České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra kybernetiky

BAKALÁŘSKÁ PRÁCE

Automatická detekce akčních potenciálů neuronů z mikroelektrodových signálů

2015 **Jakub Zamouřil**

Prohlášení autora práce

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne ....

Podpis autora

Anotace a klíčová slova

Před analýzou mikroEEG záznamů zachycených mikroelektrodami při hloubkové mozkové stimulaci je nutné oddělit signál od šumu, tedy detekovat v záznamu akční potenciály neuronů, které jsou dále tříděny a zpracovávány. Spolehlivě v záznamu detekovat akční potenciály je obtížné, protože šum je podobný signálu ve frekvenční i časové oblasti záznamu. V současnosti již pro detekci i třídění akčních potenciálů existuje řada algoritmů, které pro řešení této problematiky využívají celou škálu metod, avšak většina z nich není veřejně dostupná. Cílem této práce je tedy vytvořit volně šiřitelný automatický algoritmus detekce akčních potenciálů, který nahradí nevyhovující detekční algoritmus v současnosti používaný na katedře kybernetiky. S využitím tří metod detekce spiků byl vytvořen algoritmus s automaticky nastavovaným prahem, který v přesnosti detekce překonal dosud používaný algoritmus s $p < 0.01$.

Klíčová slova: hloubková mozková stimulace, mikroEEG, automatická detekce, akční potenciály, spiky, Wave_clus, Osort

Abstract and keywords

For the analysis of microEEG recordings captured with microelectrodes during deep brain stimulation it is necessary to separate the signal from the noise by detection of neural action potentials, which are then further sorted and processed. It is difficult to reliably detect action potentials in microEEG recordings because the noise is similar to the signal in time and frequency domain. Currently many spike detection and sorting algorithms exist, which tackle these problems by variety of methods, however they are usually inaccessible to the public. The aim of this work is thus to create a free and automatic spike detection algorithm to replace the inadequate detection algorithm currently used by the Department of Cybernetics. Using three different spike detection methods an algorithm with an automatically adjusted threshold was created, which surpassed currently used algorithm in detection accuracy with $p < 0.01$.

Keywords: Deep Brain Stimulation, microEEG, automatic detection, action potentials, spikes, Wave_clus, Osort

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra kybernetiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: Jakub Z a m o u ř i l

Studijní program: Kybernetika a robotika (bakalářský)

Obor: Robotika

Název tématu: Automatická detekce akčních potenciálů neuronů z mikroelektrodových signálů

Pokyny pro vypracování:

1. Prostudujte problematiku hloubkové stimulace mozku u pacientů s Parkinsonovou nemocí.
2. Seznamte se s toolboxem Wave_clus pro detekci a třídění akčních potenciálů z jednokanálových signálů.
3. Vyhledejte v odborné literatuře různé metody detekce akčních potenciálů a alespoň dvě implementujte.
4. Navrhněte vlastní metodu pro automatickou detekci akčních potenciálů.
5. Porovnejte implementované metody na umělých datech, vygenerovaných podle nahradných mikroelektrodových záznamů a diskutujte výhody či nevýhody jednotlivých metod.
6. Nejvhodnější metodu zakomponujte do toolboxu Wave_clus.

Seznam odborné literatury:

[1] Wilson, Scott et al.: Spike detection: a review and comparison of algorithms. Clinical Neurophysiology, Volume 113, Issue 12, 2002.
[2] Quiroga, Quian R. et al.: Unsupervised Spike Detection and Sorting with Wavelets and Superparamagnetic Clustering. Neural Comp 16:1661/1687, 2004.
[3] Lewicki, M. S.: A review of methods for spike sorting: the detection and classification of neural action potentials. Network 9 (4), England 1998.

Vedoucí bakalářské práce: Ing. Jiří Wild

Platnost zadání: do konce letního semestru 2015/2016

L.S.

doc. Dr. Ing. Jan Kybic
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 23. 1. 2015

Poděkování

Rád bych zde poděkoval svému vedoucímu práce, Ing. Jiřímu Wildovi, za jeho rady a připomínky, pravidelné schůzky a obecně jeho angažovanost, díky které jsem dokázal tuto práci dokončit (snad) správně, včas, a bez zbytečného stresu.

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

Lidé v těžkém stadiu Parkinsonovy nemoci, kterým již nepomáhá tradiční léčba za pomoci léčiv jako levodopa, bývají kvůli symptomům nemoci jako je třesavka či svalová ztuhlost odkázání na své blízké či personál nemocnice. Jejich nadějí na návrat do běžného života je neurochirurgická léčba, a to zejména tzv. hloubková mozková stimulace (DBS), díky které je většina příznaků Parkinsonovy nemoci výrazně utlumena.

Hloubková mozková stimulace je procedura, při které je do mozku pacienta zavedena mikroelektroda napojená na implantovaný generátor impulzů. Elektrické impulzy z této mikroelektrody utlumují funkci mozkových center, které byly v důsledku Parkinsonovy nemoci nadměrně aktivní, čímž narušovaly motoriku pacienta.

Při zavádění mikroelektrody do mozku je vyžadována vysoká přesnost jejího umístění, aby elektroda vysílala impulzy do správných jader mozku. Přesnosti je dosaženo dvěma způsoby - pomocí předoperačního osnímkování mozku magnetickou rezonancí a za pomoci analýzy signálu mozkové aktivity (mikroEEG) zachyceného mikroelektrodou při samotné operaci.

Při analýze signálu je využíváno faktu, že různé skupiny neuronů komunikují pomocí akčních potenciálů (spiků) s různými tvary a různou frekvencí. Ze zachyceného signálu jsou tedy nejprve vyjmuty detekované akční potenciály (*spike detection*) a ty jsou následně roztříděny a přiřazeny jednotlivým skupinám neuronů (*spike sorting*). Díky tomu je následně možné určit oblast pacientova mozku, ve které se hrot elektrody nachází.

Tato práce se zabývá algoritmy detekce akčních potenciálů (*spike detection*) a jejich implementací. V současnosti již existuje široké spektrum různých algoritmů detekce akčních potenciálů, které využívají různých metod filtrování a úprav signálu. Algoritmus detekce spiků je obsažen i v balíku funkcí *Wave_clus* pro prostředí MATLAB, který je v současnosti používán na katedře kybernetiky k detekci a třídění akčních potenciálů. Používaný algoritmus má však při detekci neuspokojivou přesnost, špatně funguje pro

signály, kde se mohou vyskytovat spiky i s opačnou polaritou, a je nutné jej na každý signál manuálně nastavovat, aby došlo k vylepšení kvality detekce. Proto bylo žádoucí algoritmus detekce v balíku *Wave_clus* vylepšit tak, aby se nastavoval automaticky (nebylo potřeba zásahu uživatele) a detekoval akční potenciály co nejlépe na různých typech signálu s různými úrovněmi šumu.

Cílem této práce je tedy:

1. prozkoumat různé používané metody detekce spiků,
2. pomocí jedné nebo více vybraných metod navrhnout a implementovat vlastní algoritmus automatické detekce spiků,
3. porovnat jeho úspěšnost s dalšími již existujícími algoritmy detekce,
4. v případě úspěchu tento algoritmus zakomponovat do balíku *Wave_clus*.

Kapitola 2

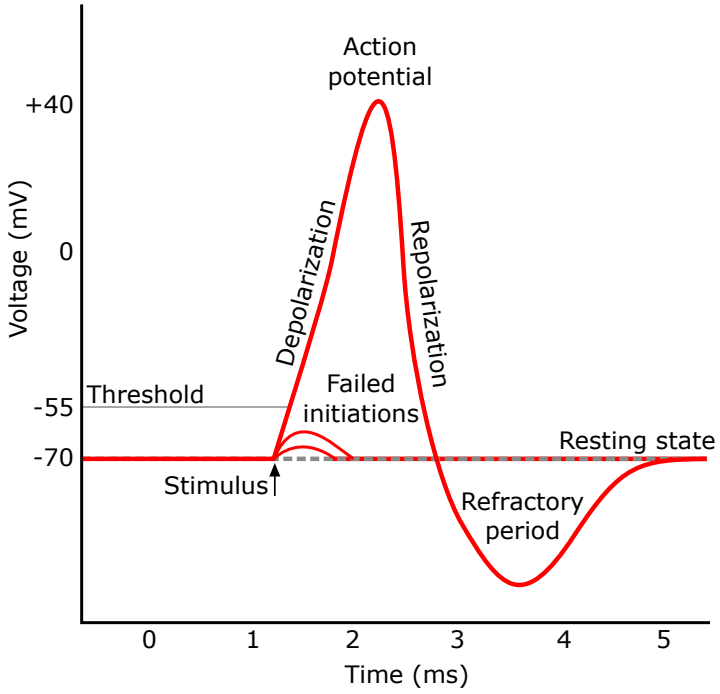
Popis problematiky v oboru medicíny

2.1 Přenos informace mezi neurony

„Neuron (nervová buňka) je vysoce specializovaný typ buňky, schopný přijímat, zpracovávat, vysílat a přenášet informace.“ [1, str. 380] Neurony se zpravidla skládají z mnoha dendritů (vstupních výběžků), axonu (výstupního výběžku) a samotného těla neuronu. Neurony se mezi sebou propojují synapsemi, které nejčastěji spojují axon jednoho neuronu s jedním z dendritů jiného neuronu.

Samotná komunikace mezi neurony pak probíhá pomocí akčních potenciálů (spiků), což jsou elektrochemické impulsy o napětí 30 - 40 mV vznikající v těle neuronu a šířící se axonem přes synapse do dendritů ostatních neuronů. Pokud je neuron v klidu, je na jeho výstupu tzv. klidový potenciál (typicky -70 mV). Protože neuron může mít na výstupu pouze jeden z těchto dvou stavů (spike nebo klidový stav), dá se říci, že informace jsou v mozku kódovány binárně.

Neurony pak mají v sobě zakódovány různé váhové funkce, jimiž sčítají své jednotlivé vstupy. Pokud tento váhovaný součet vstupních potenciálů překročí kritickou hodnotu (prahový potenciál), vyšle neuron na svůj výstup akční potenciál. Neurony však nedokáží akční potenciál na svém výstupu udržovat trvale, vysílají jej ve formě krátkých impulsů trvajících několik milisekund, po kterých vždy následuje refrakterní perioda, po jejíž trvání není možné vysílat další spiky. Pokud po skončení refrakterní periody (také trvající několik ms) je na vstupu neuronu stále nadprahový podnět, je na výstup vyslán nový akční potenciál. Typický průběh akčního potenciálu je uveden na obrázku 2.1. Podrobnější informace o způsobu přenosu informace mezi neurony je možné nalézt v knize [1, kapitola 10].



