnání RMSE pro všechny členitosti pro n. sítě v programu GRASS GIS a R Project

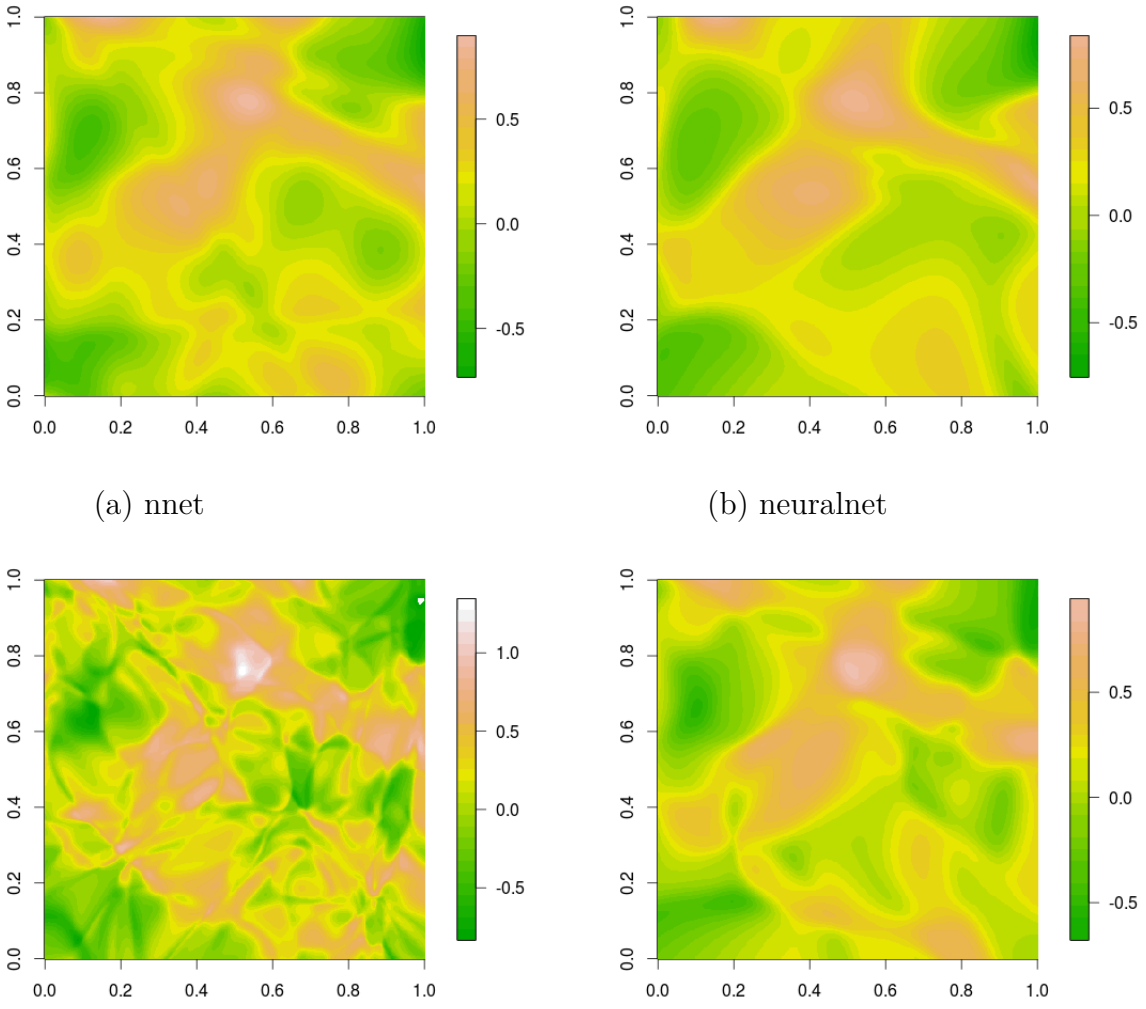
Rychlost učení sítě v modulu *ann.** a v balíčcích z programu R Project závisí především na počtu neuronů ve skrytých vrstvách, na členitosti vstupních dat a na velikosti trénovacího datasetu. U neuronové sítě z modulu *ann.** také záleží na vhod-

Tabulka 5.14: Hodnoty RMSE pro všechny členitosti pro GRASS GIS a R Project

	n. síť (20, 25, 17)	n. síť (32, 38, 27)	nnet	neuralnet
členitost 1	0.1407	0.2221	0.1418	0.1427
členitost 2	0.0982	0.1285	0.0843	0.1002
členitost 3	0.0252	0.0172	0.0177	0.0264

nosti nastavených parametrů vzhledem k vlastnostem dat. Celkově se však neuronové sítě z balíčků v R Project učí rychleji než sítě z modulu *ann.**.

Obrázek 5.9 ukazuje výsledné rastry z použitých neuronových sítí pro *členitost 1* zatímco výsledné rastry na dílčích obrázcích a, b a d jsou si vizuálně docela podobné a příliš se neliší ani hodnotou RMSE, výsledný rastr na dílčím obrázku c je od ostatních výrazně odlišný vizuálně i hodnotou RMSE. Další interpolované rastry lze nalézt v příloze 1.



(a) nnet

(b) neuralnet

(c) síť 32, 38, 27

(d) síť 20, 25, 17

Obrázek 5.9: Porovnání interpolovaných rastrů pro *členitost 1*

6 DISKUZE

Testování interpolace pomocí neuronových sítí probíhalo na uměle vytvořených datech. Tato data byla zvolena z důvodu snadné dostupnosti a také díky možnosti kontrolovat jejich vznik. Toto bylo výhodou při testování interpolace. Bylo vytvořeno 1024 bodů a z toho 724 použito jako trénovací a zbylých 300 jako testovací data. Počet bodů byl zvolen s ohledem na výpočetní nároky tak, aby interpolace netrvala příliš dlouho a zároveň byl tento počet pro interpolaci dostatečný. Po konzultaci s vedoucím práce proběhlo rozdělení na trénovací a testovací data v poměru 70 % : 30 %.

Nastavení parametrů neuronových sítí (počet skrytých vrstev, počet neuronů, aj.) v programech GRASS GIS i R Project bylo provedeno na základě testů, při kterých se zkoumalo, při jakém natavení je nejnižší výsledná RMSE. Parametry zvolené na základě testů jako nejlepší však nemusí být zcela správné. Vždy je zde určitý stupeň náhody při inicializaci vah před prvním průchodem sítě. Testování parametrů mohlo být provedeno důkladněji, ale z důvodů velké časové náročnosti byly provedeny testy jen do určité míry důkladnosti.

Pro srovnání interpolace pomocí neuronových sítí s jinými metodami byly vybrány metody IDW a kriging. Metoda IDW byla vybrána, protože je jednou z nejjednodušších a nejznámějších interpolačních metod. Kriging byl vybrán jako známá metoda s dobrým teoretickým základem, ordinary kriging byl zvolen, protože jde o standardní a běžnou verzi krigingu. Porovnání s dalšími metodami nebylo uskutečněno kvůli omezenému rozsahu práce.

Největší nevýhodou modulu *ann.** byla nutnost vytvářet data k učení neuronové sítě z rastrových dat. Nutnost použít rastrová data byla odstraněna použitím vlastního skriptu, který převáděl data z formátu .csv do formátu .dat, ze kterého se neuronová síť učila. Tento skript není součástí modulu *ann.** a ačkoliv výsledky interpolace z dat vytvořených jeho pomocí jsou srovnatelné s ostatními metodami, nelze podle nich hodnotit celý modul *ann.**. Problémem tohoto modulu je i špatná dostupnost a chybějící návody a manuál. Ke správné instalaci a zjištění, jak přesně s modulem *ann.** pracovat byla nutná komunikace s autorem modulu.

