

ZÁPADOČESKÁ UNIVERZITA V PLZNI
FAKULTA ELEKTROTECHNICKÁ

Katedra elektroenergetiky a ekologie

DIPLOMOVÁ PRÁCE

Klasifikace částečných výbojů

ZÁPADOČESKÁ UNIVERZITA V PLZNI

Fakulta elektrotechnická
Akademický rok: 2019/2020

ZADÁNÍ DIPLOMOVÉ PRÁCE (projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení: **Bc. Václav MAYER**
Osobní číslo: **E18N0055P**
Studijní program: **N2612 Elektrotechnika a informatika**
Studijní obor: **Elektroenergetika**
Téma práce: **Klasifikace částečných výbojů**
Zadávací katedra: **Katedra elektroenergetiky a ekologie**

Zásady pro vypracování

1. Zpracujte přehled v současné době používaných způsobů měření a vyhodnocování výbojové činnosti v elektrických zařízeních vn s ohledem na platné normativní dokumenty i nové trendy v této oblasti.
2. Diskutujte metodiku měření a možnosti vyhodnocení a interpretace změřených veličin.
3. Analyzujte používané klasifikační techniky a využijte možností měřicího systému částečných výbojů a digitálního osciloskopu k posouzení některých znaků používaných pro klasifikaci výbojů.

Rozsah diplomové práce: **40 – 60 stran**
Rozsah grafických prací: **podle doporučení vedoucího**
Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. <https://www.sciencedirect.com/science/article/pii/S0263224115000901>

Vedoucí diplomové práce: **Doc. Ing. Eva Müllerová, Ph.D.**
Katedra elektroenergetiky a ekologie

Datum zadání diplomové práce: **4. října 2019**
Termín odevzdání diplomové práce: **28. května 2020**





Prof. Ing. Zdeněk Peroutka, Ph.D.
děkan

L.S.



Doc. Ing. Karel Noháč, Ph.D.
vedoucí katedry

Abstrakt

Předkládaná práce se zabývá novými možnostmi klasifikace na poli počítačového zpracování dat z měření částečných výbojů. Úkolem práce je popsat používané možnosti parametrizace a klasifikace, jaké jsou k dispozici pro vyhodnocení měření a také jejich výhody a nevýhody. Dalším úkolem je vybrat a následně aplikovat jednu z možných metod na změřená data a zhodnotit spolehlivost této metody.

Práce je rozdělena do několika částí. V první části jsou zmíněny možnosti, jakými lze částečné výboje měřit. Dále je zpracována část, ve které jsou sepsány způsoby, jakými lze popsat změřená data tak, aby byla charakterizována co nejméně parametry a usnadňovala klasifikaci. V části s klasifikačními metodami jsou poté sepsány základní možnosti toho, jak tyto parametry mezi sebou porovnávat a vyvozovat z nich nějaké závěry. Poslední část se zabývá praktickou aplikací vybrané metody v praxi a detailnějším popisem této metody.

Klíčová slova

Klasifikace, částečné výboje, AC, DC, statistické zpracování, neuronové sítě

Abstract

The master thesis focuses on application of new classification methods in the field of computer processing of data from partial discharge measurement. The first task of this thesis is to describe feature extracting and classification methods which can be used for the evaluation of the measurement and their advantages and disadvantages. The second task is to choose one method, apply it to measured data and evaluate the reliability of this method.

The thesis is divided into several parts. The possibilities of partial discharge measuring are mentioned in the first part. The second part describes ways how we can extract features from measured data to characterize them by only several parameters. Furthermore, the part with classification methods describes possibilities how we can compare computed features and summarize whole classification process. The last part deals with the practical application of one selected method and give advanced description how the classification process was achieved.

Key words

Classification, partial discharges, AC, DC, statistic evaluation, neural networks

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně, s použitím odborné literatury a pramenů uvedených v seznamu, který je součástí této diplomové práce.

Dále prohlašuji, že veškerý software, použitý při řešení této diplomové práce, je legální.

.....
podpis

V Plzni dne 18.6.2020

Mayer Václav

Poděkování

Tímto bych rád poděkoval vedoucí mé diplomové práce doc. Ing. Evě Müllerové, Ph.D. za metodické vedení práce, cenné profesionální rady a připomínky k práci.

Obsah

1.	ÚVOD DO PROBLEMATIKY ČÁSTEČNÝCH VÝBOJŮ	12
2.	ZÁKLADNÍ TYPY ČÁSTEČNÝCH VÝBOJŮ A JEJICH CHARAKTERISTIKY	13
2.1	PLOVOUCÍ ELEKTRODA.....	13
2.2	BUBLINA V IZOLACI.....	13
2.3	POVRCHOVÝ VÝBOJ	13
2.4	KORÓNA.....	13
3.	ZÁKLADNÍ ROZDĚLENÍ MOŽNOSTÍ MĚŘENÍ.....	14
3.1	AKUSTICKÁ DETEKCE	14
3.2	OPTICKÁ DETEKCE	15
3.3	UHF DETEKCE	16
3.4	MĚŘENÍ POMOCÍ TEV METODY	19
3.5	HFCT.....	20
3.6	DGA METODY.....	21
4.	MOŽNOSTI GALVANICKÉHO MĚŘENÍ.....	24
4.1	OBEČNÉ PARAMETRY URČOVANÉ PŘI MĚŘENÍ ČÁSTEČNÝCH VÝBOJŮ	24
4.2	OBEČNÉ POŽADAVKY NA MĚŘÍCÍ SYSTÉM.....	26
4.3	STŘÍDAVÁ MĚŘENÍ.....	27
4.4	STEJNOSMĚRNÁ MĚŘENÍ	29
5.	EXTRAKCE CHARAKTERISTICKÝCH INFORMACÍ Z MĚŘENÍ.....	31
5.1	STATISTICKÉ PARAMETRY	31
5.2	PROSTŘEDKY ZPRACOVÁNÍ OBRAZU	32
5.3	FREKVENČNÍ A ČASOVÉ VLASTNOSTI (TF MAPPING)	35
5.4	WEIBULLOVO ROZDĚLENÍ NAD PHA KŘIVKAMI	36
5.5	CROSS WAVELET SPECTRUM.....	37

5.6	VYUŽITÍ PAMĚŤOVÉHO EFEKTU IZOLACE	39
5.7	HISTOGRAMY PRAVDĚPODOBNOTNÍHO ROZDĚLENÍ VELIKOSTI VÝBOJE	41
5.8	PRŮMĚRNÁ FREKVENCE VÝBOJE	41
6.	KLASIFIKACE.....	43
6.1	NEURONOVÉ SÍTĚ.....	43
6.2	KLASIFIKACE POMOCÍ FUZZY LOGIKY	48
6.3	KLASIFIKACE POMOCÍ VÝPOČTU VZDÁLENOSTÍ	48
6.4	STATISTICKÁ KLASIFIKACE	50
7.	VYUŽITÍ MOŽNOSTÍ KLASIFIKACE V LABORATOŘI	51
7.1	MOŽNOSTI PULSE-SHAPE ANALÝZY	51
7.2	PŘÍPRAVA DAT PRO NEURONOVOU SÍŤ	52
7.3	VYUŽITÍ NEURONOVÉ SÍTĚ PRO KLASIFIKACI.....	57
8.	ZÁVĚR.....	63
	SEZNAM LITERATURY A INFORMAČNÍCH ZDROJŮ	64
	PŘÍLOHY	1
	PŘÍLOHA 1 – NORMALIZOVANÉ PRŮBĚHY KAŽDÉ KATEGORIE	1
	PŘÍLOHA 2 – PRŮMĚRNÉ VEKTORY PARAMETRŮ PRO OBĚ POLARITY KAŽDÉ KATEGORIE	3
	PŘÍLOHA 3 - PREPAR.PY	4
	PŘÍLOHA 4 – READTRC.PY	8
	PŘÍLOHA 5 – NN.PY.....	13
	PŘÍLOHA 6 – PARAMS.PY	16
	PŘÍLOHA 7 – PLOT.PY	19

Úvod

Předkládaná práce se zabývá moderními metodami vyhodnocování elektrických měření částečných výbojů. Měření částečných výbojů jako takové nemá pevně stanoveny metody normou. Normou ČSN EN 60 270 jsou stanoveny především nutné požadavky, jaké musí měřicí systémy a jejich dílčí součásti splňovat, požadavky na kalibraci těchto systémů a jsou zde popsány obecné požadavky na provádění zkoušek. Velkou část zkoušek je nicméně nutné a vhodné upravit tak, aby odpovídala zvolené metodě vyhodnocování a jejím požadavkům.

Tato práce se zabývá převážně moderními metodami klasifikace částečných výbojů, které se staly dostupné až v posledních letech, díky narůstajícímu výpočetnímu výkonu a větší dostupnosti softwarových prostředků a algoritmů. Potřeba těchto metod vznikla z požadavků, které dnes vznikají v důsledku používání modernějších a ekologičtějších materiálů izolací, olejů, či v důsledku širšího využívání stejnosměrného přenosu elektrické energie.

Obecně se dá říci, že se tyto nové metody skládají ze dvou kroků. V prvním kroku se zjišťují charakteristické vlastnosti změřených průběhů, jejich vzájemná rozdílnost mezi jednotlivými typy výboje nebo variantami jednoho typu nebo i v různých geometrických uspořádáních, či s jiným typem izolačního materiálu.

Dalším krokem k úspěšnému posouzení a rozčlenění těchto průběhů jsou samotné klasifikační metody. Tyto metody musí předpokládat, že předkládaná vstupní data mají určitý stupeň variability. S touto variabilitou musí pracovat a na základě společných znaků musí porozumět sebemenším rozdílům v jejich uspořádání a tím správně rozhodnout mezi jednotlivými třídami, které byly na začátku měření uvažovány.

Některé z těchto klasifikačních metod lze s výhodou použít i na jiné než galvanické měření, jež je v této práci výhradně popisováno. Potom je možné uvažovat i online nasazení vyhodnocování.