Obrázek 2.1: Typický průběh akčního potenciálu. [16]

2.2 Parkinsonova nemoc

2.2.1 Popis Parkinsonovy nemoci

Parkinsonova nemoc je chronické, pomalu se rozvíjející onemocnění centrální nervové soustavy postihující hlavně starší pacienty, jehož hlavním příznakem jsou pohybové potíže jako třesavka, ztuhlost, zpomalenost či nestabilita. Pacienti v pokročilém stadiu Parkinsonovy nemoci často mají problém s chůzí, komunikací a úkoly vyžadujícími jemnou motoriku. Onemocnění začíná zvolna a s časem se jeho příznaky postupně zhoršují, u některých lidí rychleji než u jiných. Hlavním znakem této nemoci je nedostatek dopaminu v těle pacienta.[2]

Dopamin je neurotransmiter, tedy jedna z látek zajišťujících přenos akčních potenciálů mezi buňkami nervové soustavy. V lidském těle zajišťuje mnoho různých funkcí, z nichž nejznámější je zřejmě jeho působnost ve středním mozku, kde se podílí na tvorbě emocí. Z pohledu naší problematiky je ale důležitější, že hladina dopaminu v mozku také určuje míru aktivity centra *globus pallidus*, části bazálních ganglií. Toto centrum pak různými cestami stimuluje i inhibuje thalamus, což je jedno z mozkových center ovládajících motoriku.[3]

Ve zdravém těle je dopamin vytvářen v části středního mozku zvané *substantia nigra* neboli černá substance. Parkinsonova nemoc vzniká, když buňky tvořící dopamin začnou

odumírat. Kvůli nedostatku dopaminu v centru *globus pallidus* pak dojde k nadměrné inhibici thalamu, což vede k narušení motoriky pacienta. Příčina odumírání těchto buněk je dosud neznámá. [4]

Správně diagnostikovat Parkinsonovu nemoc je obtížné, protože na ni zatím neexistuje žádný laboratorní test. Nadměrný úbytek buněk produkujících dopamin je však možné nepřímo vypozorovat pomocí pozitronové emisní tomografie (PET).[2]

2.2.2 Léčba Parkinsonovy nemoci

V dnešní době zatím neexistuje účinná metoda, která by dokázala Parkinsonovu nemoc vyléčit, nicméně existuje několik způsobů jak potlačovat či alespoň zmírňovat její příznaky. V současnosti nejrozšířenějším způsobem je podávání látky zvané *levodopa*, což je přirozeně se vyskytující látka v lidském těle, která se v mozku enzymaticky přeměňuje na dopamin. Podávat pacientům přímo dopamin není žádoucí, protože samotný dopamin špatně proniká z krevního řečiště do mozku a způsobuje nežádoucí vedlejší účinky. Je však možné pacientům podávat tzv. agonisty dopaminu, což jsou látky, které do mozku procházejí snadno a působí v něm na receptory dopaminu (tedy dopamin do určité míry nahrazují). K potlačování příznaků se používá i mnoho dalších léčiv, které různými způsoby pomáhají ke zlepšení stavu pacienta, hlavními léčebnými látkami však zůstává levodopa a agonisté dopaminu.

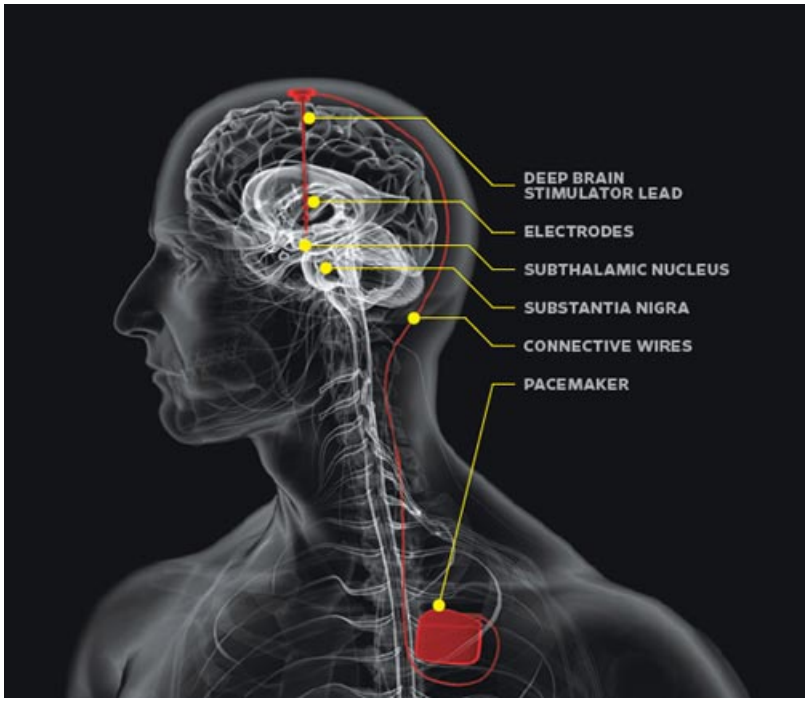
Při vážných případech onemocnění, kdy už je podávání látek méně účinné či zcela neúčinné, je možné využít neurochirurgickou léčbu. Dříve se používala procedura zvaná *stereotaktická léze*, která spočívá v cíleném poškození vybraných skupin jader v bazálních gangliích, čímž se potlačí jejich nadměrná aktivita vyvolaná nedostatkem dopaminu. Tato procedura u některých pacientů vedla ke zlepšení příznaků, avšak byla zatížena rizikem nevratného poškození mozku. Proto se dnes už přistupuje spíše k hluboké mozkové stimulaci *(Deep Brain Stimulation)*.[2, kapitola 9]

2.3 Hloubková mozková stimulace (DBS)

2.3.1 Popis DBS

Hloubková (někdy také nazývána hluboká) mozková stimulace (DBS) je šetrnější neurochirurgickou léčbou Parkinsonovy nemoci, při níž nedochází k cílenému poškození mozkové tkáně. Tato technika spočívá v trvalém zavedení tenké elektrody do určité části mozku pacienta a její napojení podkožními kabely na generátor impulzů uložený v hrudi pacienta (podobně jako kardiostimulátor).

Elektroda se v současnosti zavádí nejčastěji do subthalamického jádra (STN) nebo méně často do centra *globus pallidus*, kam přivádí elektrický signál o frekvenci 130 Hz (nebo více) z generátoru impulzů. Tato stimulace příslušných motorických center pomáhá k výraznému zmírnění některých příznaků nemoci, a to zejména třesavky a ztuhlosti, což umožňuje pacientům návrat do běžného života. Protože však tato metoda nijak neadresuje příčinu Parkinsonovy nemoci, tedy odumírání buněk tvořících dopamin, nemoc nadále trvá a rozvoj jejích příznaků může pokračovat. Mezi příznaky, které se při DBS nezlepšují, patří například poruchy rovnováhy a plynulosti řeči.[2, kapitola 9.4]



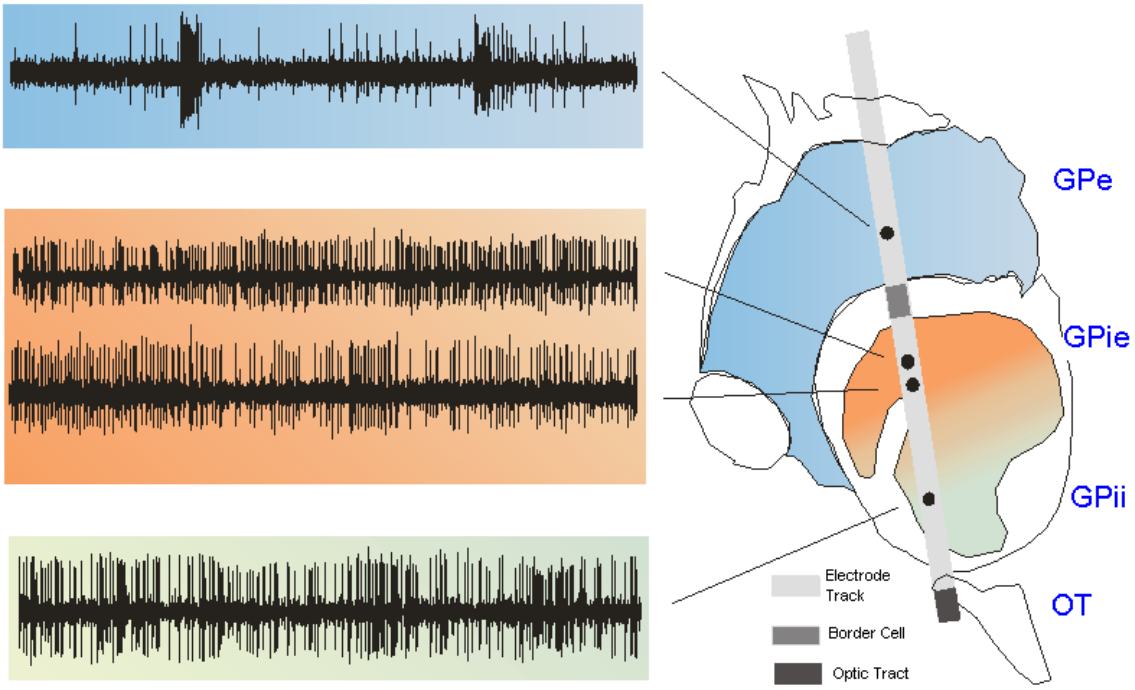
Obrázek 2.2: Schéma hluboké mozkové stimulace. [17]

2.3.2 Zavádění elektrody při DBS

Zavádění mikroelektrody do mozku pacienta je poměrně obtížné, protože chirurg při operaci nevidí, kde přesně v mozku se nachází špička elektrody a musí se spoléhat na signály, které z elektrody přicházejí, a na zpětnou vazbu od pacienta (při tomto druhu operací obvykle pacient není uspáván, aby bylo možné ihned hodnotit jeho reakce na zavádění elektrody a stimulaci).

Před operací obvykle dochází k připevnění stereotaktického rámu k hlavě pacienta (ten poskytuje uvnitř hlavy nepohyblivou třírozměrnou souřadnicovou soustavu) a k oskenování mozku pacienta pomocí magnetické rezonance (MRI). Z MRI dat pak chirurg

dokáže určit přesnou polohu požadovaného mozkového centra v souřadnicovém systému rámu. Toto určení polohy je většinou dostatečné pro zavedení elektrody do správného místa, avšak mezi MRI pacienta a samotnou operací většinou uplyne určitá doba, ve které může dojít k malému vzájemnému posunu jednotlivých částí mozku pacienta. Také je třeba vzít v úvahu možné nepřesnosti a anomálie při MRI. Proto se většinou při operaci jako podpůrná metoda pro zvýšení přesnosti zaznamenává a analyzuje signál ze zaváděné mikroelektrody. [4] Tento signál je v podstatě částečně zašuměný průběh akčních potenciálů vysílaných neurony v okolí hrotu elektrody. Ukázka signálů zachycených v různých mozkových jádrech je na obrázku 2.3



Obrázek 2.3: Ukázky signálů zachycených v různých oblastech mozkového centra *globus pallidus* na jedné trajektorii elektrody. [18]

Analýzu signálu z mikroelektrody v současnosti provádí sám chirurg, a to buď vizuálně (sleduje na obrazovce v reálném čase průběh signálu), nebo akusticky, kdy je signál přiveden na reproduktor. Ze signálu pak chirurg díky své zkušenosti pozná, ve kterém mozkovém centru se právě nachází špička elektrody. Úspěch této metody však velmi závisí na schopnostech a znalostech chirurga. Proto v současnosti existuje snaha tento postup zautomatizovat, tedy vytvořit přístroj analyzující v reálném čase zachycená data, jehož výstupem bude určení mozkového centra, v němž se nachází hrot elektrody. Tento přístroj musí obsahovat tři základní algoritmy:

1. algoritmus detekce akčních potenciálů (spike detection), který v signálu nalezne

relevantní spiky a oddělí je od šumu

2. algoritmus třídění spiků (spike sorting), který roztřídí akční potenciály podle jejich tvaru a určí, z kterých neuronů pravděpodobně pocházejí

3. algoritmus, který na základě informací ze spike sorting algoritmu určí, ve kterém mozkovém centru se nachází hrot elektrody.

Kapitola 3

Problematika detekce akčních potenciálů

Uspokojivě vyřešit problém automatické detekce akčních potenciálů je poměrně obtížné, a to z následujících důvodů:

- tvary spiků i pozadí (šum) signálu jsou u každého pacienta odlišné
- odborníky vytvořené popisy spiků jsou zobecněné a zjednodušené
- protože do signálu pronikají i utlumené akční potenciály od vzdálených neuronů, je těžké jednoznačně rozhodnout, co ještě budeme považovat za relevantní spike a co už za šum - často se na tom u jednoho záznamu neshodnou ani dva odborníci. Z toho důvodu je také téměř nemožné v časové či frekvenční oblasti oddělit signál od šumu (ukázka frekvenčního spektra signálu je na obrázku 4.2b)
- odstup signálu od šumu může být v některých signálech velmi nízký. [5]

Proto se po dlouhou dobu nedařilo automatickou detekci spiků uspokojivě vyřešit; podle přehledu [5] z roku 2002 se žádný ze zmíněných algoritmů nedokázal v detekci spiků vyrovnat odborníkům. Od té doby však již proběhlo mnoho nových pokusů o vylepšení automatické detekce využívajících celou škálu různých postupů a algoritmů (některé z nich budou představeny v následující kapitole).