Z časových důvodů nebylo provedeno řádné testování interpolace pomocí neuronových sítí na reálných datech. Je možné, že na takovýchto datech by byly výsledky testů zcela odlišné. Neuronové sítě v programech GRASS GIS i R Project se trénovaly a následně prováděly interpolaci pouze ze dvou vstupních parametrů x a y. Přidání dalších parametrů by mohlo zlepšit výsledky interpolace pomocí neuronové sítě.

Netzel (2011) ve svém posteru uvedl, že pracuje na rozšíření modulu *ann.** i pro vektorová data. Datum dokončení není známo. Pokud bude tento modul vytvořen, lze předpokládat, že by mohl být pro interpolaci vhodnější než současný modul *ann.**. Dalším námětem do budoucna může být porovnání starého i nového modulu *ann.** s dalšími interpolačními metodami.

7 ZÁVĚR

Cílem této práce bylo zhodnotit kvalitu interpolace pomocí neuronových sítí v programu GRASS GIS a porovnání této interpolace s klasickými interpolačními metodami. Zvolenými metodami byly metody IDW a kriging. Dalším cílem bylo porovnání interpolace pomocí neuronových sítí z programu GRASS GIS s neuronovými sítěmi z jiného programu,v tomto případě s programem R Project.

V programu R Project byla nejprve vytvořena umělá data, na kterých probíhaly všechny testy. K testování interpolace bylo vybráno takové nastavení parametrů, aby poskytovalo co nejlepší výsledky u všech použitých metod.

Vlastní interpolace probíhala především v programu GRASS GIS, kde byly otestovány různé neuronové sítě na rastrových i vektorových datech. Dále zde byla provedena interpolace metodou IDW. V programu R project proběhla interpolace pomocí sítí ze dvou různých balíčků a také metodou IDW. Interpolace metodou kriging byla provedena pouze jednou, protože program GRASS GIS využívá pro kriging propojení se softwarem R Project. Pro všechny interpolované rastry byla vypočítána RMSE, podle které byla hodnocena kvalita interpolace a srovnávány metody mezi sebou. Metody byly dále srovnávány vizuálním porovnáním a odečtením výsledných rastrů.

Výsledky testování a hodnocení ukázaly, že neuronové sítě v programu GRASS GIS je možné využít k prostorové interpolaci, nicméně výsledky z těchto sítí nejsou lepší než výsledky pomocí metod IDW a kriging. Nevýhodou modulu *ann.** je práce s rastrovými daty, protože ve většině případů probíhá interpolace z vektorových dat. Bez zásahů uživatele (v této práci použití vlastního skriptu, který převáděl vektorová data do požadovaného formátu) není modul *ann.** příliš užitečný pro běžnou interpolaci. Po porovnání s neuronovými sítěmi v programu R Project, se ukázalo že jsou pro běžnou interpolaci vhodnější než sítě z programu GRASS GIS, ačkoliv ani sítě v programu R Project neměly lepší výsledky než metody IDW a kriging.

Použití vícevrstvého perceptronu pro prostorovou interpolaci je zajímavou variantou ke klasickým metodám, ale vyžaduje větší znalosti ze strany uživatele a je také časově náročnější, přičemž výsledky jsou často nejisté a je třeba interpolaci mnohokrát opakovat, než je dosaženo uspokojivých výsledků. Modul *ann.** v současné podobě zatím nelze považovat za rovnocennou náhradu klasických metod, nicméně budoucí vývoj tohoto modulu by mohl toto změnit.

LITERATURA

BHASKARAN, P. a. k. A new approach for deriving temperature and salinity fields in the indian ocean using artificial neural networks. *Journal of Marine Science and Technology*, 15, 2, s. 160–175, 2010. ISSN 1437-8213.

CHOWDHURY, M. a. k. Comparison of ordinary kriging and artificial neural network for spatial mapping of arsenic contamination of groundwater. *Stochastic Environmental Research and Risk Assessment*, 24, 1, s. 1–7, 2010. ISSN 1436-3259.

FRITSCH, S., GUENTHER, F., MARC SULING. *neuralnet: Training of neural networks*, 2012. Dostupné z: `http://CRAN.R-project.org/package=neuralnet`. R package version 1.32. cit. 2013-05-13.

GRASS Development Team. *Geographic Resources Analysis Support System (GRASS GIS) Software*. Open Source Geospatial Foundation, 2012. Dostupné z: `http://grass.osgeo.org`. cit. 2013-05-13.

HENGL, T. *A Practical Guide to Geostatistical Mapping*. Luxembourg : Office for Official Publications of the European Communities, 2009. ISBN 978-92-79-06904-8.

HORÁK, J. *Zpracování dat v GIS*. Ostrava : VŠB-TU Ostrava, 2011.

LONGLEY, P. A., GOODCHILD, M. F., MAGUIRE, D. J., RHIND, D. W. *Geographic Information Systems and Science*. : John Wiley & Sons, 2005. ISBN 0470870001.

NETELER, M., MITASOVA, H. *Open Source GIS: A GRASS GIS Approach*. : Springer, New York, 2008.

NETZEL, P. Implementation of ann in grass - an example of using ann for spatial interpolation. International Conference on Free Software and Open Source in Geoinformatics, 19-20 May 2011, Prague, Czech Republic, 2011.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2012. Dostupné z: `http://www.R-project.org/`. ISBN 3-900051-07-0. cit. 2013-05-13.

RUMELHART, D. E., WIDROW, B., LEHR, M. A. The basic ideas in neural networks. *Communication of the ACM*, 37, 3, s. 87–92, 1994. ISSN 1557-7317.

SNELL, E. S. a. k. Spatial interpolation of surface air temperatures using artificial neural networks: Evaluating their use for downscaling gcms. *Journal of Climate*, 13, 5, s. 886–895, 2000. ISSN 1520-0442.

TARASSENKO, L. *A guide to neural computing applications*, chapter 2. Mathematical background for neural computing. Elsevier Ltd., 1998. ISBN 0340705892.

VENABLES, W. N., RIPLEY, B. D. *Modern Applied Statistics with S*. New York : Springer, fourth edition, 2002. Dostupné z: `http://www.stats.ox.ac.uk/pub/MASS4`. ISBN 0-387-95457-0. cit. 2013-05-13.

VOLNÁ, E. *Neuronové sítě 1*. Ostrava : Ostravská univerzita v Ostravě, 2008.

VOŽENÍLEK, V. a. k. *Metody umělé inteligence v geoinformatice*, chapter Umělá inteligence. Univerzita Palackého v Olomouci, 2011. ISBN 978-80-244-2945-8.

SUMMARY

Spatial interpolation is a common and often used method in geoinformatics. There are many different interpolation methods, each of them used for different purposes. Artificial neural networks are popular computation method used in various scientific areas including geoinformatics.

The aim of this thesis was to test the use of module *ann.** in GRASS GIS for spatial interpolation and to compare it with common interpolation techniques IDW and ordinary kriging. This module was also compared with neural networks packages *nnet* and *neuralnet* in R Project. The evaluation of methods was based mainly on RMSE, although time demands and user experience was also shown.

Multilayer perceptron with backpropagation training algorithm was used to perform the interpolation both in GRASS GIS and R Project. All the tests were done on artificial data created in R Project which simulated three surfaces with different characteristics. The final interpolation was made once for each of the surfaces. These data were split into two parts: training data (70 %) and testing data (30 %).

In order to find the best configuration for the multilayer perceptron many different settings were tested. Number of neurons in hidden layers was the main tested parameter. Then the interpolation was done. In GRASS GIS two slightly different approaches were used for interpolation using multilayer perceptron. The first model of neural network was trained on raster data file. This was the original use of *ann.** module. Then a vector data file was prepared with the use of new script in Java (not a part of the *ann.** module) and a new neural network model was trained on this file. After the multilayer perceptron models were trained interpolation was made in the same way with both networks.