Seznam symbolů a zkratek

UHF	Ultra high frequency / Velmi vysoká frekvence
TEV	Transient earth voltage / Přechodné zemní napětí
HFCT	High frequency current transformer / vysokofrekvenční proudový transformátor
DGA	Dissolved gas analysis
q	Zdánlivý náboj výboje
n	Četnost pulzů
N	Opakovací kmitočet pulzů
Φ_i	Fázový úhel pulzu
t_i	Čas mezi průchodem napájecího napětí nulou a pulzem výboje
I	Střední proud částečných výbojů
P	Výkon částečných výbojů
U_i	Zapalovací napětí č.v.
U_e	Zhášecí napětí č.v.
PRPD pattern	Phase resolver partial discharges pattern / fázová podoba měření č.v.
DWT	Discrete wavelet transform / diskrétní wavelet transformace
MSD	Multiscale decomposition / vícesložkový rozklad
w	Váhový koeficient neuronu, váhový vektor neuronu
a	Vstupní hodnota neuronu, vstupní vektor hodnot neuronu
b	Bias
σ	Aktivační funkce, směrodatná odchylka
a	Výstupní hodnota neuronu, výstupní hodnota vrstvy
δ	Chybový faktor
t_k	Požadovaný výstup
y_k	Skutečný výstup
Δw	Korekce
x_i	Řešený vektor
m_i	Průměrný vektor clusteru
s_i	Vektor jednotlivých samplů
M	Počet řešených tříd
N	Délka porovnávaných vektorů
L	Celkový počet samplů
d_i	Vzdálenost
CI	Interval 95% kvartilu

1. Úvod do problematiky částečných výbojů

Částečné výboje jsou lokalizované elektrické výboje, které přemost'ují izolaci mezi vodiči pouze částečně, či se mohou objevit v okolí vodiče. Z pohledu rozdělení poruchovosti izolačních systémů stojí částečné výboje za více než 60 % z celkového počtu poruch, což je řadí mezi jevy, které je důležité vyhodnocovat jak při kusových zkouškách ve výrobě energetických zařízení, tak i dlouhodobě online v provozu. Při poruše, která je zaviněna selháním izolace, se většinou musí zařízení nahradit kus za kus či provést komplexní opravy, což vede k výdajům v řádech milionů korun.

Mezi základní typy, na jaké lze částečné výboje rozdělit, patří koróna, plovoucí elektroda, bublina v izolaci a povrchový výboj. Typickým dlouhodobým projevem částečných výbojů je pak poškození izolačního systému, ve kterém k výbojové činnosti dochází, k opalování jeho povrchu a další postupné degradaci, což může časem vyústit v kompletní destrukci izolačního systému.

Částečné výboje se do svého okolí projevují jak světelně, tak tepelně, akusticky, chemicky a elektromagneticky. Z toho vyplývá základní rozdělení způsobů obecného vyhodnocování, které bude popsáno v kapitole 3. Cílem vyhodnocování je stanovit míru částečných výbojů, které se v zařízení vyskytují, případně i místo vzniku výboje, jelikož s touto informací lze zařízení opravit efektivněji a v kratším čase. Pomocí známé míry částečných výbojů a jejího vývoje v čase lze následně určit předpokládanou životnost stroje, na kterou, stejně jako na izolaci, mají také velmi velký negativní vliv vnější poruchy, jako například zkraty na zařízení, či přepětí.

2. Základní typy částečných výbojů a jejich charakteristiky

2.1 Plovoucí elektroda

Tento typ částečného výboje se vyskytuje v případě, že se v elektrickém poli vodičů nachází neuzemněný vodič. Za názorný příklad lze považovat neuzemněné stínění, špatné pospojení či zbytky kovových pilin. Ve vztahu k izolaci mohou být tyto vady bezpečné i nebezpečné. Záleží na konkrétním případě.

Tento typ výboje má silné vyzařování v UHF spektru a slabé akustické projevy. Také je lze vcelku dobře detekovat pomocí TEV a HFCT měření.

2.2 Bublina v izolaci

Při výrobě pevných izolací nebo při přetěžování olejových izolací může docházet k tvorbě bublin v izolaci. Překročením pevnosti plynu bubliny pak dochází k výbojům uvnitř této bubliny, což způsobuje další rozšiřování oblasti a tvorbu elektrických stromečků. Podle typu izolace může také docházet ke zvyšování vlhkosti plynu, což způsobí rychlejší degradaci.

Lze je dobře detekovat pomocí akustické, HFCT nebo UHF metody.

2.3 Povrchový výboj

Typicky vzniká na izolaci, která má znečištěný povrch prachem. Výboj je nebezpečný, neboť při něm dochází k vývinu tepla, což v kontaktu se izolačním materiálem může vést k požárům.

Je snadné je detekovat měřením v UHF, akusticky, TEV metodou i pomocí HFCT. Velmi dobře je možné je detekovat i okem, podle viditelně opáleného povrchu součásti.

2.4 Koróna

Korónový výboj se objevuje na ostrých hranách vodičů, které jsou umístěny ve vzduchu. Při výskytu v okolí pevných izolací může často přecházet v povrchový výboj. Změřené charakteristiky korónových výbojů se od ostatních typů velmi liší, proto je lze snadno rozeznat. Chování výboje je poté ovlivněno atmosférickou vlhkostí a teplotou.

Na venkovních zařízeních ji lze detekovat pomocí UV kamer a je dobře slyšitelná a detekovatelná pomocí UHF měření. [20]

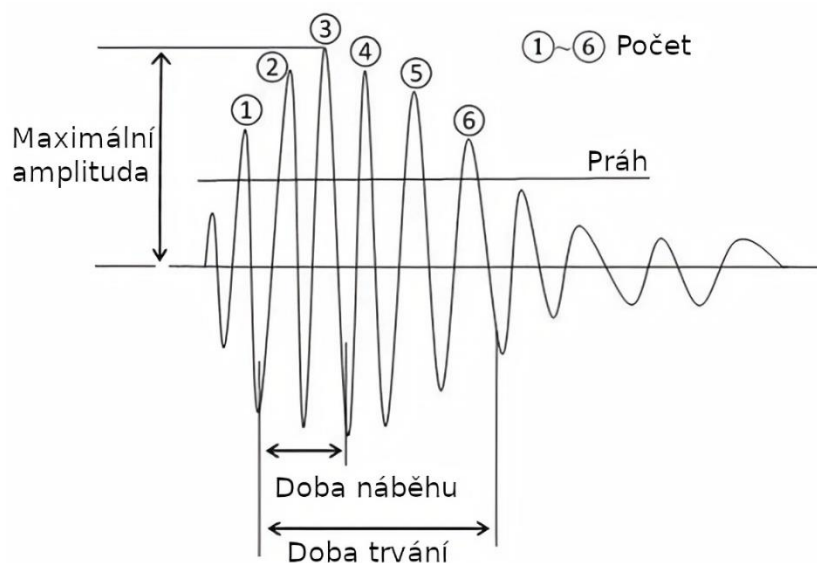
3. Základní rozdělení možností měření

V úvodní kapitole práce je naznačeno, jaké projevy lze u částečných výbojů očekávat. Následující metody se časem vyvinuly na vysokou úroveň, avšak základním pilířem vyhodnocování stále zůstává elektrické měření, které v posledních letech zažívá další období rozšiřování možností, jehož je dosaženo pomocí velkých výpočetních možností počítačů a dostupných pokročilejších metod softwarového zpracování. Z těchto důvodů je nejdříve věnována pozornost neelektrickým metodám měření a až za nimi se nachází galvanické měření, na něž tyto nové možnosti zpracování navazují.

3.1 Akustická detekce

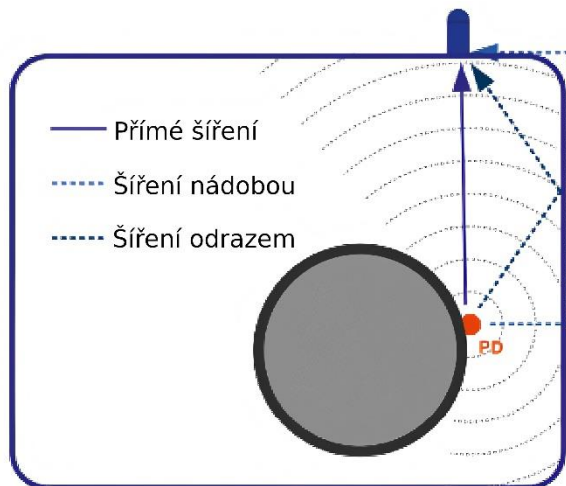
Při akustickém vyhodnocování se využívá jevu, kdy se při vytvoření výboje lokálně zvýší tlak, což vede ke zvýšení akustického tlaku na snímacích membránách. Do elektrické podoby je signál převeden buďto pomocí piezokrystalu či kapacitního mikrofonu a následně je zesílen odpovídajícím zesilovačem, přenesen přes vodiče a digitalizován pro další zpracování. V základu je touto metodou možno měřit až do frekvence cca 300 KHz, která se typicky vyskytuje u zařízení s olejem nebo SF6. Výboje v dutině mohou typicky generovat akustické signály v rozsahu několika Hz až do několika MHz s tím, že než ve slyšitelném pásmu je mnohem více energie vyzářeno v pásmu ultrazvukovém.

Při vyhodnocování změřeného signálu, který je zobrazen na obrázku 1, je třeba uvažovat, že jsou na signál superponovány akustické signály z jiných zdrojů, například z vibrací mechanických částí, a je třeba také počítat s útlumem signálu, jak v materiálu, tak při přestupu do senzoru, a odrazy uvnitř zařízení.