Od roku 2002 už žádný přehled používaných algoritmů nevyšel, a tak následující přehled vychází přímo z článků, které nové postupy navrhovaly. Tyto články však většinou pouze popisují metodu detekce a už ne samotný algoritmus (ani nenabízejí algoritmus ke stažení), a tak není možné ověřit jejich výsledky. Proto je úspěšnost mnou vytvořeného algoritmu porovnávána pouze s úspěšností běžně používaných algoritmů, které jsou veřejně dostupné i se svými zdrojovými kódy.

3.1 Přehled běžně používaných algoritmů

3.1.1 Prahování (Amplitude thresholding)

Zcela nejjednodušším způsobem detekce je tzv. *thresholding*, tedy manuální či automatické zvolení jednoho amplitudového prahu pro celý signál a označení všech míst v signálu, které ho přesahují, jako spike.

Nejdůležitějším rozhodnutím při použití této metody je zvolení prahu tak, abychom maximalizovali počet správně detekovaných spiků (přesnost, sensitivita algoritmu) a minimalizovali počet případů, kdy je šum nesprávně označen jako spike (více o tomto tématu v podkapitole 3.3).

Při programování algoritmu amplitudového prahování je potřeba vzít v potaz několik problémů, a to zejména:

- zajistit, aby algoritmus dobře fungoval pro různé signály s různou úrovní šumu. Proto je nutné velikost prahu měnit jak v závislosti na amplitudě signálu, tak v závislosti na poměru signál/šum. Tento poměr je navíc nutné správně odhadnout (viz podkapitola 3.2).

- zajistit, aby byly správně detekovány spiky jak v kladné, tak v záporné (opačné) polaritě

- protože v refrakterní periodě akčního potenciálu často dochází k zákmitu napětí do opačné polarity (viz obrázek 2.1), je nutné naprogramovat algoritmus tak, aby tento zákmit také nebyl označen jako samostatný spike.

- zároveň je ale třeba počítat s možností, kdy jsou v signálu bezprostředně za sebou dva spiky (od různých neuronů). Odlišit od sebe zákmit akčního potenciálu v refrakterní periodě od druhého, slabšího spiku je proto poměrně obtížné.

Hlavní výhodou amplitudového prahování je jeho jednoduchost, rychlost a výpočetní nenáročnost, což umožňuje jeho snadnou hardwarovou implementaci. Jeho nevýhodami jsou vysoká citlivost na šum a obzvláště na posun nuly. [6] Metodu amplitudového prahování používá pro detekci spiků i balík *Wave_clus*.

3.1.2 Nonlinear energy operator (NEO)

Často používané rozšíření předchozího algoritmu se nazývá *Nonlinear Energy Operator (NEO)*. Tato metoda je amplitudovému prahování velmi podobná; spočívá ve výpočtu

energie každého vzorku signálu pomocí rovnice (3.1) a následném prahování této energie. [7]

$$\Psi[n] = x^2[n] - x[n+1]x[n-1] \tag{3.1}$$

kde Ψ je *nonlinear energy operator*.

Tento algoritmus v podstatě v signálu nachází vysoké špičky. Při výpočtu energie vzorku není nutné uvažovat jen předcházející a následující vzorek signálu, je možné počítat energii i využitím vzdálenějších vzorků pomocí rovnice (3.2):

$$\Psi[n] = x^2[n] - x[n+k]x[n-k] \tag{3.2}$$

kde k je libovolné (malé) přirozené číslo, které volíme s ohledem na vzorkovací periodu signálu.

Oproti prostému amplitudovému prahování může být správně nastavená metoda NEO přesnější, neřeší však problém citlivosti na šum - zatímco amplitudová detekce chybně detekuje místa, kde náhodný šum přesáhne přes stanovený práh, NEO chybně detekuje místa, kde v šumu náhodně vzniknou vysoké špičky. Úspěšnost těchto dvou algoritmů tedy silně závisí na tvaru šumu ve zkoumaném signálu. NEO oproti prahování není citlivé na posun nuly; je ovšem zase citlivé na diskontinuity v signálu. [6] Druhou nevýhodou NEO je jeho větší výpočetní složitost.

Dále je složitější NEO správně nastavit - zatímco u prahování stačilo zvolit jeden parametr (výška prahu), u NEO už musíme nastavovat dva parametry (k z rovnice (3.1) a ještě výšku prahu na upraveném signálu). I když je možné použít $k = 1$, většinou to není žádoucí, protože signály mohou mít různou vzorkovací periodu a také algoritmus při $k = 1$ detekuje příliš mnoho šumu jako spiky.

NEO je možné počítat i z více než dvou okolních vzorků signálu, pak už se ale algoritmus začíná podobat spíše metodě *template matching*.

3.1.3 Template matching

Hledání spiků v signálu podle jejich tvaru, neboli *template matching*, je dalším používaným způsobem detekce. Využívá se při něm manuálně vytvořená databáze tvarů spiků, se kterou se průběžně porovnávají části signálu. Opět je nutné při použití této metody nastavovat práh detekce (kdy je část signálu dostatečně podobná některému z tvarů spiku).

S touto metodou je možné dosáhnout mnohem lepších výsledků než pouhým prahováním. [8] Protože však pro každý neuron či skupinu neuronů je charakteristický jiný tvar spiků a tyto tvary se liší i mezi pacienty, je při použití této metody nutné manuálně

vytvářet databázi tvarů spiků pro každý analyzovaný signál. Proto je tato metoda nevhodná pro automatickou detekci spiků. [6]

3.1.4 Další algoritmy

Kompletní přehled používaných algoritmů detekce je nad rámec této práce. Většina běžně používaných metod detekce spiků (krom jmenovaných) však spočívá jen v transformaci signálu mající za cíl zvýšit odstup signálu od šumu a následném prahování upraveného signálu. Z používaných transformací signálu mohu jmenovat například *morphological filtering*, [6] *deconfusion*, [9] či vlnkovou transformaci *(wavelet transform)*, [10] kterou při detekci spiků využívá i balík funkcí *Osort*, jehož algoritmus detekce je veřejně dostupný. [11]

3.2 Detekce odstupu signál/šum

Pro správné nastavení prahu při amplitudovém prahování je potřeba odhadnout poměr mezi signálem a šumem - čím je vyšší hladina šumu v signálu, tím výše musíme nastavit práh, aby nedocházelo k velkému množství chybných detekcí.

Obvyklou metodou je vytvoření histogramu signálu a pomocí statistických metod určení, nakolik odlehlé hodnoty od průměru budeme považovat za spike. Obvykle je nastavován amplitudový práh na $\pm\ 3\ -\ 4\ \sigma$, kde σ je směrodatná odchylka signálu. Násobek směrodatné odchylky je obvykle nutné pro každý signál nastavit manuálně, protože ideální násobek se mění s frekvencí spiků v signálu. [12] Tento postup je však odvozen z předpokladu, že signál má normální rozdělení, což u mikroEEG záznamů vždy splněno není.

Je možné nastavit práh přímo i na určitý násobek vybraného percentilu signálu. Takto nastavený práh může fungovat dobře na podobných signálech s podobnou frekvencí spiků, avšak detekce se silně zhorší, pokud algoritmus narazí na signál s menším množstvím spiků - pak začne detekovat i nejvyšší špičky v šumu. To lze obejít tím, že budeme pro každý signál manuálně nastavovat násobek vybraného percentilu, pak ale přijdeme o možnost automatické detekce. Navíc nastavit tuto metodu správně pro automatickou detekci je složitější, protože má dva parametry (percentil a násobek).

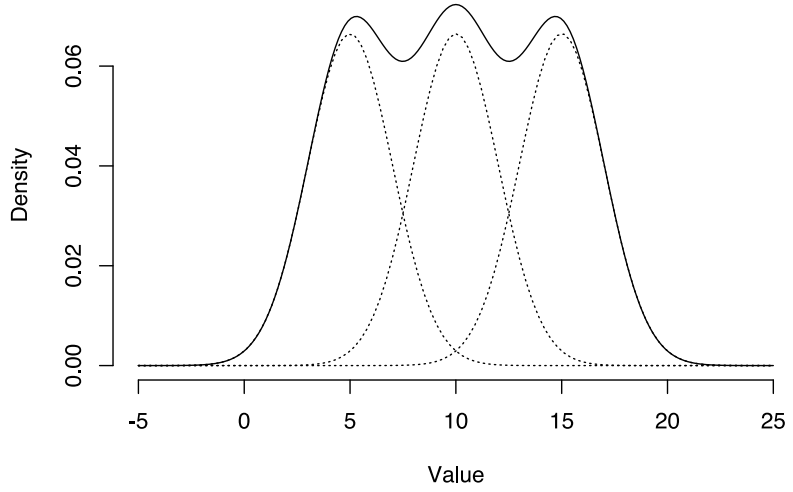
Tuto metodu využívá i balík *Wave_clus*, jehož algoritmus detekce používá 50. percentil (medián) absolutní hodnoty signálu, který je posléze násoben parametrem, který zadá uživatel.

Další používanou statistickou metodou, která umožňuje odhad hladiny šumu i v signálu s nenormálním rozdělením je mezikvartilové rozpětí (*interquartile range, IQR*):

$$IQR = Q_3 - Q_1 \tag{3.3}$$

kde Q_1 je první kvartil a Q_3 třetí kvartil signálu.

Pro detekci odstupu signál/šum je možné použít i tzv. metodu gaussovských směsí. Ta vychází z předpokladu, že histogram signálu o libovolném tvaru lze rozložit na součet Gaussových křivek o daném průměru a směrodatné odchylce a že šum v signálu má normální rozdělení. Algoritmus se pak snaží histogram signálu rozložit na jedno nebo více normálních rozdělení představujících šum a hledá v nich nepravidelnosti, které by měly představovat signál. Poměr signál/šum se pak vypočte pomocí průměrných hodnot a směrodatných odchylek těchto křivek.



Obrázek 3.1: Ukázka metody gaussovských směsí. Histogram signálu (plná čára) je rozložen na tři Gaussovy křivky (tečkované čáry). [19]

3.3 Hodnocení úspěšnosti algoritmů při detekci

V medicínském prostředí je obvyklé hodnotit úspěšnost binárních klasifikačních testů pomocí dvou hlavních kritérií:

Senzitivita neboli úspěšnost testu, která je definována následovně:

$$senzitivita = \frac{TP}{P} \tag{3.4}$$

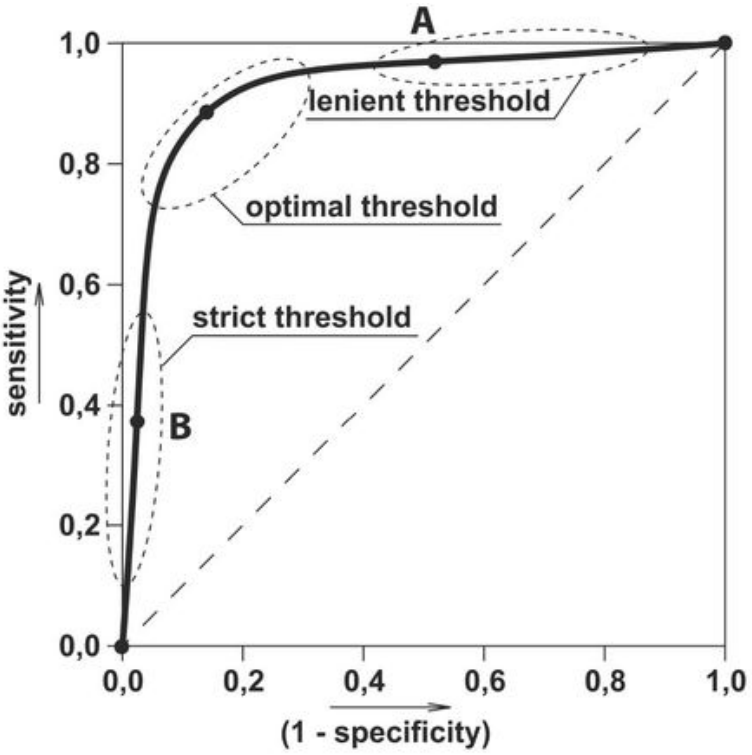
kde TP je počet správně detekovaných pozitivních případů (true positives) a P je celkový počet pozitivních případů.