The results indicate that multilayer perceptron in the *ann.** module can be used for spatial interpolation purposes. However the resulting RMSE was higher then RMSE from IDW and ordinary kriging methods. Also the time demands were higher when using the neural networks *ann.** module. When compared with neural network packages in R Project it is better to use the packages in R Project. Training of multilayer perceptron was faster in this case and results were the same or slightly better. However the resulting RMSE was still higher than the RMSE of the other methods (IDW, ordinary kriging). Also the use of neural networks is difficult for inexperienced users.

All the test were done on artificial data. If the real data were used, the results would be probably different. Also adding more input parametres could have improved the performance of the multilayer perceptron model.

According to the author of the *ann.** module a new modules for vector data will be prepared (Netzel, 2011). The date of completion is unknown but it is possible that this new modules will be better for the purpose of spatial interpolation than the recent one. The *ann.** module tested in this thesis cannot be considered as an equal replacement of the classical interpolation techniques, but it has a potential.

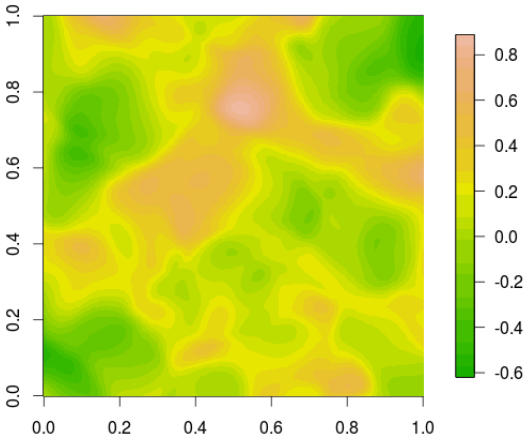
PŘÍLOHY

SEZNAM PŘÍLOH

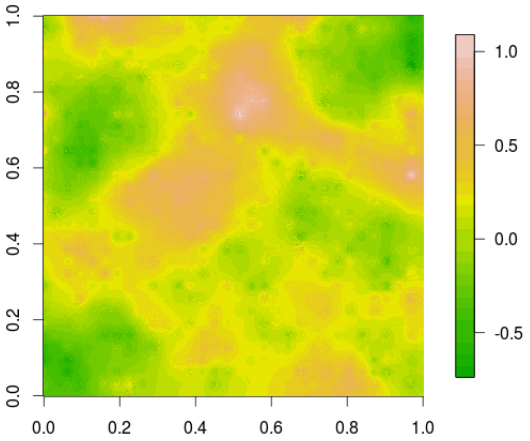
Vázané přílohy

Volné přílohy

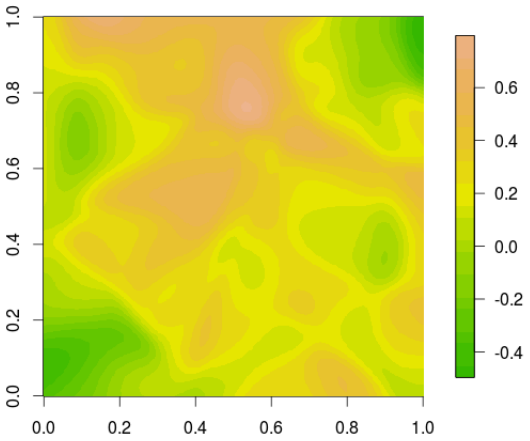
Příloha 1



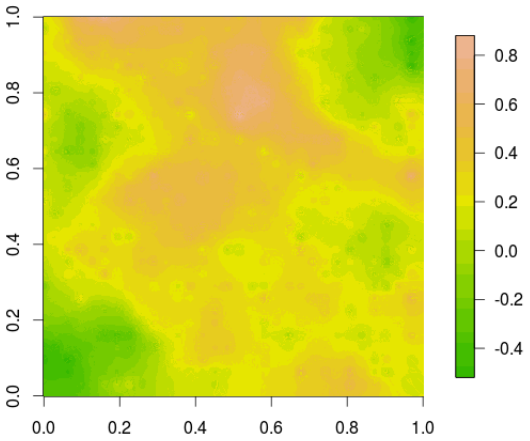
n. síť - členitost 1



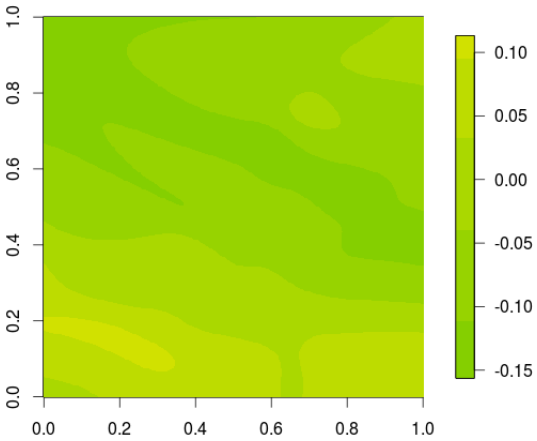
IDW - členitost 1



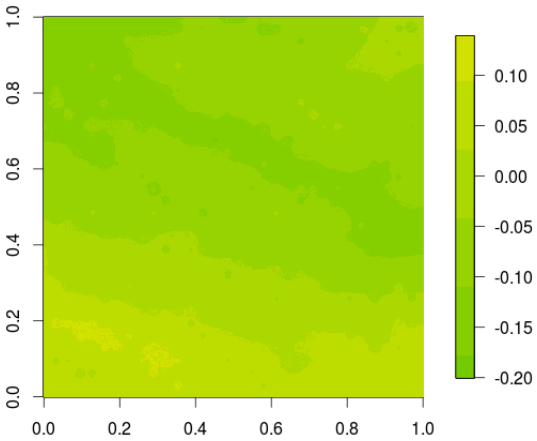
n. síť - členitost 2



IDW - členitost 2

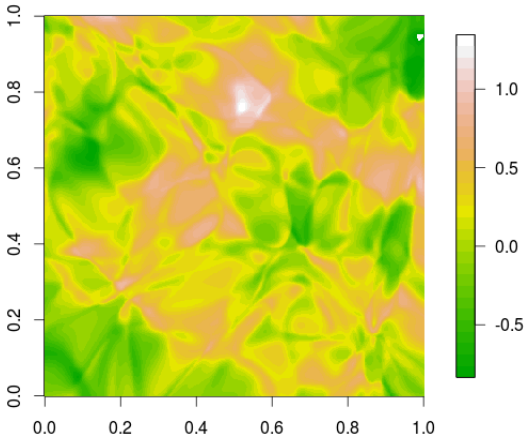


n. síť - členitost 3

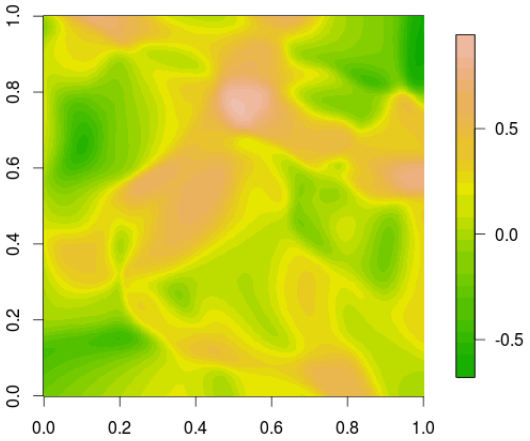


IDW - členitost 3

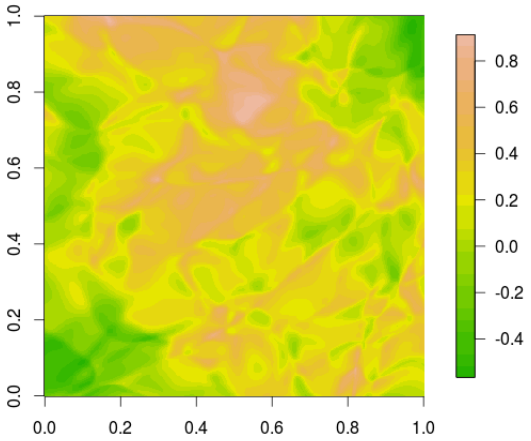
Obrázek 1: Výsledné rastry z neuronové sítě (rastr) a IDW - GRASS GIS