Obrázek 1: Znázornění veličin akustického signálu [25]

Pokud jde o samotnou prvotní ultrazvukovou vlnu, je možné předpokládat její přímočaré šíření. Tato vlna ale bývá velmi tlumena při průchodu izolantem. Nejlépe se šíří kovem, díky čemuž by teoreticky neměl být problém vlnu detekovat na kterémkoliv místě nádoby. Tento jev je však omezen nehomogenním prostředím vnitřku stroje, například



Obrázek 2: Šíření vlny ve stroji [21]

olejovou izolací. Ve skutečnosti, v případě výkonového transformátoru, senzor detekuje zdrojovou vlnu třikrát. Tato tři opakování jsou ovlivněna různou trasou šíření, kdy jednou je vlna detekována při přímé cestě, poté odrazem od nádoby a také v momentě, kdy se šíří čistě nádobou. Výsledná podoba signálu závisí na geometrii uspořádání a na rychlostech šíření v daných materiálech

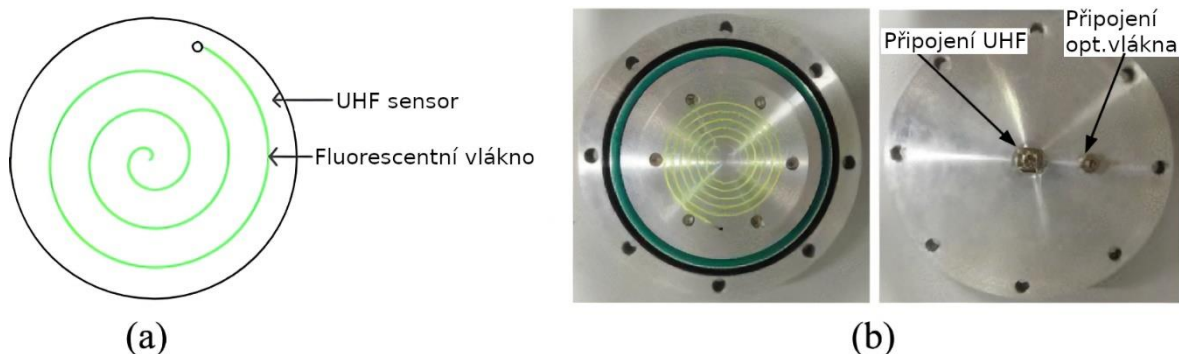
Ze signálu lze následně dopočítat jeho amplitudu, energii, počet opakování,

dobu náběhu a dobu trvání. Z těchto informací lze získat alespoň přibližné informace o typu výboje.

Akustická metoda je nicméně velmi výhodná, pokud je potřeba částečný výboj lokalizovat. Při vhodném uspořádání měřicích senzorů lze dopočítat polohu za předpokladu, že známe rychlost šíření zvuku v materiálu izolace a nádoby či krytu přístroje a geometrické uspořádání senzorů, ze zpoždění mezi jednotlivými signály. Standardně se pak lokalizace doplňuje o UHF nebo elektrické měření, pomocí kterého se stanoví okamžik vzniku výboje. Metoda je také vhodná pro online měření v místech, které jsou zarušeny elektromagnetickým polem, neboť nedochází k jejich vzájemné interakci. [21,22,25]

3.2 Optická detekce

Při optickém měření se využívá toho, že vznikající výboje vyzařují fotony jak v IR, viditelném, tak i UV spektru. Výsledné vyzařované spektrum je poté ovlivňováno teplotou a tlakem. Tento způsob snímání lze dále rozdělit podle toho, jaké lokace částečných výbojů chceme zjišťovat, na prostředí venkovní a vnitřní. Ve venkovním prostředí lze pro snímání použít kamery pracující v oblasti viditelného a UV spektra, kde slouží primárně pro detekci korony na vedení a izolátorech. Pro části vnitřní lze použít fluorescentní optické vlákno,



Obrázek 3: Ukázka kombinovaného optického a UHF senzoru [23]

kteří je schopno zachytit dopadající světelné záření z jakéhokoliv směru, a je proto vhodné do míst, ve kterých neznáme pozici světelného zdroje, a přivést jej na světlocitlivý senzor.

Pro samotnou detekci signálu se používají fotonásobiče. Fotonásobiče se skládají ze skleněné baňky a soustavy elektrod. Prvotní impulz je založen na principu fotoelektrického jevu, kdy do systému je dodána fotonem energie, která vyrazí z fotokatody elektron. Elektron postupuje přes soustavu elektrod, které mají potenciál rovnoměrně odstupňován a mají činitel sekundární emise > 1 . Tímto dochází k postupnému násobení proudu elektronů, přičemž typický zisk celého fotonásobiče se pohybuje v hodnotách mezi 10^5 až 10^7 . [23,24]

3.3 UHF detekce

Měření na UHF (ultra krátkých vlnách) se používala prvotně u plynem izolovaných systémů, jako jsou například SF₆ vypínače a podobné. Postupem času se začala používat i na výkonových transformátorech.

Mezi hlavní výhody této metody patří její aplikovatelnost na online měřeních, vyšší odolnost proti rušení oproti měření podle IEC 60270 a možnost relativně dobře lokalizovat zdroj rušení. Společně s akustickou metodou je tato metoda zařazena do normy IEC TS 62478, kde jsou, mimo možností použití těchto metod, popsány hlavně způsoby korelace těchto metod na metodu galvanickou. Využívá se zde procesu kalibrace podle IEC 60270, které zajistí známou hodnotu velikosti náboje přivedeného na svorky zkušence, pro určení míry korelace mezi galvanickou a UHF a akustickou metodou. Tuto korelaci nelze považovat vždy za správnou, neboť v reálném nasazení záleží také na relativních pozicích snímačů vůči zdroji rušení, ale je možné s těmito údaji o vzájemné korelaci do určité míry pracovat.

Pokud jde o výkonové transformátory, jakožto jedno z nejdůležitějších zařízení, na kterých je nutnost kontinuálního online měření, nachází se na těchto strojích typicky 3 až 4 okna pro instalaci UHF senzorů. U starších typů transformátorů, které nejsou těmito okny

vybaveny, se pak přistupuje buď k jejich vytvoření v rámci revize, nebo instalaci antény do výpustního otvoru transformátoru.

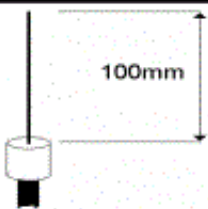
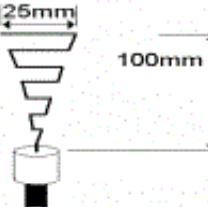
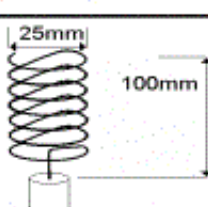
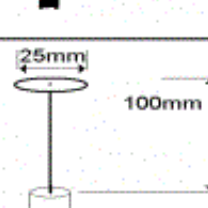
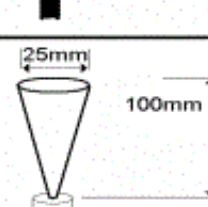
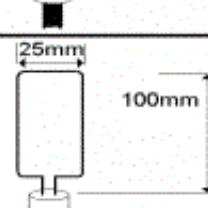
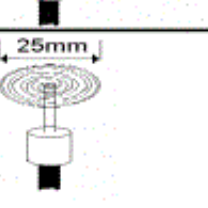


Obrázek 4: Okno transformátorové nádoby s nainstalovaným UHF senzorem [28]

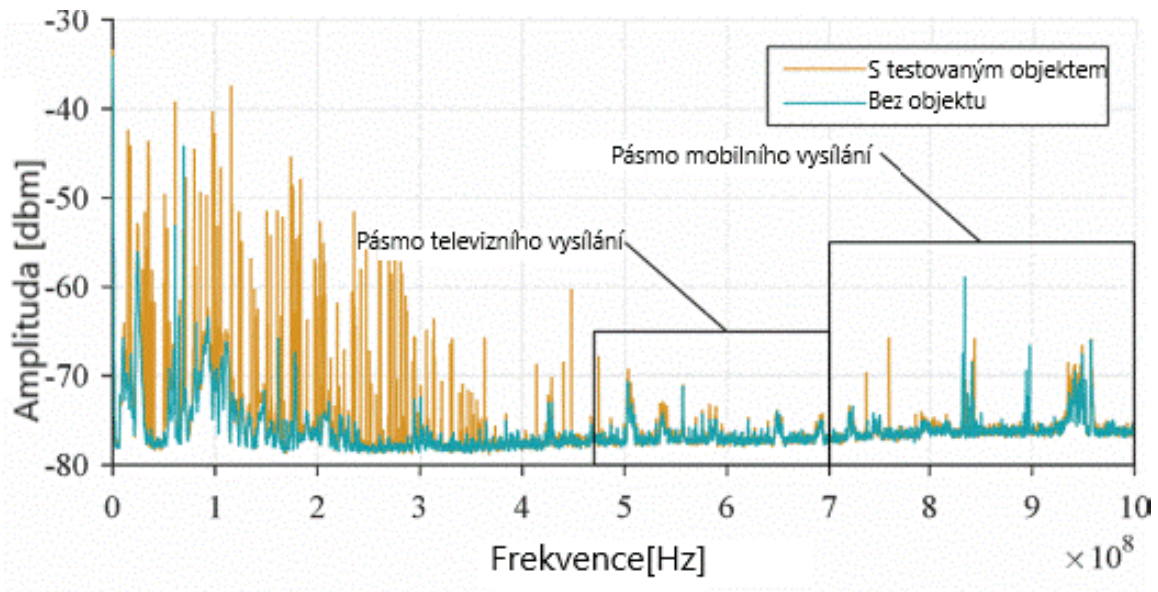
Typy antén, které lze v praxi používat, pak byly v [26] změřeny a byly slovně ohodnoceny na obrázku 5. UHF senzory poté pracují jako širokopásmové snímače v rozsahu 300 MHz až 3 GHz.

Při měření je dobré si všimnout vlastních rezonančních frekvencí nádob transformátorů, které se pro výkonové transformátory pohybují v řádech desítek MHz. Také je dobré věnovat zvýšenou pozornost, zvláště při použití vnějších senzorů mimo nádobu, pásmu využívaném pro mobilní telefony, které se pohybuje v mezích 876–915 MHz a 921–960 MHz, a také pásmu pozemního televizního vysílání v rozsahu 470–694 MHz. V ostatních pásmech se nevysílá v takovém výkonu a jejich vymezení lze nalézt na stránkách ČTÚ.