Specificita neboli úspěšnost detekce negativních jevů je definována následovně:

$$specificita = \frac{TN}{N} \tag{3.5}$$

kde TN je počet správně detekovaných negativních případů (true negatives) a N je celkový počet negativních případů.

Z těchto dvou (a někdy i několika dalších pomocných) hodnot se pak různými definicemi (podle aktuální potřeby) vypočítává celková úspěšnost algoritmu. Obvykle jsou si tyto dvě hodnoty přibližně nepřímo úměrné (tedy se zvyšováním sensitivity testu klesá specificita a naopak). Vykreslením závislosti sensitivity na specificitě vzniká *receiver operating characteristic*, neboli ROC křivka (viz obrázek 3.2). [13]



Obrázek 3.2: Typický průběh ROC křivky. Převzato ze článku [13].

Detekce akčních potenciálů však není binární klasifikační test v pravém slova smyslu; vzhledem k tomu, že hledáme pouze pozitivní jevy (spiky) a neexistuje definice negativního jevu (mezera mezi dvěma spiky může být libovolně dlouhá), nemůžeme určit specificitu algoritmu. Proto se obvykle pro určení úspěšnosti algoritmu uvažuje jeho sensitivita a *false positive rate*, neboli počet chybně detekovaných spiků za jednotku času.

Při výpočtu průměrné úspěšnosti algoritmu na více signálech je nutné tyto výsledky určitým způsobem váhovat, protože signály mohou být různě dlouhé a mohou mít různou

hustotu spiků - například nedetekování jednoho spiku v signálu s pěti spiky by zhoršilo výsledek mnohem více než nedetekování jednoho spiku v signálu s 500 spiky, i když jde prakticky o stejnou chybu. Obdobně celkový počet chybných detekcí závisí na délce signálu. Podle článku [14] je proto ideální váhovat výsledky testu jak podle celkového počtu spiků v signálu, tak podle délky signálu.

Běžným způsobem porovnání výsledků různých algoritmů je pak provedení párového t-testu, který ověřuje, zda je rozdíl středních hodnot výsledků těchto algoritmů roven nule. Podmínkou pro provedení t-testu je normalita srovnávaných výsledků (která je při větším počtu výsledků většinou splněna díky centrální limitní větě) a stejný rozptyl (který se dá ověřit F-testem). Pokud je střední hodnota výsledků jednoho algoritmu větší a pravděpodobnost (p-hodnota), že k tomuto výsledku došlo vlivem náhodných jevů je menší než 5 %, říkáme, že lepší výsledek tohoto algoritmu je statisticky významný. Při současném porovnávání více algoritmů je nutné vzít v úvahu i Bonferroniho korekce.

Kapitola 4

Tvorba algoritmu detekce

4.1 Trénovací a testovací signály

Pro trénování a testování algoritmu bylo použito celkem 83 uměle vytvořených signálů s vyznačenými akčními potenciály. Tyto signály pocházejí ze tří zdrojů:

1. z ukázkových dat stažených ze stránek balíku funkcí *Osort*, [11]
2. z ukázkových dat přibalených k balíku funkcí *Wave_clus*, [12]
3. z ukázkových dat stažených ze stránek katedry kybernetiky, které byly vytvořeny jako příloha k článku [15].

Signály vygenerované algoritmem z článku [15] byly navrženy tak, aby byly co nejvíce podobné reálným signálům, a proto:

- signál je tvořen akčními potenciály s různou amplitudou a náhodně dlouhou mezerou mezi nimi (při zohlednění refrakterní periody jednotlivých neuronů)
- spiky mohou mít jak kladnou, tak zápornou polaritu
- tvar každého spiku je také náhodný (vybírá se až z devíti rozdílných tvarů spiků, které byly vymodelovány na základě reálných záznamů)
- šum (jehož výška je jeden z parametrů generujícího algoritmu) je tvořen mnoha náhodně umisťovanými spiky s malou amplitudou.

U signálů přibalených k balíkům *Wave_clus* a *Osort* není uvedeno, jakým způsobem byly vygenerovány, ale vzhledem k tomu, že jsou v časové i frekvenční oblasti podobné signálům z přílohy článku [15], byly zřejmě vytvořeny podobným způsobem (ukázky signálů z jednotlivých zdrojů a jejich srovnání s reálným signálem jsou na obrázku 4.1).

Signály z přílohy článku [15] i z balíku *Wave_clus* mají vzorkovací frekvenci 24 000 Hz. Vzorkovací frekvence signálů od balíku *Osort* není uvedena, ale s přihlédnutím k jejich podobnosti v časové oblasti k ostatním signálům je zřejmě stejná.

Všechny uměle vytvořené signály pak byly rozděleny na trénovací a testovací množinu, a to tak, aby signály z každého zdroje byly zastoupeny v obou množinách pokud možno rovnoměrně. Celkem tedy bylo používáno 40 trénovacích signálů, z toho 24 z přílohy článku [15], 8 od balíku *Osort* a 8 od balíku *Wave_clus*, a 43 testovacích signálů, z kterých 25 bylo z přílohy článku [15], 4 byly od balíku *Osort* a 14 bylo od balíku *Wave_clus*.

Trénovací i testovací množinu je možné rozdělit do menších skupin, kdy v každé skupině jsou signály se stejnou polohou a tvarem spiků a liší se pouze úrovní šumu. Toto rozdělení později umožnilo lépe zobrazit závislost úspěšnosti algoritmů na úrovni šumu v signálu.

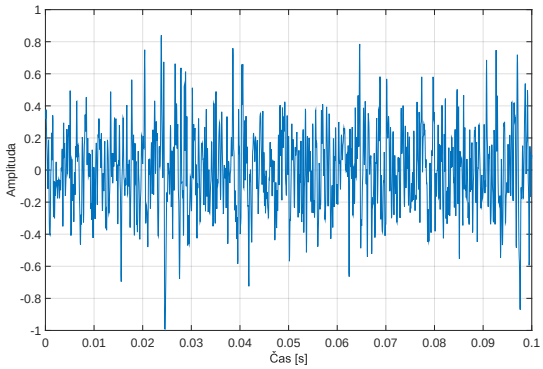
Pro porovnání charakteristik signálů a otestování funkce algoritmů bylo k dispozici i 11 reálných signálů zachycených na pacientech při zavádění elektrody do mozku. Pět z těchto signálů bylo zachyceno v mozkové oblasti STN, jeden v oblasti SNR, jeden v thalamu a čtyři v oblasti mezi mozkovými centry. Tyto signály však nebyly anotované (neměly označená skutečná umístění spiků), a proto je nebylo možné použít k trénování ani testování algoritmu.

4.2 Hledání optimálního způsobu nastavení prahu

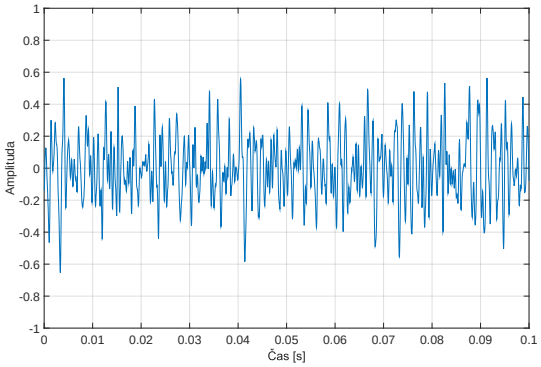
První testovací verze algoritmu byla napsána pomocí metody amplitudové detekce akčních potenciálů, protože jde o nejjednodušší metodu detekce (viz kapitola 3.1.1).

Nejprve byl tedy v programu MATLAB vytvořen jednoduchý (neautomatický) program detekce, který jako parametry přijímal signál s akčními potenciály a výšku amplitudového prahu a na výstupu vypisoval vektor s indexy vzorků, kde amplituda signálu přesahovala přes práh. Protože každý spike může být tvořen mnoha po sobě jdoucími vzorky, které přesahují přes práh, bylo nutné nějakým způsobem vybrat pokaždé jen jeden vzorek, aby jeden spike byl ve vektoru výsledků zastupován právě jedním indexem vzorku. Proto byl algoritmus naprogramován tak, aby v každé skupině po sobě jdoucích vzorků přesahujících práh nalezl amplitudové maximum signálu v absolutní hodnotě a index vzorku, ve kterém se toto maximum nachází, uložil do vektoru výsledků.

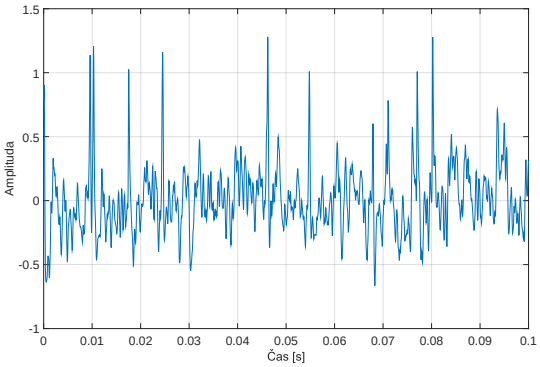
Pro vytvoření automatického algoritmu je nejprve nutné najít ideální způsob nastavování amplitudového prahu. Proto poté, co byla na manuálně zvoleném prahu ověřena funkčnost tohoto algoritmu, byl v algoritmu postupně nastavován práh pomocí různých statistických funkcí (viz kapitola 3.2) a bylo zkoumáno, při použití které funkce algoritmus dosahuje nejlepších výsledků. Úspěšnost algoritmu byla hodnocena jeho průměrným skóre na trénovací množině (o způsobu hodnocení více v kapitole 4.3).