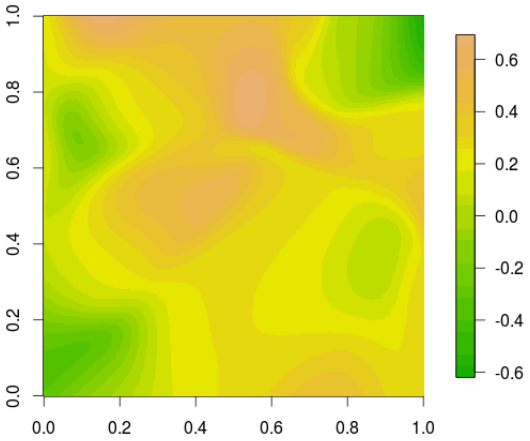
n. síť (32, 38, 27) - členitost 1



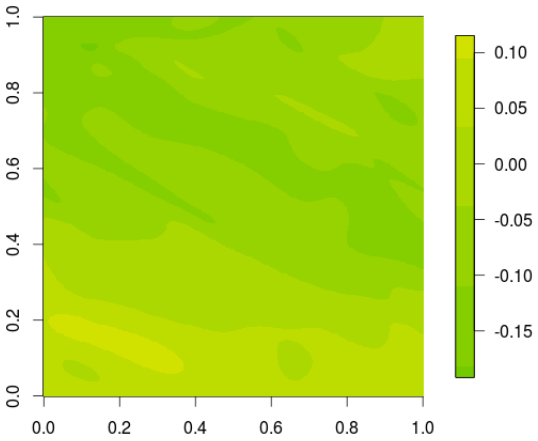
n. síť (20, 25, 17) - členitost 1



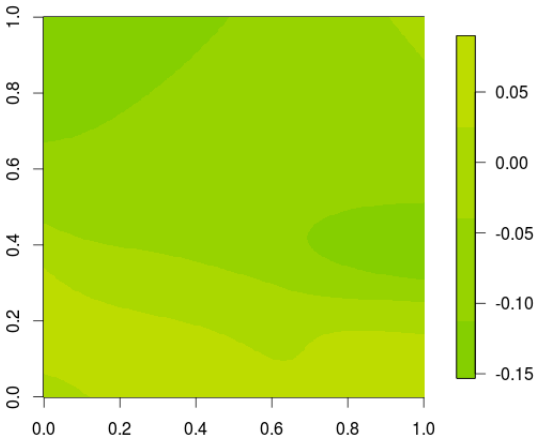
n. síť (32, 38, 27) - členitost 2



n. síť (20, 25, 17) - členitost 2

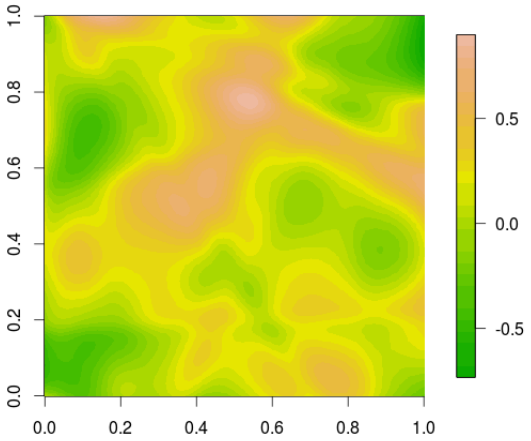


n. síť (32, 38, 27) - členitost 3

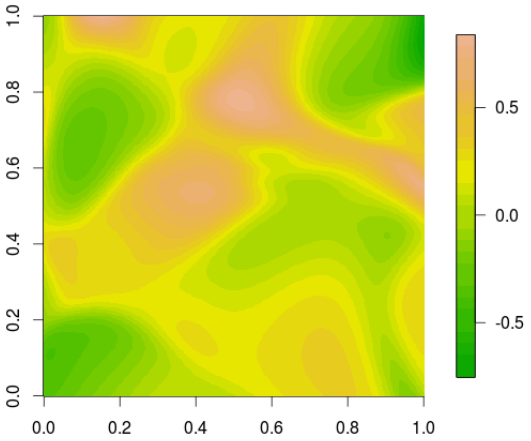


n. síť (20, 25, 17) - členitost 3

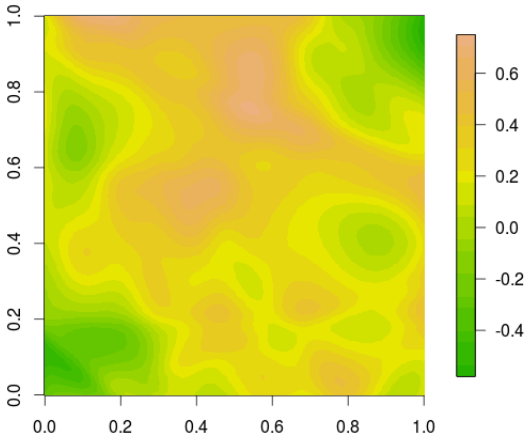
Obrázek 2: Výsledné rastry z neuronových sítí (vektor) - GRASS GIS