Výsledné frekvenční spektrum, které je zobrazeno na obrázku 6, je těmito spektry taktéž ovlivněno a jejich vymezení je zde znázorněno. Podle rozložení výsledného spektra a jím vyzařované energie je taktéž možno, na základě majoritních frekvencí, určit typ částečného výboje, neboť mají jiné vyzařovací frekvence a rozdílný průběh integrální křivky energie přes frekvenci. [26, 27, 28,29,41]

Typ antény	Provedení	Výsledky
Rovný drát Monopól		Dobrý frekvenční rozsah mezi monopóly. Dobré chování vertikálně i horizontálně ke zdroji.
Lichoběžníkový Monopól		Podobné chování jako rovný drát. V oblastech rezonanční frekvence mírně lepší zisk.
Spirála Monopól		Menší zisk než většina ostatních typů. Pracuje dobře v horizontální poloze ke zdroji.
Disk Monopól		Podobné charakteristiky jako u rovného drátu. Žádné zlepšení v možnostech natočení.
Kónický Monopól		Lepší odezva na širokopásmový signál. Zisk mírně vyšší, než rovný drát.
Smyčka Anténa		Menší zisk než ostatní typy.
Spirála Anténa		Malý zisk. Průměr spirály příliš malý k získání výhod u širokopásmového příjmu.

Obrázek 5: Typy antén a jejich vlastnosti [26]



Obrázek 6: Měření objektu a radiového pozadí pomocí externí UHF antény [41]

3.4 Měření pomocí TEV metody

Měření pomocí TEV (Transient Earth Voltage) probíhá za pomoci kapacitní sondy ve frekvenčním rozsahu 3 MHz až 100 MHz v okolí zařízení. Díky nedokonalému stínění a konečné impedanci skříně proti zemi se tato skřín pohybuje na potenciálu několika milivoltů proti zemi. Doba, po kterou tento stav trvá, je v řádu nanosekund.

Pomocí této metody je téměř nemožné odhalit místo zdroje částečných výbojů. Taktéž nejdou touto metodou odhalit některé typy částečných výbojů, neboť jejich



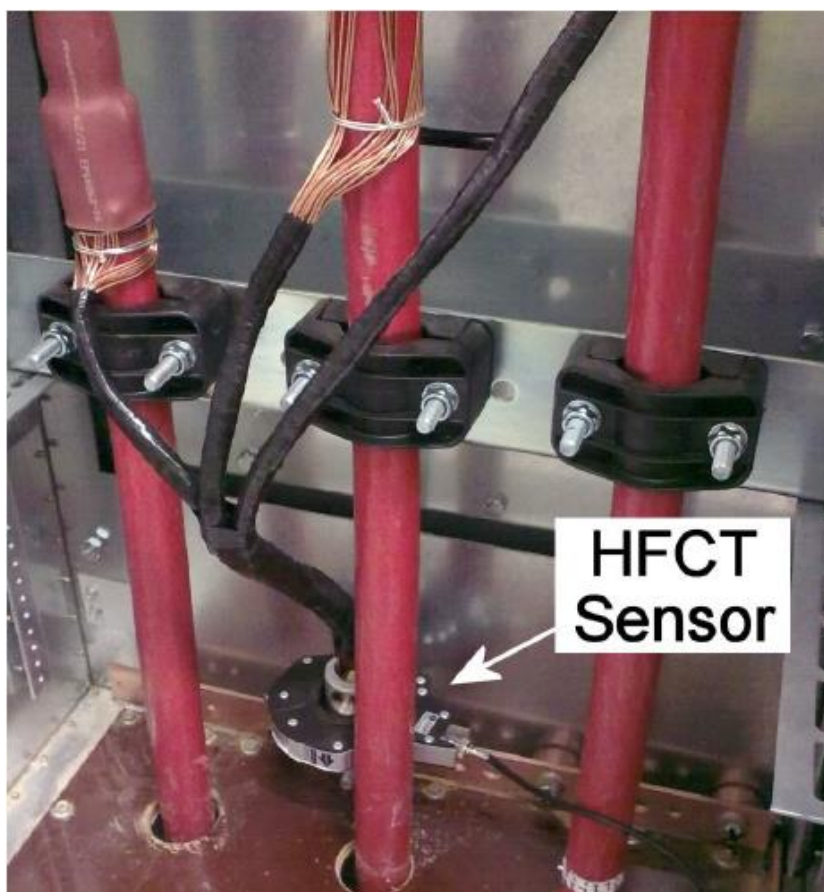
Obrázek 7: Přenosná sonda UltraTEV [31]

vyzařovaná energie je příliš malá. Typicky se jedná o bubliny v izolaci. Dále je metoda dosti omezená použitou izolací, neboť při použití olejové izolace není TEV emise detekovatelná. TEV metoda se tedy používá výhradně jako doplňková například k měření pomocí UHF či HFCT. [20,30]

3.5 HFCT

HFCT (High Frequency Current Transformer) měření je, naopak od TEV metody, vhodné pro použití v širokém spektru zařízení. Typicky se HFCT umisťují na zemní vodič u transformátorů, kabelů, skupin zařízení či rozvodny. Měření probíhá v rozsahu 0,5 – 50 MHz. HFCT, jakožto snímače, jsou poté proudové transformátory, které jsou v provedení s děleným jádrem, aby bylo možné je snadno instalovat okolo vodičů.

Výhoda použití HFCT je poté v jejich instalaci na kabelu. Oproti jiným energetickým prvkům lze totiž na kabelech měřit i zpětnou složku proudu a pomocí časové diference této a dopředné složky je možné, za předpokladu známé rychlosti šíření vlny po vedení, spočítat vzdálenost, v jaké se nachází zdroj signálu. V závislosti na stínění vodiče je možné detekovat částečné výboje až do vzdálenosti jednoho kilometru. [20]



Obrázek 8: Příklad instalace HFCT na stínící vodiče svazku [32]

3.6 DGA metody

DGA metody jsou způsoby analýzy, které se zabývají látkovým složením izolace a produktů, které vznikají při výbojové činnosti či tepelným namáháním.

Využívají se výhradně u zařízení, u kterých je izolační systém tvořen olejem. Obecně lze DGA metody rozdělit na dvě skupiny: metody, které využívají poměrů sloučenin plynů, a metody, jež pro analýzu využívají absolutní vyjádření množství plynu.

3.6.1 Metoda Rogersova poměru

Metoda Rogersova poměru spadá do první kategorie výše zmíněného rozdělení. Využívá tedy poměru zastoupení jednotlivých sloučenin v objemu a přiřazených kódů jednotlivým poměrům. Mezi sledované sloučeniny se řadí methan (CH_4), ethan (C_2H_6), ethylen (C_2H_4), acetylen (C_2H_2) a vodík.

Složka	Kód poměru
CH_4/H_2	i
$\text{C}_2\text{H}_6/\text{CH}_4$	j
$\text{C}_2\text{H}_4/\text{C}_2\text{H}_6$	k
$\text{C}_2\text{H}_2/\text{C}_2\text{H}_4$	l

Tabulka 1: Kódy přiřazené poměrům sloučenin

Na základě vytvořeného kódu lze pak dané charakteristice poruchy přiřadit její konkretizaci. Označení týkající se částečných výbojů jsou poté uvedeny v tabulce 3.

Kód poměru	Rozsah	Kód
i	≤ 0.1	5
	$> 0.1, < 1.0$	0
	$\geq 1.0, < 3.0$	1
	≥ 3.0	2
j	< 1.0	0
	≥ 1.0	1
k	< 1.0	0
	$\geq 1.0, < 3.0$	1
	≥ 3.0	2
l	< 0.5	0
	$\geq 0.5, < 3.0$	1
	≥ 3.0	2

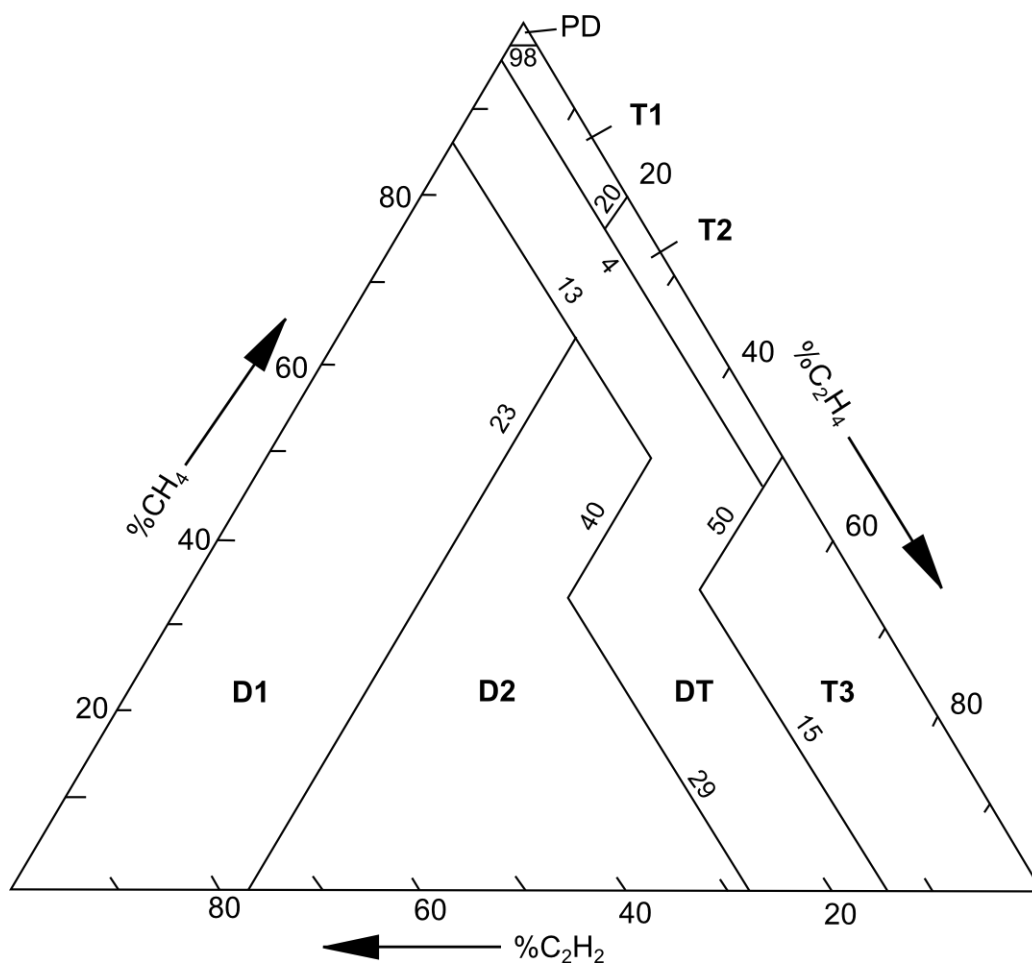
Tabulka 2: Přiřazení kódových označení konkrétním poměrům sloučenin