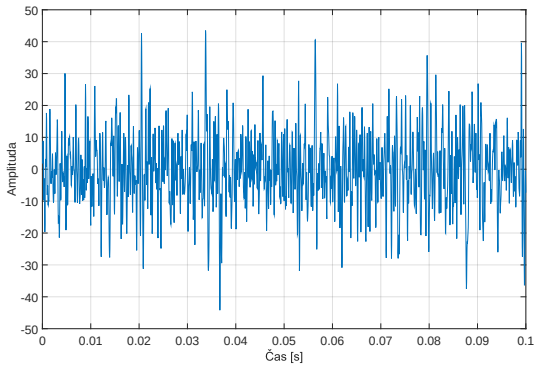
(a) Ukázka signálu z přílohy článku [15]



(b) Ukázka signálu ze stránek *Osort*

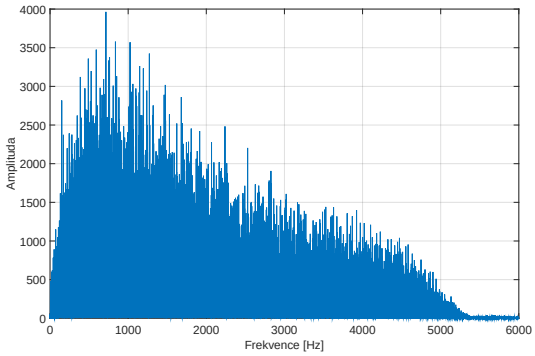


(c) Ukázka signálu od *Wave_clus*

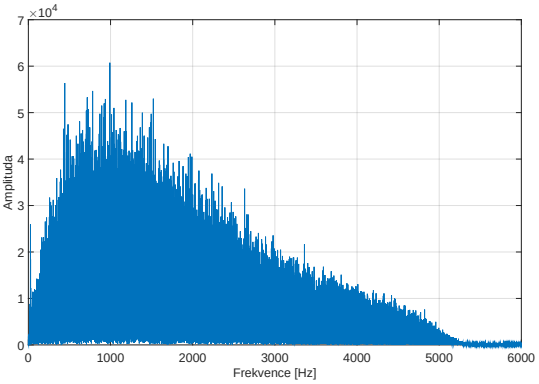


(d) Ukázka reálného signálu z STN

Obrázek 4.1: Ukázky 0,1 sekundy dlouhých úseků používaných signálů



(a) Frekvenční spektrum umělého signálu z [15]



(b) Spektrum reálného signálu z STN

Obrázek 4.2: Srovnání frekvenčního spektra reálného a vygenerovaného signálu

Pro zachování jednoduchosti a vyvarování se možnému přeučení algoritmu byla výška prahu nastavována vždy jen změnami násobku výsledku určité statistické funkce nad signálem. Práh byl tedy nastavován rovnicí:

$$threshold = k \cdot f(signal) \tag{4.1}$$

kde $f(signal)$ je použitá statistická funkce (např. *std* - směrodatná odchylka nebo *iqr* - mezikvartilové rozpětí) a k je konstanta, při které algoritmus dosahuje na trénovací množině nejlepšího výsledku (ta byla vždy nalezena hrubou silou a zaokrouhlena na jedno desetinné místo). Nebylo tedy zkoumáno, která funkce nejpřesněji odhaduje šum, ale pouze která funkce umožní nejpřesnější detekci spiků.

Po vyzkoušení více běžně používaných statistických funkcí bylo zjištěno, že optimálních výsledků algoritmus dosahuje při použití funkce mezikvartilové rozpětí, tedy při nastavování prahu pomocí funkce:

$$threshold = k \cdot iqr(signal) \tag{4.2}$$

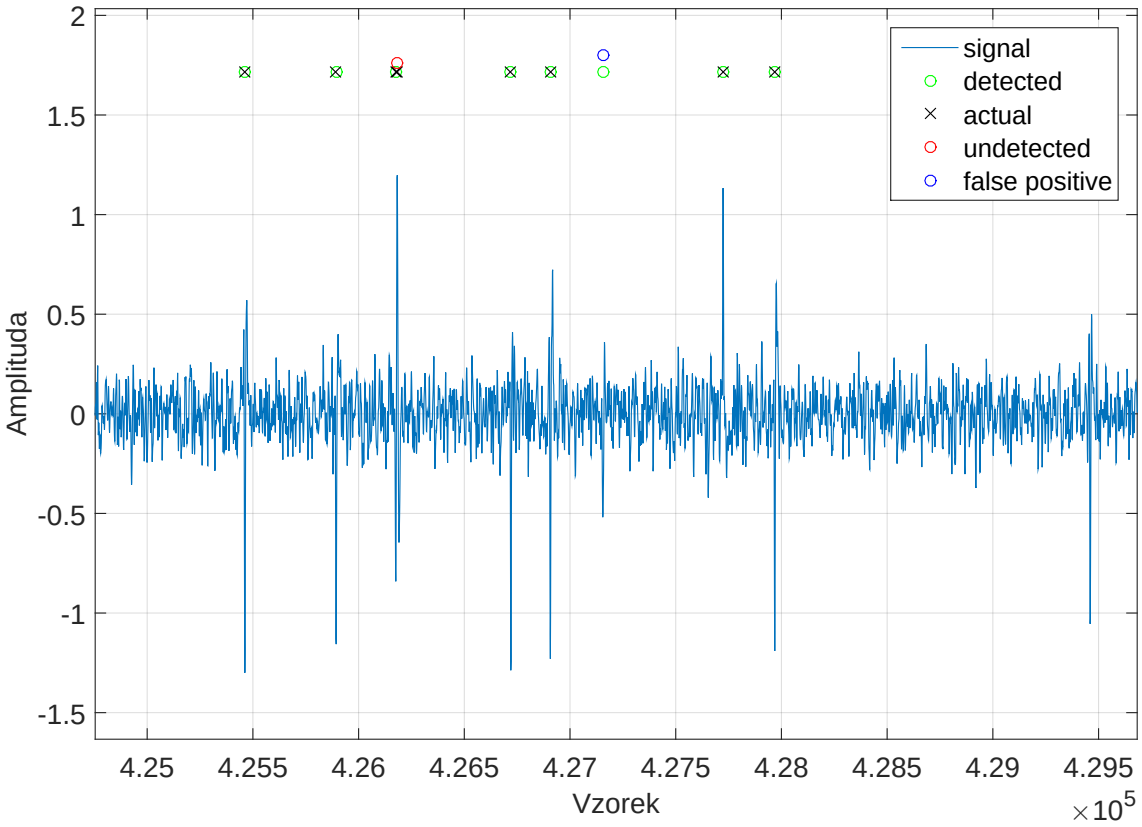
4.3 Trénování algoritmu

Aby bylo možné algoritmus trénovat, je nejprve nutné stanovit hodnoticí funkci pro posouzení výsledků detekce.

Za předpokladu, že na detekci spiků navazuje jejich třídění *(spike sorting)*, kde je možné analýzou spiků odhalit *false positives*, tedy nesprávné detekce, není při detekci spiků zcela kritické mít nízkou četnost nesprávných detekcí. Oproti tomu pokud nedojde k detekci skutečného spiku, ten už nemůže být při třídění dodatečně nalezen. Proto je důležitější mít vysokou sensitivitu detekce než mít nízkou četnost *false positives*. Pro jednoduchost byla tedy zvolena následující hodnoticí funkce:

$$score = \frac{TP - FP/2}{AP} \tag{4.3}$$

kde TP je počet správně detekovaných (nalezených) spiků, FP je počet *false positives* a AP je celkový počet spiků v signálu. Správné detekci spiků byla tedy přiřazena dvojnásobná důležitost než nízké četnosti *false positives*. Zároveň byl výsledek na každém signálu normován podle celkového počtu spiků v signálu (z důvodu podobné četnosti spiků ve všech trénovacích signálech nebylo nutné výsledky normovat i podle délky signálu). Výsledek detekce spiků v jednom signálu tedy mohl být v intervalu $(-\infty, 1\rangle$, kde výsledek 1 značí bezchybnou detekci. Protože výsledky na jednotlivých signálech už byly normované, celkový výsledek algoritmu na trénovací množině byl počítán pouze jako aritmetický



Obrázek 4.3: Ukázka zobrazení poloh spiků a chyb při detekci. Můžeme si povšimnout, že v úseku na obrázku algoritmus nedetekoval spike, který následoval těsně po předchozím spiku a naopak nesprávně detekoval špičku v šumu jako spike.

průměr jednotlivých výsledků.

Dále byl vytvořen skript, který po dokončení detekce vykreslí signál a zobrazí na něm správně detekované spiky, nedetekované spiky, *false positives* a skutečné polohy spiků (viz obrázek 4.3). Díky tomu bylo možné přímo na signálech vizuálně pozorovat, na jakých tvarech spiků či šumu dělá algoritmus nejčastěji chyby a napravovat je.

Pro účely snadného trénování a testování algoritmů byl vytvořen skript, který jako argument přijímá název algoritmu a proměnnou určující trénovací či testovací množinu a jehož výstupem je úspěšnost detekce. Tento skript zavolá uvedený algoritmus, postupně mu předá všechny signály z trénovací resp. testovací množiny a sečte celkové skóre (viz výše). Pro zrychlení tohoto procesu tento skript testuje jednotlivé signály paralelně na více jádrech procesoru.

Nakonec byl ještě vytvořen porovnávací program, který zavolá předchozí skript postupně pro všechny dostupné detekční algoritmy a vykreslí úspěšnost algoritmů v závislosti na úrovni šumu v signálu zvlášť pro každou skupinu signálů (viz obrázek 5.4).

4.4 Vylepšování algoritmu

4.4.1 Zlepšování kvality detekce

První zkoumanou metodou vylepšení úspěšnosti detekce spiků bylo filtrování signálu na vstupu algoritmu. Tato metoda vychází z předpokladu, že přestože je frekvenční spektrum zašuměného signálu prakticky spojitá křivka a nelze v ní oddělit signál od šumu (viz obrázek 4.2), mohlo by při vhodné filtraci pásmovou propustí dojít k utlumení větší části šumu a tudíž vylepšení úspěšnosti detekce.

Pro filtraci pásmovou propustí byl zvolen Butterworthův filtr, pro jehož vytvoření jsou potřeba tři parametry - spodní frekvence, horní frekvence a řád filtru. Hodnota těchto tří parametrů pak byla v cyklu postupně měněna, signál byl vytvořeným filtrem filtrován a byly hledány hodnoty těchto parametrů vedoucí k vylepšení úspěšnosti detekce. Avšak pro žádnou kombinaci těchto tří parametrů se nepodařilo na celé trénovací množině dosáhnout vylepšení detekce, a tak filtrace signálu ve finální verzi algoritmu použita nebyla. Tento neúspěch je pravděpodobně způsoben tím, že spiky mají rozložené frekvenční spektrum a skutečně nelze rozlišit frekvenční spektrum šumu a signálu. U reálných signálů je navíc už filtrace provedena hardwarově při zachycování signálu, kdy jsou z něj odstraněny artefakty, tudíž při softwarovém zpracování už není třeba.

Druhým problémem, který bylo nutné vyřešit, bylo odlišení zákmitů refrakterní periody akčních potenciálů, aby také nebyly detekovány jako spiky. Nejprve byl tento problém vyřešen prostým stanovením, že všechny vzorky vzdálené méně než 2 ms od detekovaného spiku budou z hlediska detekce ignorovány (protože celková délka spiku i s refrakterní periodou bývá 1 - 2 ms). Díky tomuto řešení už nedocházelo k nechtěným detekcím refrakterních zákmitů, avšak algoritmus také ignoroval některé skutečně spiky.

Později proběhlo několik pokusů toto řešení vylepšit, například porovnáváním výšky zákmitů s výškou předchozího spiku, ale tato implementace vedla k mnohonásobnému zpomalení detekce, přičemž úspěšnost detekce se nepodařilo vylepšit. Proto i finální verze algoritmu používá předchozí řešení, tedy ignorovat určitý počet vzorků následující po detekovaném spiku a pouze byla experimentálně určena optimální doba trvání této slepé oblasti (viz kapitola 5.1).

4.4.2 Přidání dalších metod detekce

Pro dosažení lepší úspěšnosti detekce byly do algoritmu kromě amplitudového prahování přidány ještě další, zpřesňující metody detekce. Před jejich přidáním do algoritmu bylo nejprve nutné funkci těchto metod otestovat samostatně.

Protože jde o často používanou a přitom poměrně jednoduchou metodu detekce, byla

jako druhá využita metoda *nonlinear energy operator* (NEO, viz kapitola 3.1.2). Před jejím použitím je nejprve nutné určit, z jak vzdálených vzorků se bude NEO počítat (určit parametr k z rovnice 3.2). Experimentálně bylo zjištěno, že optimální hodnota parametru k odpovídá přibližně čtvrtině obvyklé délky spiku, což je 9 vzorků při použité vzorkovací frekvenci 24 000 Hz. Další postup byl stejný jako při programování amplitudové detekce - byl napsán samotný skript a zkoumáno při použití které funkce odhadu šumu v signálu dojde k nejlepší detekci na trénovací množině. Bylo zjištěno, že je to ta samá, jako v případě amplitudové detekce, tedy mezikvartilové rozpětí. Ostatní nastavení algoritmu byla použita stejná, jako v případě amplitudové detekce.