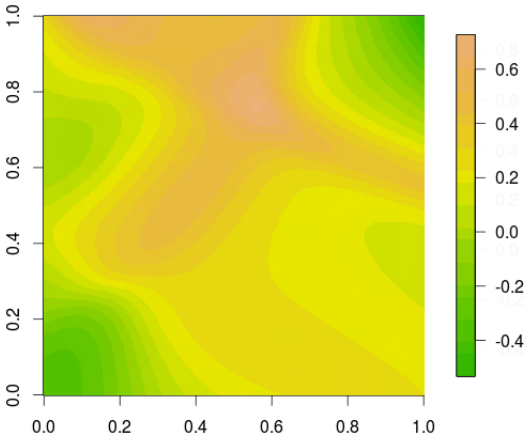
nnet - členitost 1



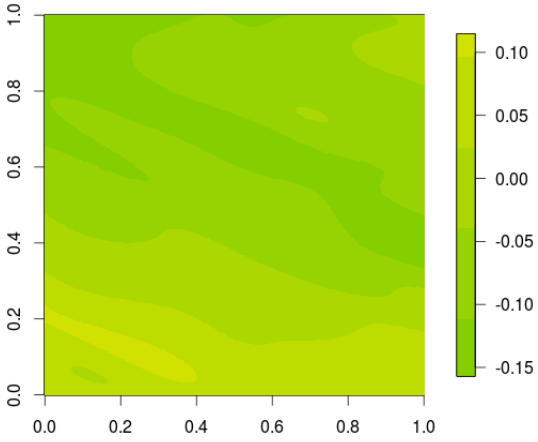
neuralnet - členitost 1



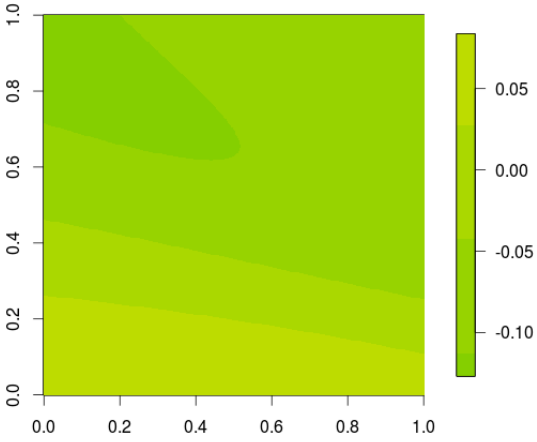
nnet - členitost 2



neuralnet - členitost 2

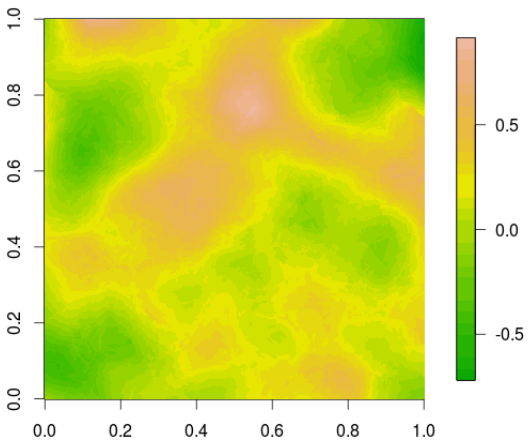


nnet - členitost 3

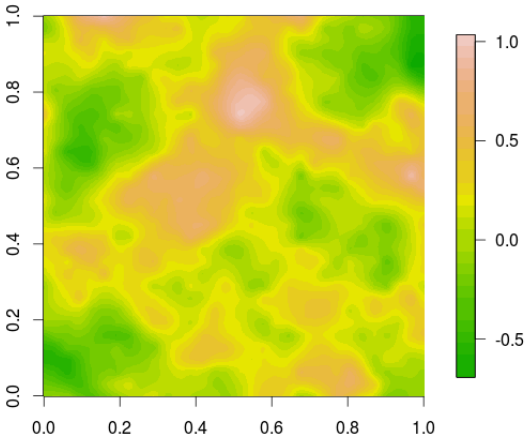


neuralnet - členitost 3

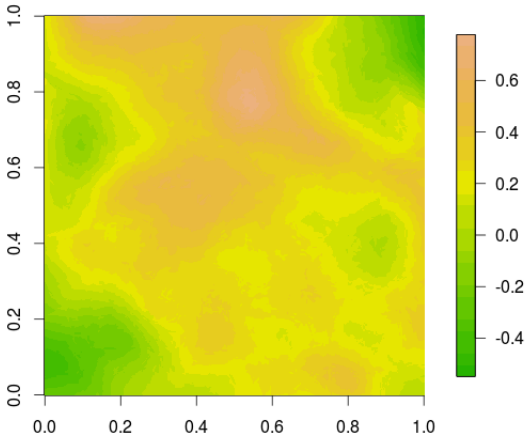
Obrázek 3: Výsledné rastry z neuronových sítí - R Project