i	j	k	l	Porucha
5	0	0	0	Částečné výboje
5	0	0	1-2	Částečné výboje s CO stopou

Tabulka 3: Typy poruch vztahující se k částečným výbojům

3.6.2 Duvalův trojúhelník

Metoda Duvalova trojúhelníku je příkladem metody je naopak dobrým příkladem metod druhé skupiny. Pracuje s absolutním množstvím plynů, které dává do poměru s celkovým objemem. Na základě procentuálního zastoupení methanu, ethylenu a acetyleny stanovuje nejpravděpodobnější příčinu jejich vzniku v zařízení. Na obrázku 9 je Duvalův trojúhelník vyobrazen.



Obrázek 9: Duvalův trojúhelník [33]

Popisky trojúhelníku odpovídají následujícímu přiřazení:

PD – částečné výboje,

T1 – Tepelná porucha s teplotou pod 300 °C,

T2 – Tepelná porucha s teplotou mezi 300 °C a 700 °C,

T3 – Tepelná porucha s teplotou nad 700 °C,

D1 – Výbojová činnost o nízké energii – jiskření,

D2 – Výbojová činnost o vysoké energii – elektrické oblouky,

DT – Společné působení elektrických i tepelných poruch.

Z trojúhelníku je dobré si povšimnout, že při výskytu částečných výbojů dochází k vývinu téměř výhradně methanu, zatímco ostatní plyny jsou zastoupeny pouze minoritně. [33]

4. Možnosti galvanického měření

Galvanické měření má dlouhou historii. Jeho standardizace je uvedena v normě ČSN EN 60270. Oproti výše zmíněným metodám má tato metoda výhodu v tom, že lze téměř přesně měřit náboj, který se ve výboji přenesl. Mezi hlavní nevýhody patří nemožnost online měření, což může u již používaných zařízení vést k finančním ztrátám z nutnosti plánovaného výpadku či náhradního provozu.

Pro účely reprodukovatelnosti a možností porovnání byly normou definovány parametry, jež jsou popsány v následující části.

4.1 Obecné parametry určované při měření částečných výbojů

Impulz částečného výboje

Napěťový, nebo proudový impulz zaznamenaný na zkoušeném objektu. Tento impulz je zaznamenaný měřicí soustavou a měřicím přístrojem převeden na *obraz impulzu částečného výboje*.

Zdánlivý náboj q impulzu částečného výboje

Náboj, který po přiložení na svorky zkoušeného objektu ve velmi krátkém čase vyvolá na měřicím přístroji stejnou výchylku jako proudový impulz částečného výboje. Nejedná se ale o náboj uvolněný při samotném výboji. Zdánlivý náboj se obvykle vyjadřuje v pC.

Četnost impulzů n

Pro pulzy v úrovni předepsaného rozsahu jde o celkový počet pulzů vydělených dobou jejich měření.

Opakovací kmitočet impulzů N

Pro periodicky se opakující výboje se jedná o počet odpovídajících stejných pulzů za sekundu.

Fázový úhel Φ_i a čas t_i výskytu impulzu

Čas t_i je měřen mezi průchodem zkušebního napětí nulou a výskytem impulzu částečného výboje. Úhel Φ_i poté odpovídá rozdílu fázového posunu pro daný čas t_i a periodě T vstupního napětí.

Střední proud částečných výbojů I

Jedná se o ekvivalentní hodnotu proudu, který projde při přemístování jednotlivých zdánlivých nábojů během časového úseku za ten samý časový úsek. Standardně se vyjadřuje v coulombech za sekundu, potažmo ampérech.

$$I = \frac{1}{T_{REF}} \cdot (|q_1| + |q_2| + \dots + |q_n|) \quad (1)$$

Výkon částečných výbojů P

Představuje výkon dodaný zkoušenému objektu vyvolaný přesunem zdánlivého náboje q_i za časový interval T_{REF} .

$$P = \frac{1}{T_{REF}} \cdot (q_1 \cdot u_1 + q_2 \cdot u_2 + \dots + q_n \cdot u_n) \quad (2)$$

Napětí v uvedeném vzorci jsou okamžité hodnoty napětí v časech výskytů jednotlivých impulzů výbojů. Při sledování je nutné dbát na polaritu těchto.

Nejvyšší opakovaně se vyskytující úroveň výbojů

Je hodnota, změřená měřicím systémem, která odpovídá maximální vstupní odezvě měřicího přístroje. Měřicí přístroj musí mít odezvu takovou, že pro velké impulzy se stejným opakovacím kmitočtem N přepočte vstupní záznamy podle uvedené tabulky. Předpokladem je nastavený rozsah a zisk tak, aby odečet byl na plném rozsahu, nebo 100 % pro $N = 100$.

N (1/s)	1	2	5	10	50	≥ 100
R_{MIN}	35	55	76	85	94	95
R_{MAX}	45	65	86	95	104	105

Počáteční napětí částečných výbojů U_i

Počáteční napětí, při kterém jsou na měřicím systému detekovány prvotní pulzy. Předpokladem je postupné navyšování hodnoty napětí z hodnoty, kdy nejsou pozorovány žádné částečné výboje.

Při stejnosměrných zkouškách je obtížné stanovit toto napětí, neboť výskyt částečných výbojů je náchylný na změnu rychlosti napětí v době jejího přiložení. Je proto nutné napětí zvyšovat s dostatečně pomalou změnou.

Zhášecí napětí částečných výbojů U_e

Nejnižší napětí, při kterém jsou na měřicím systému detekovány pulzy částečných výbojů. Předpokladem je postupné snižování hodnoty napětí z hodnoty, kdy ještě částečné

výboje byly pozorovány.

Při stejnosměrných zkouškách je nutné zkušební napětí snižovat dostatečně z důvodů uvedených u *Počátečního napětí částečných výbojů* U_i .

Dolní a horní mezní kmitočety f_1 a f_2

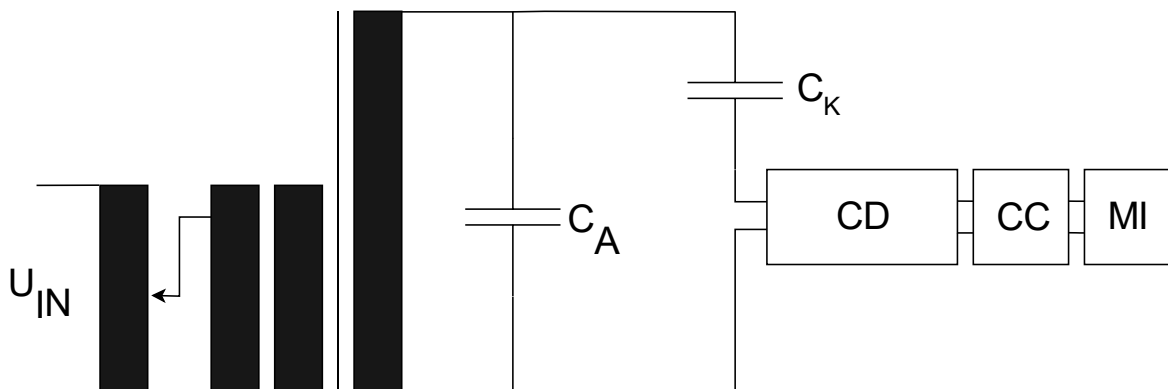
Frekvence, při které má měřicí obvod útlum 6 dB od maximální možné hodnoty.

4.2 Obecné požadavky na měřicí systém

Následující část se bude zabývat převážně elektrickým měřením, které z normy ČSN EN 60270 vychází. Zkušební obvod pro měření výbojové činnosti musí být kalibrován podle kapitoly 5 a vyhovovat požadavkům kapitoly 7 této normy. Obecně se uvažují tři způsoby zapojení měřicí impedance: v sérii s vazební kapacitou (obrázek 10), v sérii s měřeným objektem a můstkové zapojení. V případě zvláštních zkušebních objektů lze použít i jiný zkušební obvod a je doporučeno, aby výstupem měření byl zdánlivý náboj, kdykoliv je to možné. Pro daný měřicí systém poté musí být zaznamenán horní a dolní mezní kmitočety a délka nejkratšího časového intervalu mezi dvěma po sobě jdoucími stejnými pulzy, jejichž amplituda není rozdílná o více než 10 %. U stejnosměrného měření je nutné navíc použít čítač pulzů, neboť jednotlivé výboje nelze přiřadit konkrétnímu fázovému úhlu Φ_i .