Třetí metoda není běžně používána a byla vytvořena v rámci této práce. Prakticky jde o vyhledávání náběžných a sestupných hran spiků v signálu: algoritmus sčítá diference osmi prvků nalevo od aktuálního vzorku a odečítá od nich diference osmi prvků napravo (to celé v absolutní hodnotě, aby mohly být detekovány i spiky v opačné polaritě). Tento mezivýsledek značící součet výšky náběžné a sestupné hrany spiku (v pevném okně o celkové šířce 16 vzorků) je poté opět prahován experimentálně zjištěným násobkem mezikvartilového rozpětí signálu. Aby byla tato metoda univerzální, počet sčítaných prvků (šířka okna) je měněn v závislosti na vzorkovací frekvenci signálu. I když tato metoda není rozšířená, v úspěšnosti detekce dosahuje podobných výsledků, jako amplitudové prahování (viz výsledky).

Po ověření samostatné funkčnosti všech tří metod detekce byly tyto metody spojeny do jednoho algoritmu. Toto spojení bylo vyřešeno průměrováním skóre jednotlivých metod, kdy jednotlivá dílčí skóre se rovnají jedné právě tehdy, když je hodnota výsledku metody rovna příslušnému prahu (například u metody amplitudové detekce je výška aktuálního vzorku vydělena hodnotou amplitudového prahu). Vzorky s podprahovou amplitudou pak budou mít dílčí skóre menší než jedna a s nadprahovou větší než jedna. Jako spiky pak budou označena ta místa, která mají průměrné skóre větší nebo rovno jedné.

Z důvodu vysoké časové náročnosti výpočtu výsledku třetí metody byl algoritmus rozdělen na dvě verze, z nichž jedna tuto metodu pro detekci (spolu s ostatními metodami) využívá, a druhá ji přeskakuje (viz kapitola 5 a tabulka 5.2).

4.4.3 Optimalizace algoritmu

Pro zkrácení doby trvání detekce spiků a také pro zjednodušení kódu byla provedena několikanásobná refaktorizace algoritmu.

Algoritmus funguje iterativně, tedy postupně prochází všechny vzorky signálu a každý zkoumá na přítomnosti spiku, a tak je pro zvýšení rychlosti algoritmu nutné minimalizovat průměrný počet operací, které algoritmus v každém cyklu vykoná. Jako první byla tedy

na začátek cyklu přidána podmínka, že aby aktuální vzorek mohl být dál zkoumán, musí být lokálním maximem či minimem, jinak algoritmus pokračuje v dalším cyklu. Tím bylo dosaženo hned dvou zlepšení - algoritmus už nemusí zbytečně počítat skóre vzorků, které nejsou vrcholy spiků, a odpadla nutnost posléze upravovat vektor výsledků, protože každý spike je v něm zastoupen právě jedním vzorkem (v jeho vrcholu). Tato podmínka byla poté pro další zrychlení rozdělena na dvě, tedy algoritmus nejprve testuje, jestli předchozí vzorek nemá větší amplitudu než aktuální a poté stejně pro následující vzorek.

Dále byl kód všech tří dílčích metod sloučen do jedné dlouhé podmínky. Na místě, kde se původně počítalo dílčí skóre každé metody do samostatných proměnných, pak se sečetlo celkové skóre, vydělilo třemi a porovnalo s jedničkou, je nyní jedna podmínka, ve které se přímo sčítají výsledky uvedených tří metod (bez ukládání do proměnné) a porovnávají se s číslem 3.

Před vstupem do cyklu je také prováděna alokace paměti pro vektor výsledků a pro proměnné (kvůli tomu je nutné posléze po skončení cyklu oříznout nuly na konci vektoru výsledků).

4.5 Popis hotového algoritmu

Algoritmus přijímá na vstupu dva parametry - záznam mikroelektrodového signálu jako vektor prvků a strukturu *handles*, která může obsahovat všechny pomocné parametry, jako např. vzorkovací frekvenci signálu, proměnnou určující jestli se má měřit a vypisovat čas detekce či která verze algoritmu se má použít apod. Tato struktura parametrů je nepovinná a pokud ji uživatel nezadá, vytvoří se podle uložených výchozích hodnot.

V samotném algoritmu se nejprve deklarují proměnné (jako například vzorkovací frekvence) a naplní se hodnotami ze struktury *handles*. Poté se alokuje vektor výsledků - jeho implicitní velikost je délka signálu vydělená 500, což je pro všechny signály z trénovací množiny dostatečná velikost. Před samotnou detekcí se ještě na začátek a konec signálu přidá počet nul odpovídající polovině délky okna používaného u třetí metody detekce, aby mohly být otestovány i prvky na začátku a konci signálu.

Poté již následuje samotný cyklus, ve kterém se iterativně projdou všechny vzorky signálu. Nejprve se v něm ověřuje podmínka, jestli je aktuální vzorek lokální extrém; pokud není, zbytek kódu v cyklu se přeskakuje a detekce pokračuje dalším vzorkem. Pokud vzorek je lokální extrém, jako druhá podmínka se kontroluje, jestli uběhl dostatečný počet vzorků od posledního detekovaného spiku (viz kapitola 4.4.1); pokud ne, opět se zbytek cyklu přeskakuje.

Pokud vzorek splní obě tyto podmínky, následuje jeho ohodnocení uvedenými třemi (resp. dvěma, v závislosti na zvolené verzi algoritmu) metodami detekce. Když je průměrné

skóre vzorku menší než jedna, opět se přeskočí zbytek cyklu. V opačném případě byl detekován spike, a tak se index aktuálního vzorku zapíše do vektoru výsledků a inkrementuje se proměnná sčítající počet dosud detekovaných spiků v signálu (tato proměnná se využívá při testování podmínky, jestli je aktuální vzorek dostatečně vzdálen od posledního detekovaného spiku).

Po skončení cyklu už dojde jen k ořezu nul na konci vektoru výsledků a předání řízení volající funkci.

Kapitola 5

Výsledky a diskuse

5.1 Shrnutí funkce algoritmu

Navrhovaný algoritmus využívá pro detekci spiků současně tři metody detekce, jejichž dílčí skóre (vyjádřené reálným číslem) průměruje. Dílčí skóre se počítají vydělením výsledku dané metody příslušným prahem (při rovnosti výsledku s prahem je tedy dílčí skóre 1). Pokud je pak průměrné skóre vzorku větší nebo rovno jedné, je vzorek označen za spike. Použité metody detekce jsou:

amplitudové prahování: amplituda vzorku je porovnávána s hodnotou amplitudového prahu (viz kapitola 3.1.1)

***nonlinear energy operator*:** hodnota NEO vzorku (viz kapitola 3.1.2) je porovnávána s hodnotou NEO prahu

prahování výšky hran: součet výšky náběžné a sestupné hrany spiku v okně o pevné délce (viz kapitola 4.4.2) je porovnáván s hodnotou třetího prahu.

Každá z těchto metod má svou vlastní hodnotu prahu, který se pro každý signál zvlášť nastavuje funkcí:

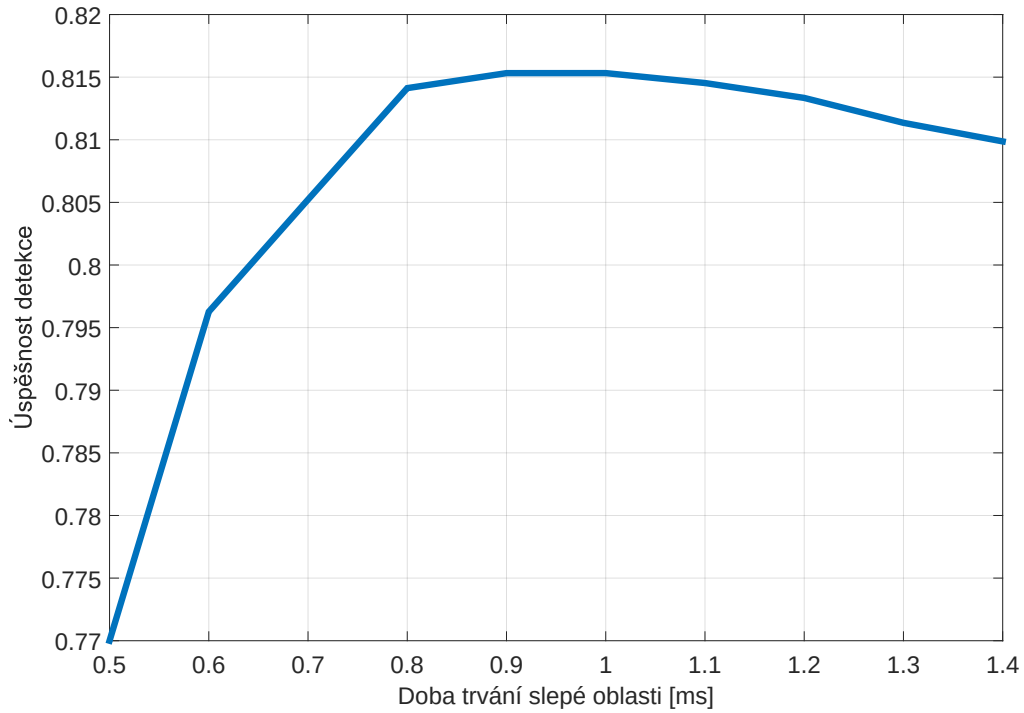
$$threshold = k \ \cdot iqr(signal) \tag{5.1}$$

kde *iqr(signal)* je mezikvartilové rozpětí signálu a k je neměnná ideální hodnota násobku experimentálně zjištěná na trénovací množině. Každá metoda detekce má tedy svou vlastní konstantu k.

Vzhledem k tomu, že při použití poslední jmenované detekční metody (prahování výšky hran) dochází k více než dvojnásobnému zpomalení rychlosti detekce (viz tabulka 5.2), přijímá algoritmus parametr určující, jestli se má při detekci tato metoda brát v potaz

či přeskakovat. V praxi je toto kvůli snížení výpočetní náročnosti algoritmu vyřešeno rozdělením algoritmu na dvě funkce s identickou strukturou, z nichž jedna pro výpočet skóre vzorku využívá všechny tři metody (funkce *detectPrecise*) a druhá jen první dvě metody (funkce *detectFast*).

Problém rozlišení zákmitu refrakterní periody akčního potenciálu od druhého spiku je řešen ignorováním určitého počtu vzorků po každém detekovaném spiku. Experimentálně zjištěná optimální doba trvání této slepé oblasti (tak, aby algoritmus měl co nejvyšší úspěšnost detekce) je 0.9 ms, což je při vzorkovací frekvenci 24 000 Hz přibližně 21 vzorků (viz obrázek 5.1).



Obrázek 5.1: Graf závislosti úspěšnosti detekce na trénovací množině v závislosti na době trvání slepé oblasti po každém detekovaném spiku.

5.2 Porovnání úspěšnosti detekce

V tomto porovnání jsou srovnávány obě verze navrhovaného algoritmu (*detectFast* i *detectPrecise*) s dvěma veřejně dostupnými a běžně používanými algoritmy detekce - jeden z nich pochází z balíku funkcí *Osort* a druhý z balíku funkcí *Wave_clus*.

Parametry obou verzí navrhovaného algoritmu byly na trénovací množině natrénovány na jejich optimální hodnotu (dle navrhovaného kritéria hodnocení úspěšnosti, viz kapitola 3.3) a tyto hodnoty pak byly používány pro detekci na všech signálech v testovací množině.

Detekční algoritmus *Wave_clus* není zcela automatický, pro uspokojivou úspěšnost detekce je nutné, aby uživatel měnil konstantu určující výšku prahu pro každý signál. Dále je možné v algoritmu měnit další parametry, zejména okrajové frekvence filtru pásmové propusti, který je v algoritmu obsažen. Všechny tyto parametry byly stejně jako u navrhovaného algoritmu natrénovány na jejich optimální hodnoty, které byly posléze používány používány pro všechny signály při testování úspěšnosti detekce.