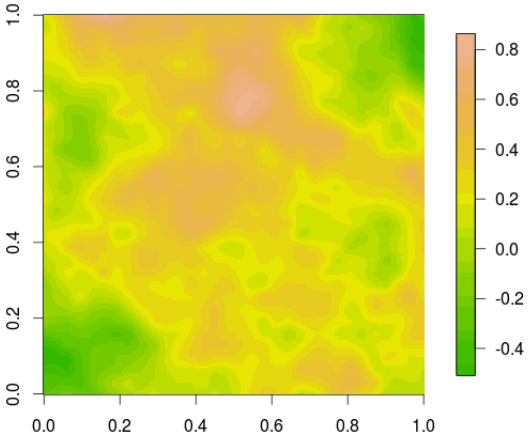
IDW - členitost 1



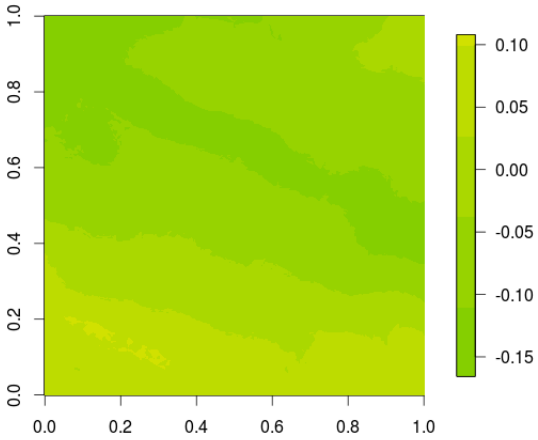
kriging - členitost 1



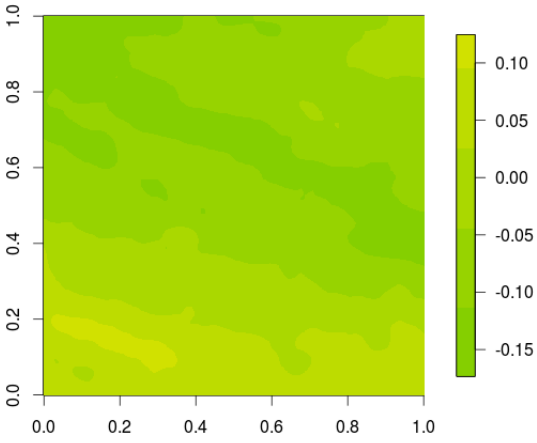
IDW - členitost 2



kriging - členitost 2



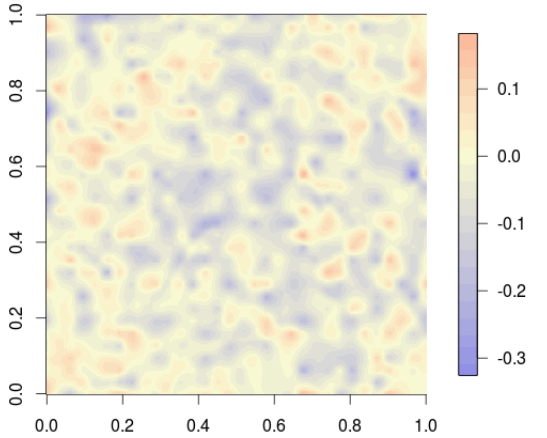
IDW - členitost 3



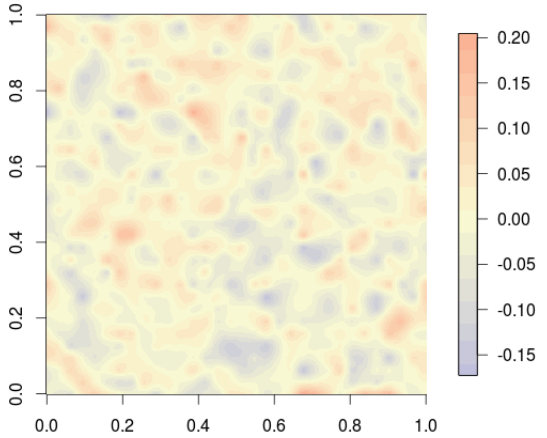
kriging - členitost 3

Obrázek 4: Výsledné rastry z IDW a krigingu - R Project

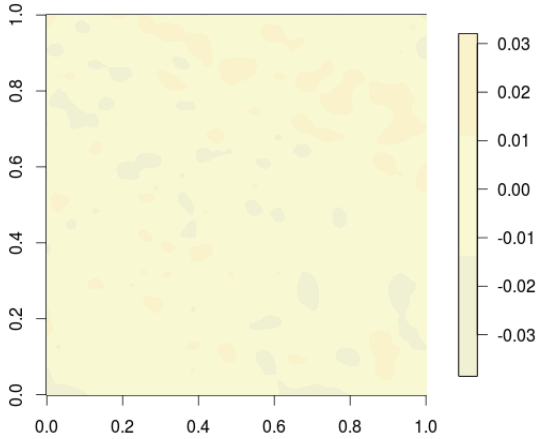
Příloha 2



členitost 1

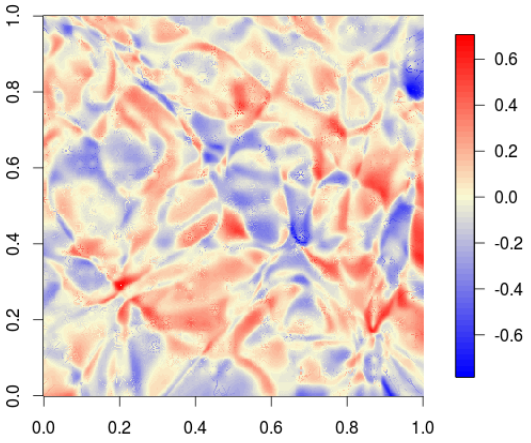


členitost 2

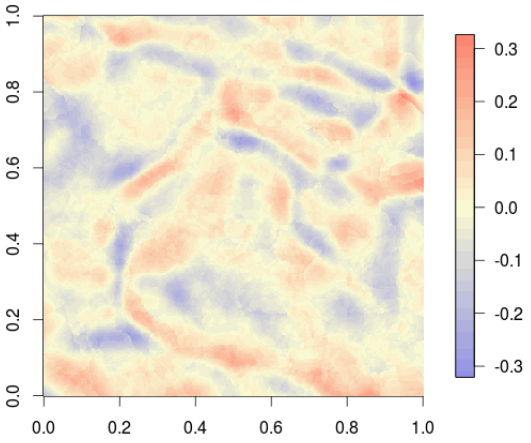


členitost 3

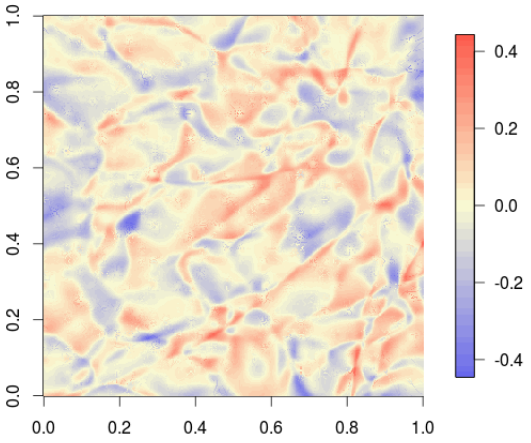
Obrázek 5: Rozdíl mezi novým rastrem z n. sítě a původním rastrem - GRASS GIS



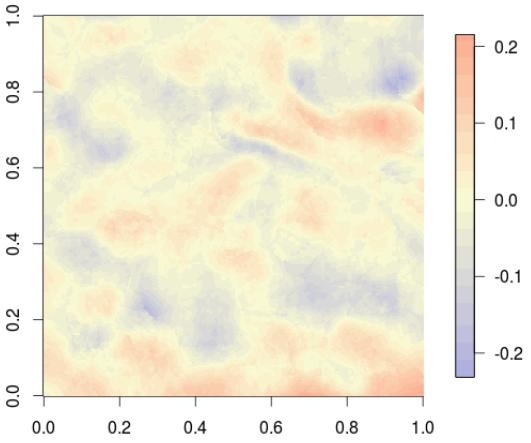
n. síť (32, 38, 27) - členitost 1



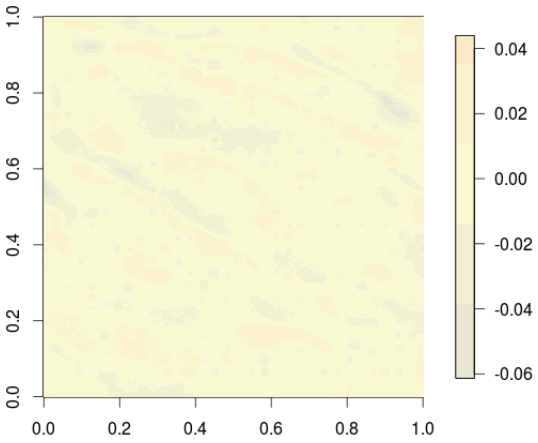
n. síť (20, 25, 17) - členitost 1



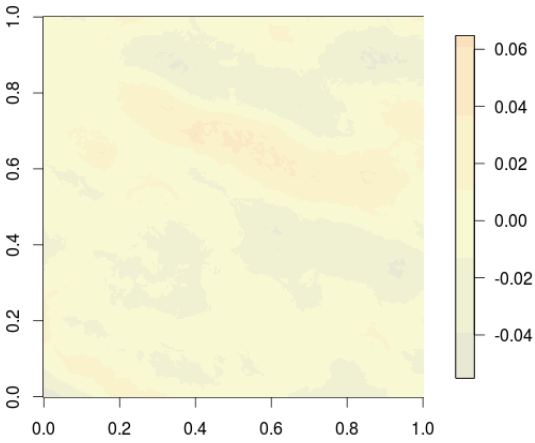
n. síť (32, 38, 27) - členitost 2



n. síť (20, 25, 17) - členitost 2

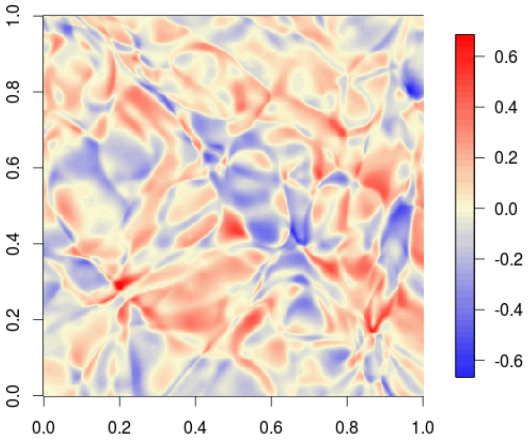


n. síť (32, 38, 27) - členitost 3

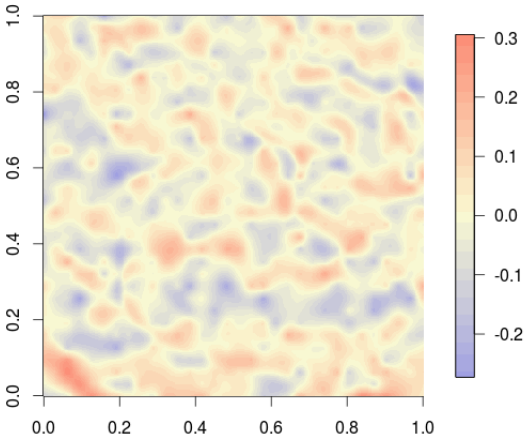


n. síť (20, 25, 17) - členitost 3

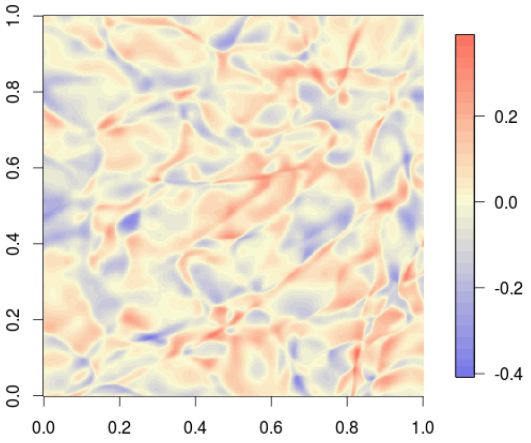
Obrázek 6: Rozdíl mezi rastry z n. sítí (vektor) a z IDW - GRASS GIS