Obecné schéma pro měření částečných výbojů se poté skládá především z:

- Zkoušeného objektu
- Vazebního kondenzátoru, v provedení s nízkou indukčností, nebo druhého zkoušeného objektu podobného zkoušenci, který vykazuje podstatně menší výbojovou činnost při stejném napětí. Stejná, nebo vyšší hodnota je přípustná, pokud lze zdroje pulzů odlišit.
- Samotného měřicího přístroje
- Nízko šumového vysokonapěťového zdroje
- Nízko šumových vodičů

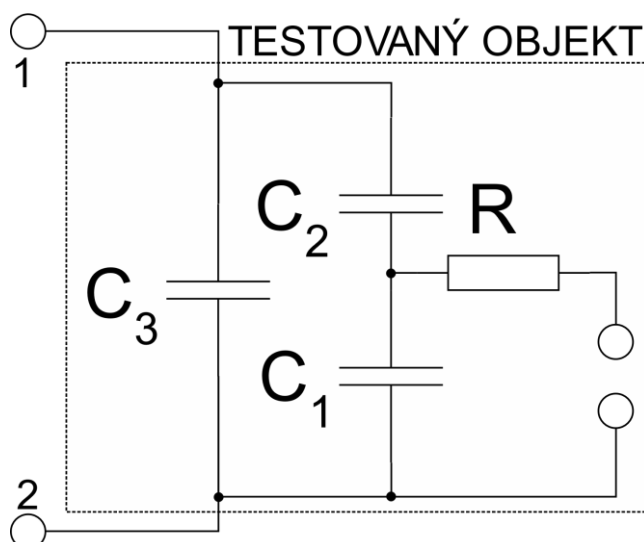


Obrázek 10: Základní schéma pro měření [3]

Pro zaručení správnosti je nutné mít dobře známé přenosové impedance každé dílčí součásti a mít celý systém zkalibrovaný odpovídajícím kalibrátorem. Velký vliv na výsledek měření mají především měřicí impedance, kde se proudový impulz převádí na napěťový impulz, dále přívodní vodiče a osciloskop samotný. Další, avšak již druhořadý, vliv na výsledek měření může mít také dlouhodobé vyhodnocování měření pomocí PC, díky kterému lze vyhodnocovat nejen již proběhlé degradace izolačního systému, ale zároveň určovat trend, s jakým izolace mění své vlastnosti v delším časovém horizontu. [2,3]

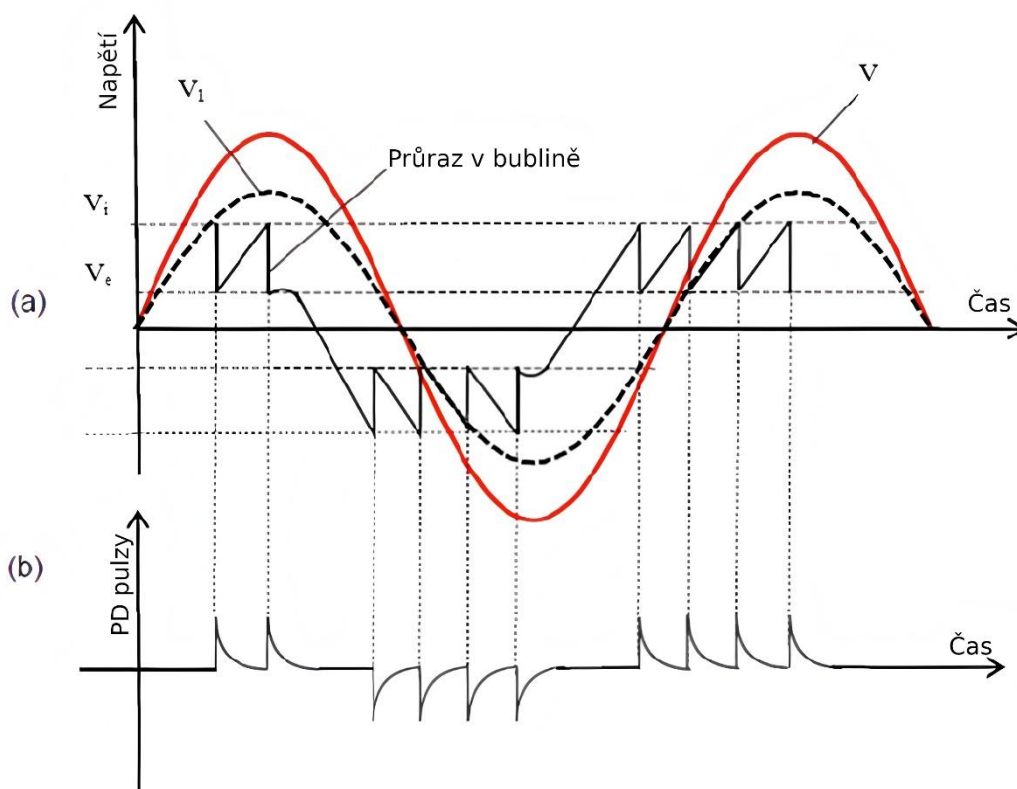
4.3 Střídavá měření

Jak pro AC, tak DC měření lze vcelku spolehlivě změřit časové průběhy pulzů částečných výbojů. U měření při střídavém napětí lze navíc pozorovat jejich pravidelný výskyt opakující se s konkrétním fázovým posunem vůči základní harmonické složce. To je způsobeno tím, že tyto výboje jsou citlivé nejen na velikost přiloženého napětí, ale i na jeho derivaci. Jako příklad takového chování je zde uveden model dutiny v izolaci, na kterém je toto chování názorně zobrazeno.



Obrázek 12: Náhradní schéma pro částečný výboj v dutině [6]

V obrázku č.11 je znázorněno náhradní schéma izolace s plynovou dutinou. Dutina je reprezentována kombinací prvků C1, R a jiskřiště. Kapacity C2 a C3 poté vyjadřují ostatní parazitní kapacity izolace. Průběh napětí a odpovídající pulzy lze pozorovat z následujícího obrázku.



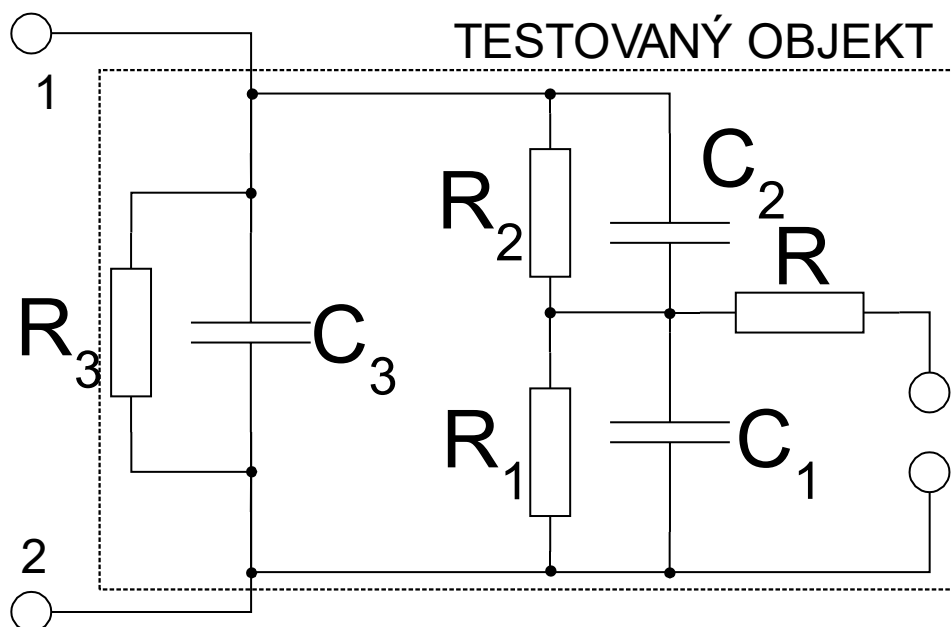
Obrázek 11: Průběhy výboje v dutině. a) Napětí na zkušenci a v dutině b) Napět'ové pulzy na měřící impedanci [5]

Výsledky AC měření lze ukládat v časové, nebo fázové podobě. Pro převedení do fázové podoby je nutné data z časové oblasti rozdělit na určité množství časových intervalů v závislosti na posunu a odpovídající změřené pulzy seskupit. Tímto vznikne fázová podoba, která se obvykle označuje jako PRPD pattern, obsahující informace o fázovém posunu Φ_i každého výboje od napětí zdroje, zdánlivý náboj q a četnost výbojů n . [4,5,6]

4.4 Stejnoseměrná měření

Pro měření na stejnosměrném napětí nelze použít stejné náhradní schéma dutiny, jako je uvedeno na obrázku 11. Pro popis chování na stejnosměrném napětí je nutné toto schéma doplnit o parazitní vodivosti každé části izolace tak, jako je tomu uvedeno na obr. č. 13.

Oproti výbojům při AC napětí, které se opakují s periodou 50 Hz, dochází k výbojům

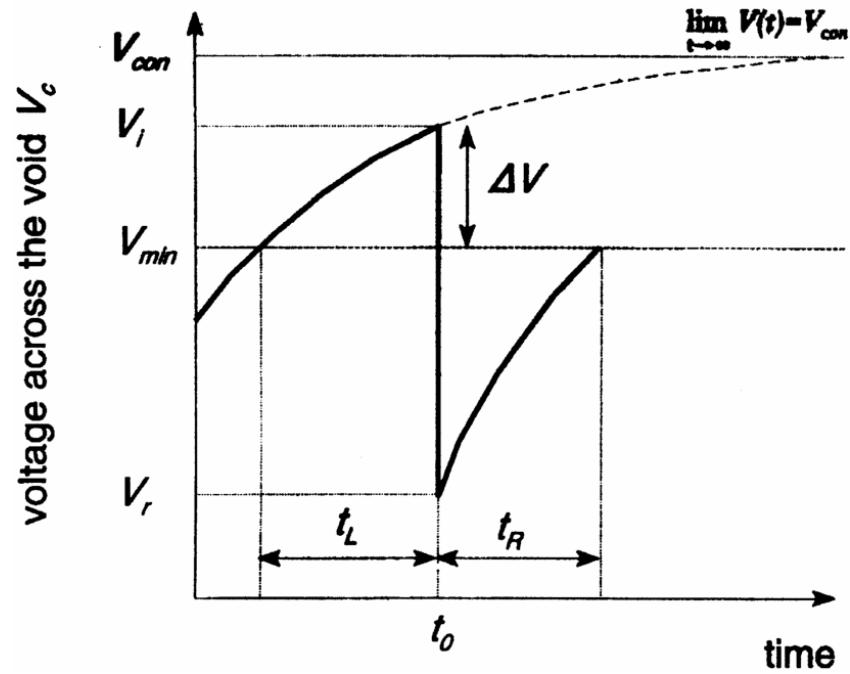


Obrázek 13: Náhradní schéma pro částečný výboj v dutině – DC [6]

na DC napětí s různým časovým rozdílem. Tento rozdíl je způsoben časovou konstantou dutiny a nutností mít vždy v prostoru dutiny startovací elektron.