Detekční algoritmus *Osort* funguje automaticky (nevyžaduje od uživatele zadání žádné konstanty). Před detekcí si může uživatel vybrat, kterou metodu detekce algoritmus použije - algoritmus obsahuje algoritmy amplitudového prahování, prahování pomocí energie signálu a prahování pomocí vlnkové transformace, přičemž pro detekci používá vždy jen právě jednu z těchto metod. Na trénovací množině dosáhl tento algoritmus nejlepšího výsledku při využití vlnkové transformace, a tak pro porovnání algoritmů byla zvolena právě verze algoritmu používající vlnkovou transformaci.

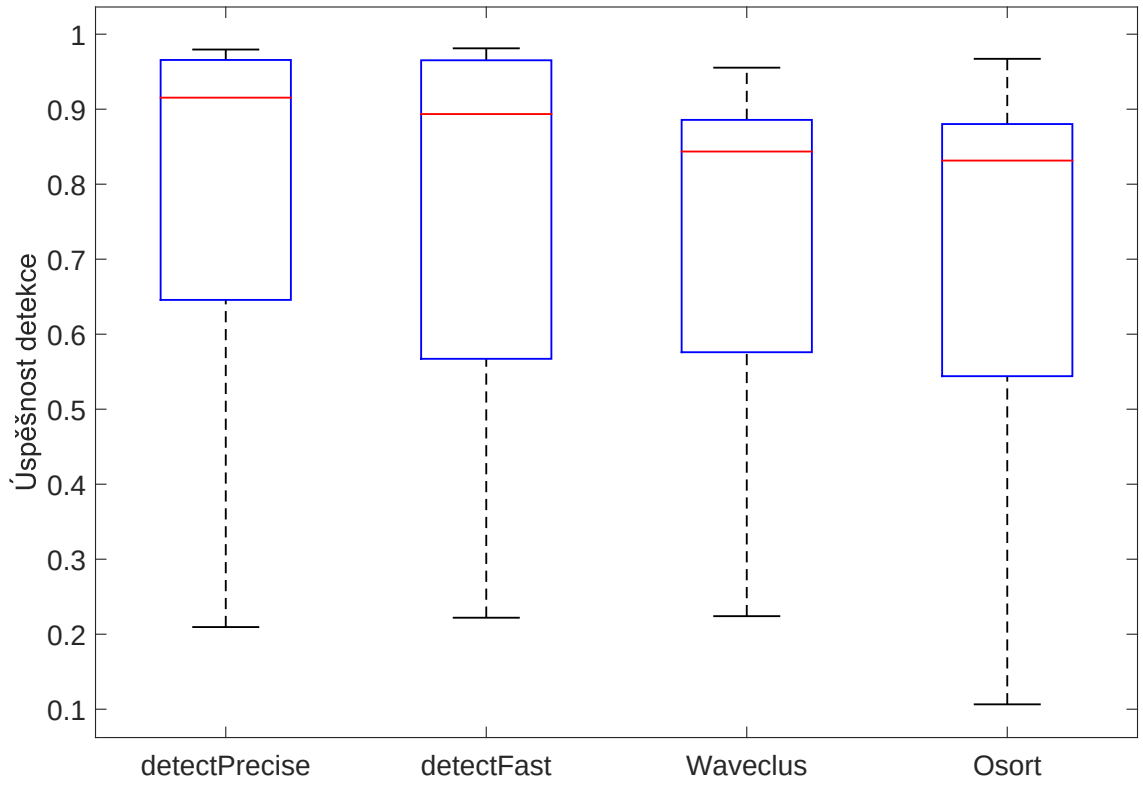
Průměrné úspěšnosti detekce spiků všech algoritmů na testovací množině jsou uvedeny v tabulce 5.1. Obě verze navrhovaného algoritmu (detectFast i detectPrecise) mají v průměru úspěšnější detekci spiků než srovnávané algoritmy detekce. Párovým t-testem výsledků algoritmů na testovací množině bylo zjištěno, že algoritmus detectFast je při detekci spiků lepší než natrénovaný algoritmus *Wave_clus* i algoritmus *Osort* s $p < 0.01$ a algoritmus detectPrecise je oproti jmenovaným algoritmům lepší s $p < 0.001$.

Algoritmus	Úspěšnost	Změna
detectPrecise	77,2 %	0 %
detectFast	75,2 %	-2,6 %
Wave_Clus	72,0 %	-6,7 %
Osort	70,5 %	-8,7 %

Tabulka 5.1: Průměrné úspěšnosti detekce algoritmů na testovací množině a procentuální změna v úspěšnosti oproti algoritmu detectPrecise

Pro měnící se odstup signálu od šumu mají algoritmy různou úspěšnost. Zatímco u signálů s vysokým odstupem signálu od šumu může úspěšnost algoritmů přesahovat 98 %, u silně zašuměných signálů může úspěšnost klesnout pod 25 %. Krabicový diagram všech výsledků detekce srovnávaných algoritmů na testovací množině je na obrázku 5.2. Na něm můžeme vidět, že i když jsou obě verze navrhovaného algoritmu v průměru lepší než algoritmus *Wave_clus*, mají také větší rozptyl výsledků a o něco hůře detekují spiky v silně zašuměných signálech.

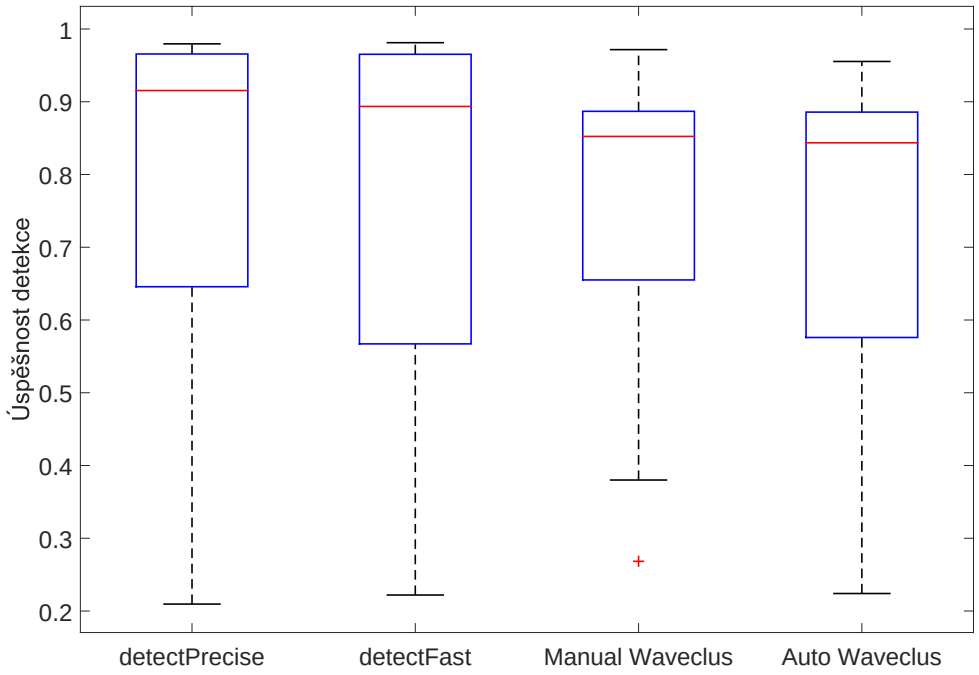
Verze navrhovaného algoritmu detectPrecise v úspěšnosti detekce překonala algoritmus z balíku *Wave_clus* i tehdy, když byly jeho parametry nastavovány zvlášť na každý signál z testovací množiny tak, aby úspěšnost jeho detekce byla co nejvyšší (obdobně, jako



Obrázek 5.2: Krabicový diagram (*boxplot*) výsledků detekce srovnávaných algoritmů na testovací množině.

je manuálně nastavován při jeho skutečném používání). Jeho průměrná úspěšnost na testovací množně je pak 76,2 % (viz obrázek 5.3), ovšem tento výsledek nedosáhl statistické významnosti.

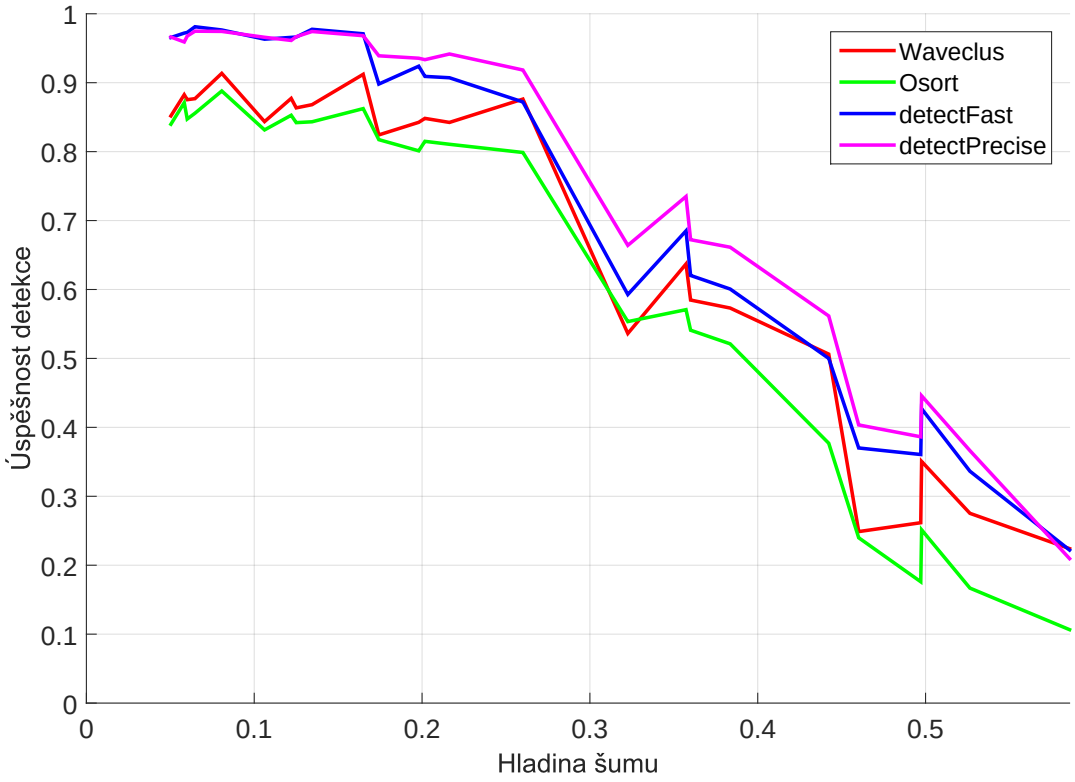
Algoritmy se také částečně liší svou úspěšností při detekci spiků na signálech z různých zdrojů. V tomto je nejvíce znatelný rozdíl u algoritmu z balíku *Osort*, jehož úspěšnost detekce je v průměru horší než u ostatních srovnávaných algoritmů, ovšem na signálech stažených ze stránek balíku *Osort* detekuje spiky nejlépe ze srovnávaných algoritmů (viz obrázek 5.4b). To je pravděpodobně způsobeno tím, že na těchto (či obdobných) signálech byl detekční algoritmus *Osort* trénován. Také si můžeme na obrázku 5.4c povšimnout, že obě verze navrhovaného algoritmu mají na silně zašuměných signálech pocházejících ze stránek *Wave_clus* horší úspěšnost detekce než algoritmy *Wave_clus* i *Osort*. To může být způsobeno nízkým zastoupením tohoto typu signálu v trénovací množině. Na signálech z přílohy článku [15], kterých byl v trénovací množině nejvyšší počet, mají naopak obě verze navrhovaného algoritmu znatelně lepší úspěšnost detekce než srovnávané algoritmy (obrázek 5.4a). Srovnání doby trvání detekce spiků srovnávaných algoritmů nalezneme v tabulce 5.2.