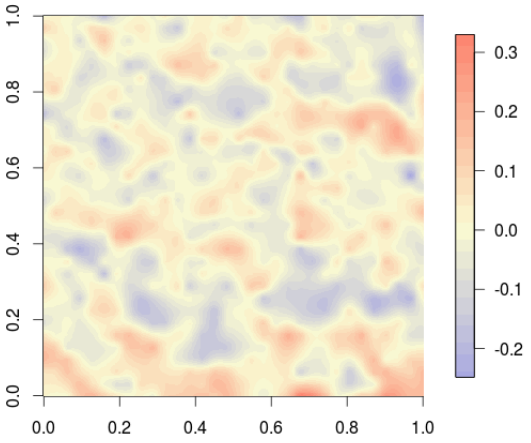
n. síť (32, 38, 27) - členitost 1



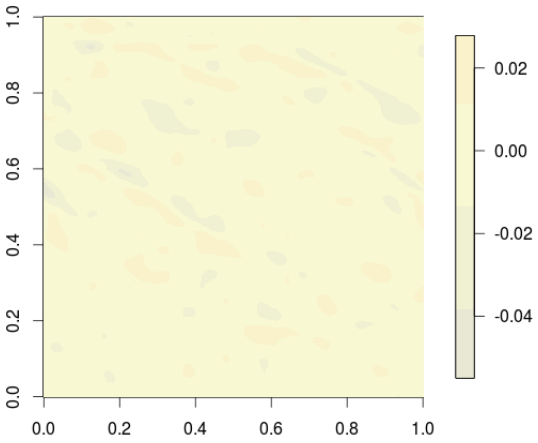
n. síť (20, 25, 17) - členitost 1



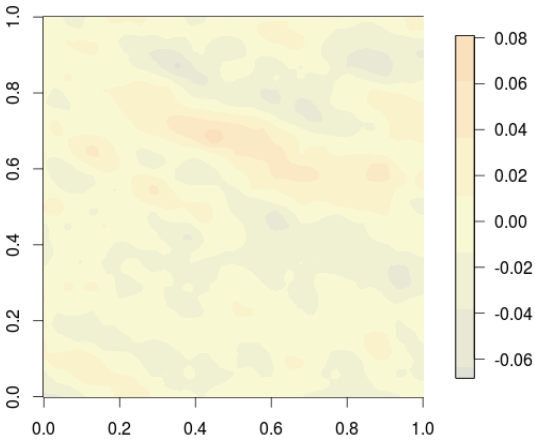
n. síť (32, 38, 27) - členitost 2



n. síť (20, 25, 17) - členitost 2

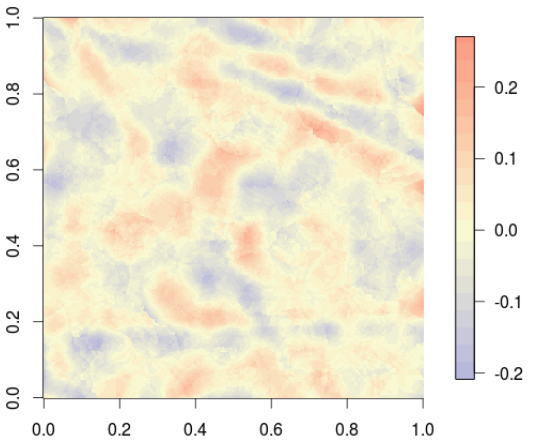


n. síť (32, 38, 27) - členitost 3

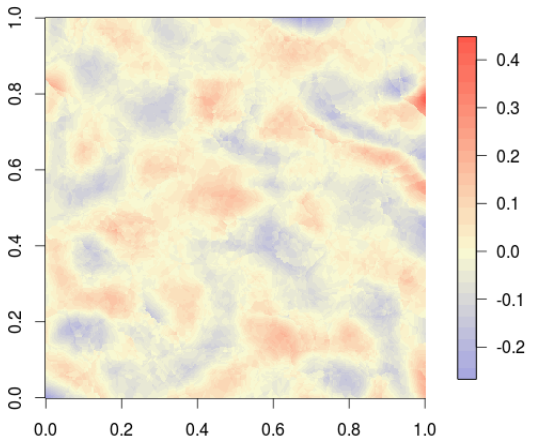


n. síť (20, 25, 17) - členitost 3

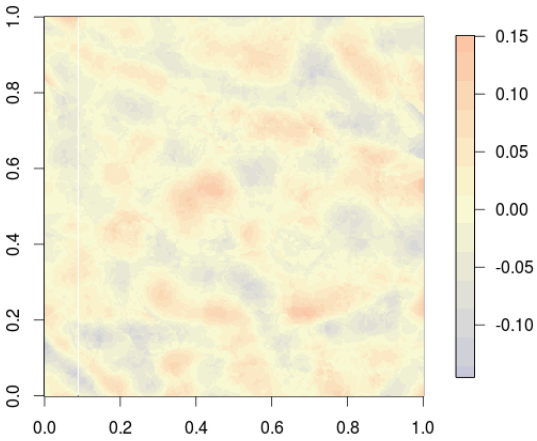
Obrázek 7: Rozdíl mezi rastry z n. sítí (vektor) a z krigingu - GRASS GIS