Během prvního výboje nemusí být tento startovací elektron přítomen, což se projeví zvýšeným napětím na dutině před prvním výbojem. Rozdíl těchto napětí je v obrázku 14 značen jako ΔV . Po konci výboje ustane napětí na hodnotě V_R a při dalších výbojích nastává výboj již při $V_{MIN}(U_I)$, za předpokladu, že napětí naroste na tuto hodnotu rychleji, než dojde k nedostatku elektronů v dutině.

Četnost výbojů n je závislá nejen na velikosti přiloženého napětí, ale také na polarizaci izolace, která četnost výrazně zvyšuje. Polarizace může také způsobit výbojovou činnost opačné polarit v okamžiku odpojení objektu od zdroje. [4,5,6]



Obrázek 14: Průběh napětí v dutině při DC namáhání [6]

5. Extrakce charakteristických informací z měření

Pro správnou kvantifikaci výboje je nutné o výbojové činnosti získat co nejvíce možných informací. Úkolem je redukovat data změřených PRPD patternů či pulzů, které jsou tímto procesem zredukovány o spoustu dat, a charakterizovat je pomocí několika různých směřodatných veličin. Tyto veličiny mohou být poté snadněji zpracovány a mají vypovídající hodnotu o změřených průbězích.

Tato kapitola je rozdělena na dvě části, z nichž se první zabývá převážně AC měřeními a druhá DC měřeními.

5.1 Statistické parametry

Za základní statistické parametry se považují následující veličiny: průměr, koeficient šikmosti, koeficient špičatosti, rozptyl a vzájemná korelace.

Průměrem lze získat rozdělení velikosti výboje na fázovém úhlu ($H_{qn}(\varphi)$) a rozdělení četnosti náboje ($H_n(\varphi)$). Obě rozdělení lze pak rozdělit podle fázového úhlu na parametry kladné půlvalny napětí a záporné půlvalny napětí.

Koeficient šikmosti poté charakterizuje tvar rozdělení oproti normálnímu rozdělení. Pokud je rozdělení hodnot nakloněno více vlevo od maxima normálního rozdělení, je šikmost záporná a naopak.

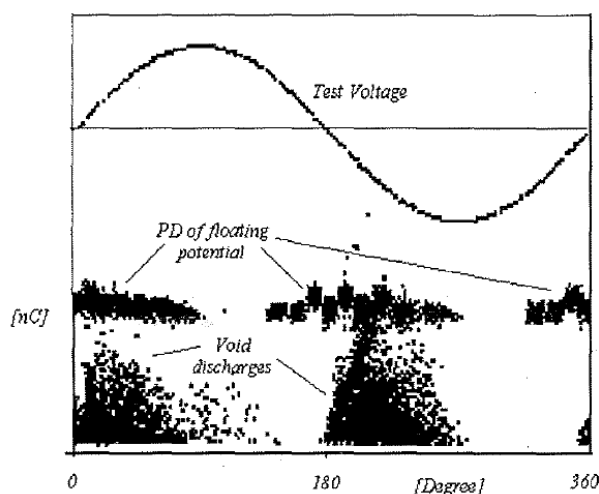
Koeficient špičatosti udává tvar průběhu rozdělení taktéž normálnímu rozdělení. Normální rozdělení má špičatost rovnou nule a kladná špičatost značí, že veličiny jsou více rozmístěny v blízkosti maxima, a naopak záporná špičatost symbolizuje rozdělení, kde jsou hodnoty rozděleny rovnoměrněji a křivka je více plochá.

Rozptyl vyjadřuje, jak jsou jednotlivé změřené body rozptýleny vůči sobě. V případě nulového rozptylu jsou pak hodnoty identické.

Koeficient vzájemné korelace se poté využívá při určování podobnosti $H_{qn}^+(\varphi)$ a $H_{qn}^-(\varphi)$, čili podobnosti rozdělení při kladné a záporné půlperiodě. Nulový koeficient znamená kompletní asymetrii a koeficient s hodnotou rovno jedné poté znamená absolutní symetrii hodnot, což značí, že rozdělení není závislé na polaritě napětí. [1]

5.2 Prostředky zpracování obrazu

Následující postup, který využívá zpracování obrazu PRPD, je vhodný zejména, pokud při měření působí vícero zdrojů výbojové činnosti najednou. Na obrázku 15 jsou poté zachyceny změřené vzorky na synchronním stroji. Z obrázku je patrné, že k výbojové činnosti dochází v důsledku dvou různých zdrojů. V tomto případě je rozlišení patrné i vizuálně. V případě překrývání je ale nutné přistoupit k pokročilejším metodám a data rozdělit, neboť by následná klasifikace vykazovala vysokou chybovost. Pro další analýzu se proto data převádí na 2D obraz, který je dále zpracováván. [7]

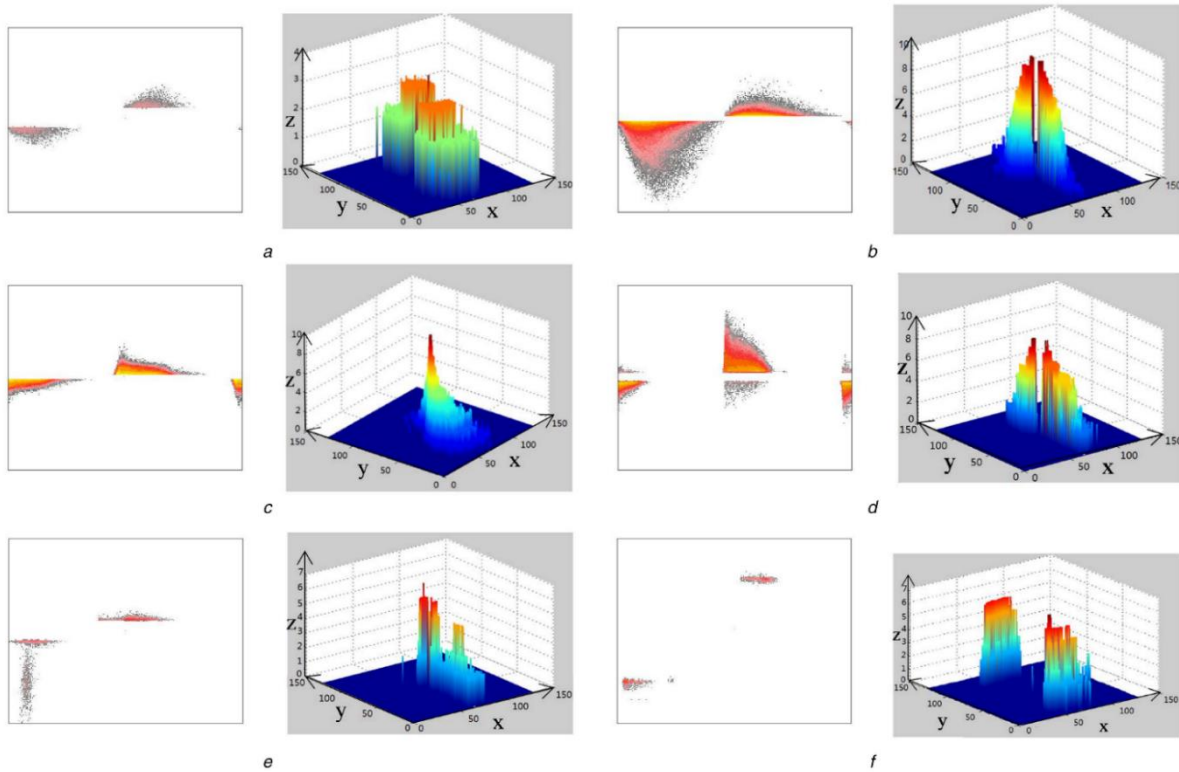


Obrázek 15: PRPD při více zdrojích výbojové činnosti [7]

Typické charakteristické obrazy je možné rozpoznat podle obrázku 16, na kterém jsou zobrazeny změřené PRPD obrazy a jejich převedení na PPD o rozměrech 128×128, kdy osy X a Y odpovídají následující transformaci a osa Z vyjadřuje počet výskytů:

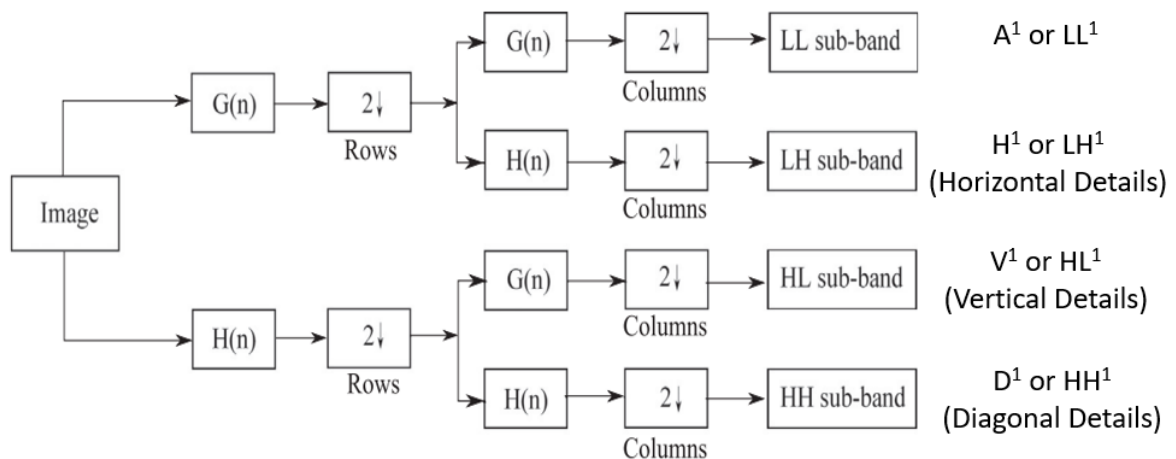
$$\begin{cases} x = \text{fix}(r \cdot \cos \Phi_i) + c \\ y = \text{fix}(r \cdot \sin \Phi_i) + c \\ z = n \end{cases}$$