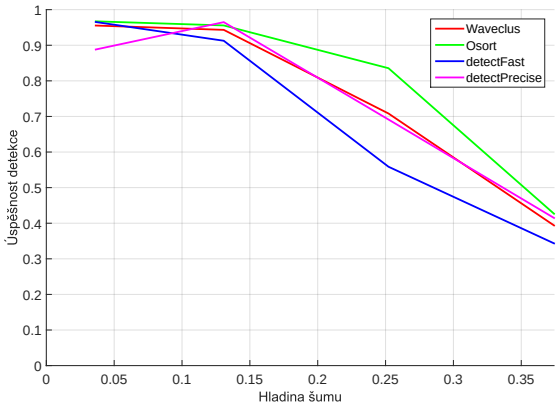
Obrázek 5.3: Krabicový diagram (*boxplot*) výsledků detekce algoritmů a jejich srovnání s manuálně nastavovaným algoritmem *Wave_clus*. *Auto Waveclus* zde značí automaticky pracující algoritmus *Wave_clus*, který používá stejné hodnoty parametrů pro všechny signály (identický s algoritmem *Waveclus* na obrázku 5.2) a *Manual Waveclus* zde značí výsledky algoritmu *Wave_clus*, když se pro každý signál nastaví parametry na ideální hodnoty pro daný signál.

Algoritmus	Doba výpočtu	Podíl	Sekund signálu/s
Osort	89 s	100 %	20
Wave_Clus	56 s	63 %	31
DetectPrecise	50 s	56 %	35
DetectFast	22 s	25 %	70

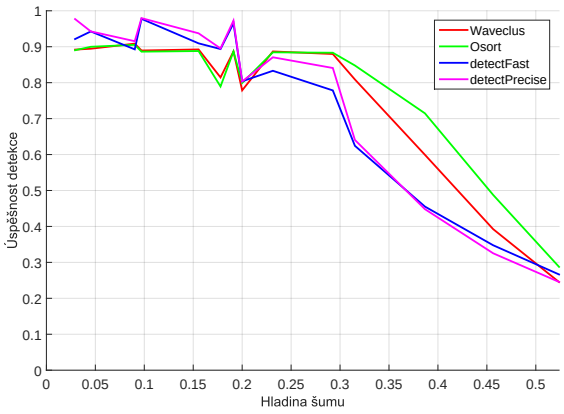
Tabulka 5.2: Doba trvání výpočtu detekce spiků na testovací množině na notebooku s dvoujádrovým procesorem Intel i5-4200M a její procentuální srovnání s algoritmem *Osort*. Třetí sloupec obsahuje informaci, kolik sekund signálu se na uvedeném notebooku zpracuje za jednu sekundu při vzorkovací frekvenci signálu 24 000 Hz.



(a) Výsledky algoritmů na signálech z přílohy článku [15]



(b) Výsledky algoritmů na signálech stažených ze stránek balíku *Osort*



(c) Výsledky algoritmů na signálech stažených ze stránek balíku *Wave_clus*

Obrázek 5.4: Srovnání výsledků detekce v závislosti na výšce šumu v signálu pro signály z tří různých zdrojů.

5.3 Použitelnost a perspektiva

V balíku *Wave_clus* bylo možné snadno nahradit původní detekční algoritmus navrženým algoritmem, aniž by byla nějak narušena funkcionalita balíku. Jako implicitní algoritmus je zvolena funkce detectPrecise, protože v praxi detekce probíhá spíše na kratších záznamech, kde na rychlosti výpočtu příliš nezáleží.

Vzhledem k tomu, že tento algoritmus nebyl vytvářen s ohledem na možnost jeho převedení do hardwarového zpracování a detekce signálu v reálném čase, bez velkých úprav kódu je jeho použití v hardwarovém přístroji nerealizovatelné. V současné podobě tedy slouží pouze k softwarovému zpracování digitalizovaného záznamu.

I přes zlepšení oproti srovnávaným algoritmům není algoritmus detectPrecise ještě zdaleka ideálně přesný. V budoucím výzkumu by bylo vhodné vylepšit algoritmus zavedením sofistikovanější metody odhadu amplitudy signálu a odstupu signál/šum. V této práci například vzhledem k její relativní složitosti nebyla zkoumána úspěšnost detekce šumu pomocí metody Gaussovských směsí, o které byla zmínka v kapitole 3.2.

Druhým místem, kde může dojít k vylepšení úspěšnosti algoritmu, je odlišení refrakterní periody spiku od jiného rychle následujícího spiku. Zde použité řešení (ignorování určitého počtu vzorků) by mohlo být problematické na signálech s vysokou četností spiků, které po sobě rychle následují.

Uměle vytvořené signály nemusí vystihovat podstatu reálných signálů zcela přesně. Ideální pro detekci spiků v reálných signálech by proto bylo algoritmus na reálných signálech natrénovat. K tomu je však nutné získat odborníky anotované reálné signály v digitální podobě.

Kapitola 6

Závěr

Cílem této práce byl vznik algoritmu detekce akčních potenciálů (spiků) v záznamech mikroelektrodových signálů, který by nahradil nevyhovující algoritmus obsažený v balíku funkcí *Wave_clus*.

Navrhovaný algoritmus byl vytvořen za pomoci tří různých metod detekce spiků: amplitudového prahování, prahování signálu transformovaného funkcí *nonlinear energy operator* a vlastní metody detekující spiky vyhledáváním náběžných a sestupných hran spiků v signálu. Pro detekci spiků je používán průměrný výsledek těchto tří metod.

Výška prahu v jednotlivých metodách je pro všechny signály nastavována na určitý násobek mezikvartilového rozpětí daného signálu. Tento násobek je pro každou metodu jiný a jeho ideální hodnota byla experimentálně zjištěna na trénovací množině uměle vytvořených signálů. Tyto hodnoty jsou neměnné a algoritmus od uživatele nevyžaduje zadání žádného dalšího parametru, jde tedy o automatický detekční algoritmus.

Porovnání úspěšnosti detekce navrhovaného algoritmu s dvěma běžně používanými a veřejně dostupnými algoritmy ukázalo, že navrhovaný algoritmus je při detekci statisticky významně úspěšnější ($p < 0.01$) a zároveň rychlejší než oba srovnávané algoritmy. Navrhovaný algoritmus byl poté úspěšně zakomponován do balíku funkcí *Wave_clus*.

V budoucím výzkumu by bylo při programování detekčního algoritmu vhodné použít sofistikovanější metody detekce šumu v signálu (například metodu Gaussovských směsí), vylepšit metodu odlišení zákmitu refrakterní periody od druhého spiku a trénovat algoritmus na anotovaných reálných signálech zachycených při hloubkové mozkové stimulaci.

Bibliografie

[1] HRUŠKA, Michal. *Fyziologie živočichů a člověka*. I. Díl. Nové vydání. Univerzita Hradec Králové, 2009.

[2] ROTH, Jan et al. *Parkinsonova nemoc*. Maxdorf, 2009.

[3] NASS, Richard a Serge PRZEDBORSKI. *Parkinson's Disease. Molecular and Therapeutic Insights From Model Systems*. Elsevier Science, 2008. URL: `http://www.sciencedirect.com/science/book/9780123740281` (cit. 16. 04. 2015).

[4] KERN, Drew S. a Rajeev KUMAR. „Deep Brain Stimulation". In: *The Neurologist* 13 (2007), 237–252. URL: `http://neuroanesthesia.ucsf.edu/residents/respdf/DBS%20review%2007.pdf` (cit. 16. 04. 2015).

[5] WILSON, Scott B. a Ronald EMERSON. „Spike detection: a review and comparison of algorithms". In: *Clinical Neurophysiology* 113 (2002), 1873–1881.

[6] LIU, Xiaofeng, Xianqiang YANG a Nanning ZHENG. „Automatic extracellular spike detection with piecewise optimal morphological filter". In: *Neurocomputing* 79 (2012), 132–139.

[7] ZARIFIA, Mohammad, Negar GHALEHJOGH a Mehdi BARADARAN-NIA. „A new evolutionary approach for neural spike detection based on genetic algorithm". In: *Expert Systems with Applications* 42 (2015), 462–467.

[8] KIM, Sunghan a James MCNAMES. „Automatic spike detection based on adaptive template matching for extracellular neural recordings". In: *Journal of Neuroscience Methods* 165 (2007), 165–174. URL: `http://www.pdx.edu/biomedical-signal-processing-lab/sites/www.pdx.edu.biomedical-signal-processing-lab/files/JNSM_Kim.pdf` (cit. 02. 05. 2015).

[9] FRANKE, Felix et al. „An online spike detection and spike classification algorithm capable of instantaneous resolution of overlapping spikes". In: *Journal of Computational Neuroscience* 29 (2010), 127–148.

[10] YANG, Yunning, Awais KAMBOH a Andrew J. MASON. „Adaptive threshold spike detection using stationary wavelet transform for neural recording implants". In: *Biomedical Circuits and Systems Conference (BioCAS), 2010 IEEE.* URL: `http://ieeexplore.ieee.org/xpl/articleDetails.jsp?reload=true&arnumber=5709558` (cit. 20. 04. 2015).

[11] RUTISHAUSER, Ueli, Erin M. SCHUMAN a Adam N. MAMELAK. „Online detection and sorting of extracellularly recorded action potentials in human medial temporal lobe recordings, in vivo". In: *Journal of Neuroscience Methods* 154 (2006), 204–224. URL: `http://www.urut.ch/new/serendipity/index.php?/pages/osort.html` (cit. 25. 04. 2015).

[12] QUIROGA, R. Q., Z. NADASDY a Y. BEN-SHAUL. „Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering". In: *Neural Computation* 16 (2004), 1661–1687.

[13] VRÁNOVÁ, J. et al. „ROC analýza a využití analýzy nákladů a přínosů k určení optimálního dělícího bodu". In: *Časopis lékařů českých* 148 (2009), 410–415. URL: `http://www.prolekare.cz/pdf?id=5403` (cit. 24. 04. 2015).

[14] CASSON, Alexander J., Elena LUNA a Esther RODRIGUEZ-VILLEGAS. „Performance metrics for the accurate characterisation of interictal spike detection algorithms". In: *Journal of Neuroscience Methods* 177 (2009), 479–487.

[15] WILD, Jiří et al. „Performance comparison of extracellular spike sorting algorithms for single-channel recordings". In: *Journal of Neuroscience Methods* 203 (2012), 369–376. URL: `https://nit.felk.cvut.cz/~wildj1/ssc/` (cit. 08. 05. 2015).

Zdroje obrázků

[16] *Action potential*. Original by en:User:Chris73, updated by en:User:Diberri, converted to SVG by tiZom - Own work. Licensed under CC BY-SA 3.0 via Wikimedia Commons. URL: `https://commons.wikimedia.org/wiki/File:Action_potential.svg#/media/File:Action_potential.svg` (cit. 18. 04. 2015).

[17] *Deep Brain Stimulation*. Neznámý autor, převzato ze stránek Univeristy of Texas. URL: `http://www.neurosurgery.uthscsa.edu/display_patients.php?ps_id=20&pg=` (cit. 23. 04. 2015).

[18] *Processing of neuronal single-cell recordings from deep structures of human brain*. Vytvořil Jonathan Dostrovsky, převzato ze stránek katedry kybernetiky. URL: `http://nit.felk.cvut.cz/~xnovakd1/Projects/Parkinson.html` (cit. 03. 05. 2015).

[19] *Gaussian mixture example*. By Smason79 (Own work). Licensed under CC BY-SA 3.0 via Wikimedia Commons. URL: `https://commons.wikimedia.org/wiki/File:Gaussian-mixture-example.svg` (cit. 03. 05. 2015).