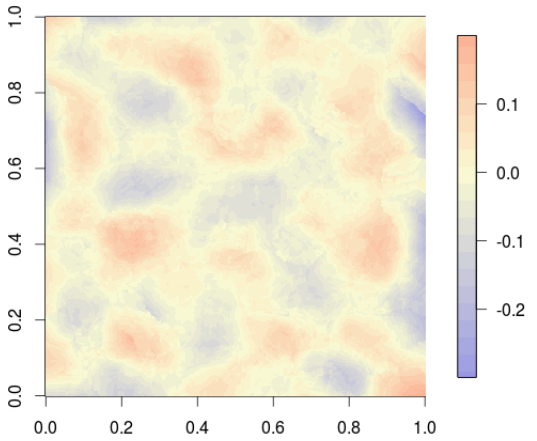
nnet - členitost 1



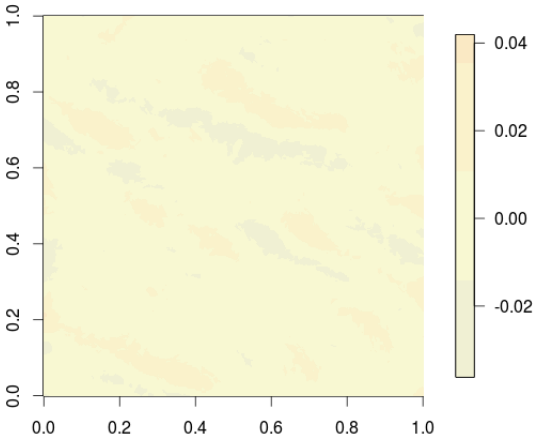
neuralnet - členitost 1



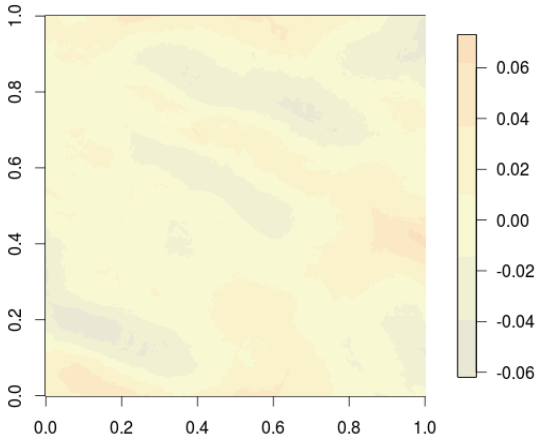
nnet - členitost 2



neuralnet - členitost 2

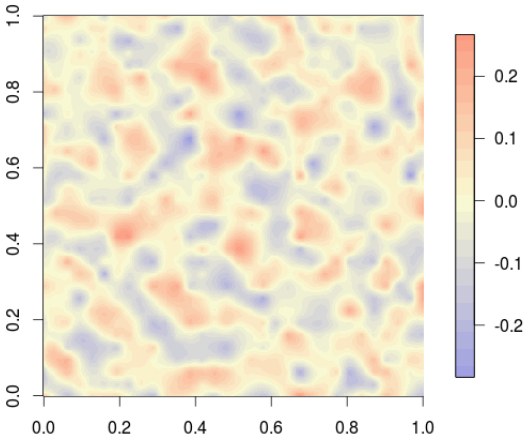


nnet - členitost 3

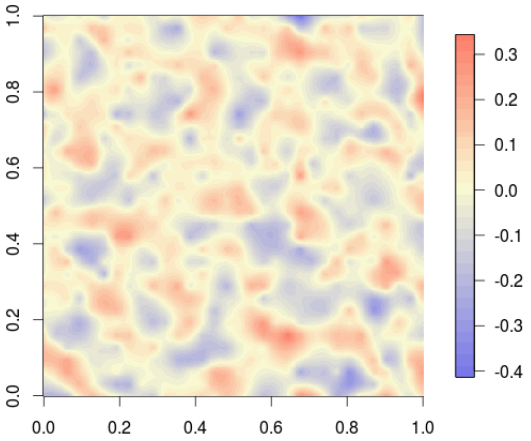


neuralnet - členitost 3

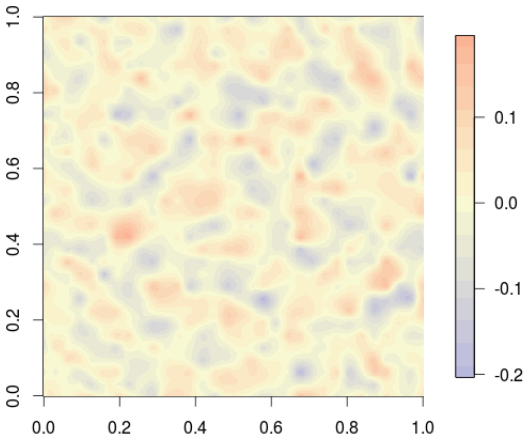
Obrázek 8: Rozdíl mezi rastry z n. sítí a z IDW - R Project



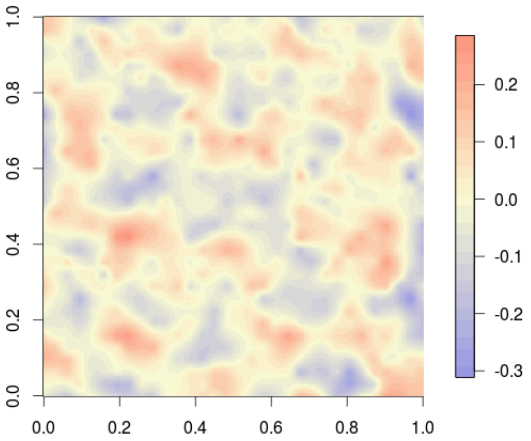
nnet - členitost 1



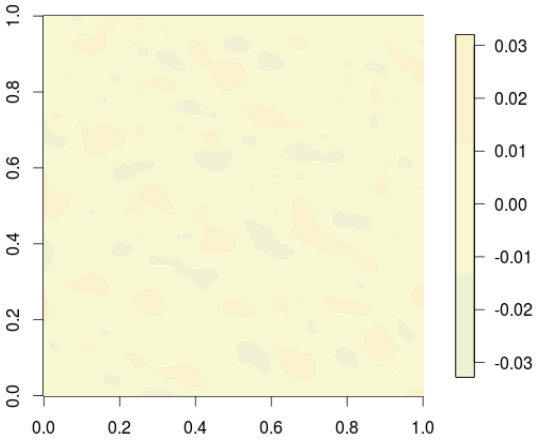
neuralnet - členitost 1



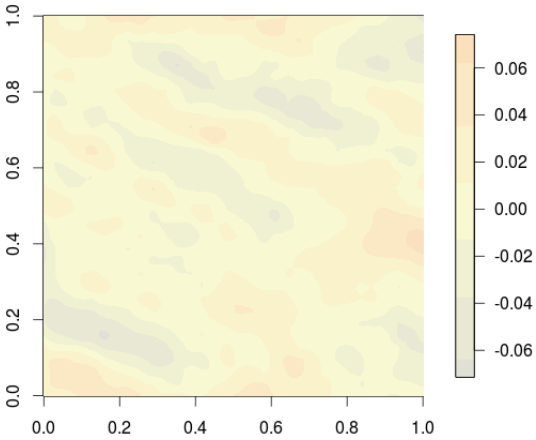
nnet - členitost 2



neuralnet - členitost 2



nnet - členitost 3



neuralnet - členitost 3

Obrázek 9: Rozdíl mezi rastry z n. sítí a z krigingu - R Project

Příloha 3

Tabulka 1: Statistiky pro obrázek 5

	min.	1. qu	median	průměr	3. qu	max.
členitost 1	-0.32560	-0.06590	-0.03058	-0.03145	0.00332	0.18220
členitost 2	-0.17260	-0.02727	-0.00064	-0.00112	0.02539	0.20460
členitost 3	-0.03854	-0.006555	-0.00121	-0.00078	0.00508	0.03207

Tabulka 2: Statistiky pro obrázek 6 - n. síť (32, 38, 27)

	min.	1. qu	median	průměr	3. qu	max.
členitost 1	-0.78410	-0.10020	0.00655	0.00656	0.11480	0.70710
členitost 2	-0.44550	-0.06058	0.00465	0.00235	0.06556	0.44350
členitost 3	-0.06131	-0.00709	-0.00064	-0.00089	0.00629	0.04398

Tabulka 3: Statistiky pro obrázek 6 - n. síť (20, 25, 17)

	min.	1. qu	median	průměr	3. qu	max.
členitost 1	-0.32070	-0.04990	0.00029	-0.00303	0.04652	0.32650
členitost 2	-0.23140	-0.03783	-0.0041	-0.00339	0.02838	0.21540
členitost 3	-0.05512	-0.01447	-0.00521	-0.00455	0.00369	0.06478

Tabulka 4: Statistiky pro obrázek 7 - n. síť (32, 38, 27)

	min.	1. qu	median	průměr	3. qu	max.
členitost 1	-0.66640	-0.09208	0.00735	0.00532	0.10440	0.68610
členitost 2	-0.40810	-0.05614	0.00210	0.00189	0.05916	0.38960
členitost 3	-0.05497	-0.00505	-0.00029	-0.00036	0.00469	0.02784

Tabulka 5: Statistiky pro obrázek 7 - n. síť (20, 25, 17)

	min.	1. qu	median	průměr	3. qu	max.
členitost 1	-0.27280	-0.04891	-0.00101	-0.00198	0.04534	0.30560
členitost 2	-0.24830	-0.04579	-0.00577	-0.00311	0.03778	0.32990
členitost 3	-0.06829	-0.01668	-0.00624	-0.00456	0.00571	0.08096

Tabulka 6: Statistiky pro obrázek 8 - nnet

	min.	1. qu	median	průměr	3. qu	max.
členitost 1	-0.208600	-0.043820	-0.003221	-0.003122	0.035660	0.270700
členitost 2	-0.145400	-0.025550	-0.000979	0.000491	0.025620	0.150700
členitost 3	-0.036220	-0.005405	0.000133	0.000207	0.005884	0.041920

Tabulka 7: Statistiky pro obrázek 8 - neuralnet

	min.	1. qu	median	průměr	3. qu	max.
členitost 1	-0.267000	-0.047780	-0.005300	-0.004578	0.037550	0.448600
členitost 2	-0.299200	-0.032380	0.001350	0.001411	0.037670	0.199900
členitost 3	-0.061930	-0.010860	0.001439	0.000443	0.012260	0.073090

Tabulka 8: Statistiky pro obrázek 9 - nnet

	min.	1. qu	median	průměr	3. qu	max.
členitost 1	-0.283500	-0.055410	-0.002914	-0.002065	0.047490	0.266900
členitost 2	-0.203500	-0.031310	0.001976	0.000774	0.032910	0.195400
členitost 3	-0.032770	-0.006253	0.000183	0.000196	0.006422	0.032050

Tabulka 9: Statistiky pro obrázek 9 - neuralnet

	min.	1. qu	median	průměr	3. qu	max.
členitost 1	-0.413700	-0.059920	-0.000036	-0.003520	0.050220	0.343800
členitost 2	-0.311300	-0.050230	-0.000131	0.001694	0.053350	0.285800
členitost 3	-0.071410	-0.013020	0.001346	0.000432	0.015230	0.074130