Velikost fázoru r poté reprezentuje normalizovanou velikost pulzu výboje. Parametr c slouží pro posun obrazce tak, aby byly hodnoty x a y větší, nebo rovny nule. Funkce fix slouží pro vrácení celého čísla. [8]



Obrázek 17:PRPD a PPD obrazy na synchronním stroji a s následujícími typy částečných výbojů: a) výboj v dutině, b) vnitřní delaminace (bublina pod elektrodou), c) delaminace mezi elektrodou a izolací, d) výboj v drážce, e) povrchový výboj, f) výboj v mezeře [8]

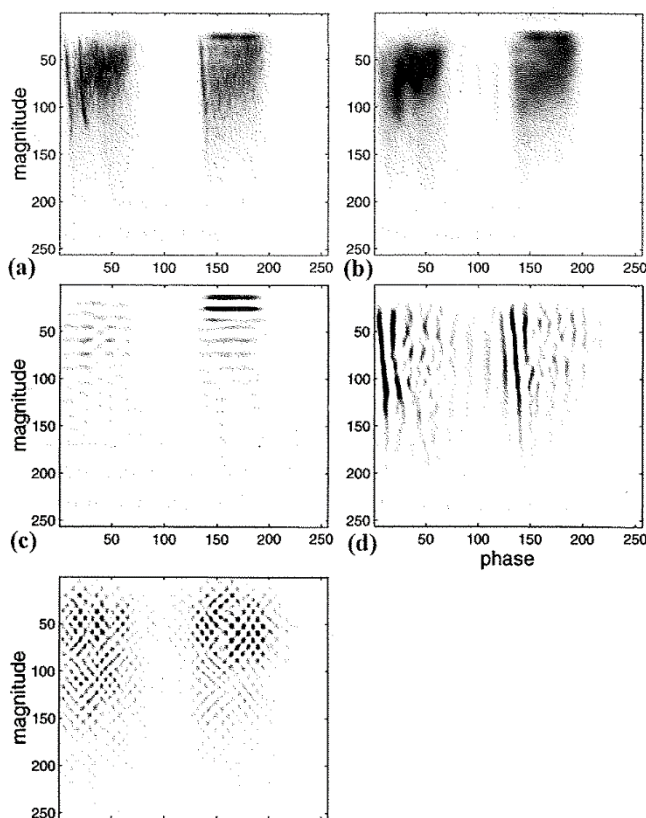
Pro analýzu těchto PPD obrazů lze využít diskrétní wavelet transformaci (DWT). V této transformaci se obraz rozdělí na řádky a k hodnotám se poté přistupuje jako k datům vztaženým k jedné proměnné. Tato data poté prochází filtry typu dolní propusti, ze které vzniknou přibližné koeficienty, a horní propusti, ze které se získají koeficienty detailní. Před získáním koeficientů je nutné snížit vzorkovací frekvenci na polovinu. Další filtrací poté



Obrázek 16: Jednoúrovňové schéma DWT transformace [9]

vznikne sada přibližných, horizontálních, vertikálních a diagonálních koeficientů, ze kterých lze zpětně vytvořit obraz každé složky. Základní blokové schéma wavelet transformace pro zpracování 2 D obrazu je poté možné vidět na obrázku 17.

V případě více zdrojů výbojové činnosti se poté jednotlivé PPD superponují a vznikne jeden součtový obraz. V něm se mohou nacházet oblasti zcela disjunktní, nebo se mohou částečně až úplně překrývat, což značně komplikuje vizuální kontrolu.



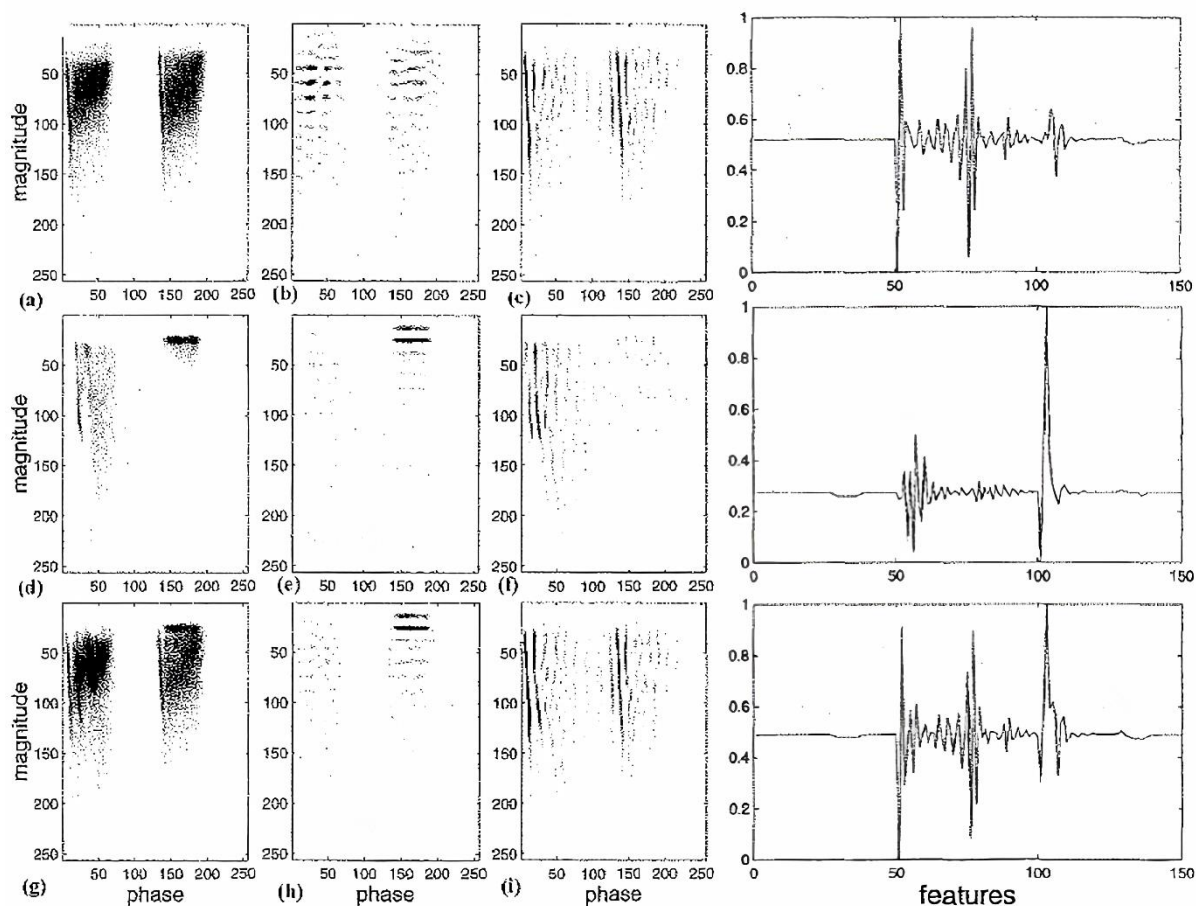
Obrázek 18: Původní a rekonstruované obrazy po použití tříúrovňové MSD: a) původní, b) přibližný, c) horizontální, d) vertikální, e) diagonální [7]

Na obrázku 19 jsou po řádcích zobrazeny rozklady výboje v dutině, povrchového výboje a jejich superpozice, vždy po originálním PR, H a poté V obrazu. Pokud se zaměříme na superponované obrazy, tak ve vertikální složce (i) jsou vidět linie v obou půlperiodách, což značí přítomnost výbojů v dutině (c). V horizontální složce (h) je poté zřetelně méně činnosti v první půlperiodě, což značí přítomnost povrchových výbojů.

Touto metodou je tedy možné, v určitých mezích, superponované zdroje výbojové činnosti rozdělit na samostatné a určit jejich typ i pouze vizuálně. Pro počítačové zpracování

Na obrázku vlevo je zobrazen obraz sestávající z kombinovaného a překrývajícího se působení výboje v dutině a povrchového výboje a jeho rozklady. V části a) se nachází původní obraz, v části b) je poté obraz přibližných koeficientů, v části c) se nachází horizontální koeficienty, v d) vertikální a v e) diagonální. Jak je možné si povšimnout, tak k největším změnám oproti původnímu obrazu došlo u horizontálního a vertikálního rozkladu a je z nich možno čerpat nejvíce relevantních informací pro další zpracování.

V případě porovnávání různých typů částečných výbojů a jejich superpozic lze poté přistoupit k porovnávání H a V obrazů jednotlivých měření.



Obrázek 19: Původní, H a V obrazy a features vektory výbojů v dutině, povrchového a jejich superpozice [7]

je nicméně nutné obrazce převést na jednodušší objekty, vektory, které jsou poté snáze zpracovatelné a obsahují podstatně méně hodnot.

Pro převod na vektor je možno využít následujícího způsobu: Nejdříve se H a V obrazce zprůměrují ve velikosti, čímž vznikne závislost průměru hodnot na fázovém posunu (pohled směrem od osy fáze). Poté se zprůměrují v pohledu od osy velikosti, čímž vznikne závislost průměru posunu na velikosti. Tyto vektory jsou poté normalizovány a poskládány za sebe, čímž vznikne vektor jeden. Příklady takto vytvořených vektorů je možno vidět na obrázku 19 v posledním sloupci. [7,8,9]

5.3 Frekvenční a časové vlastnosti (TF mapping)

Pro signál výboje, který má K samplů a $S_i(t_i)$ značí sampl detekovaný v čase t_i lze spočítat střední čas t_0 následovně:

$$t_0 = \frac{\sum_{i=0}^K t_i \cdot s_i(t_i)^2}{\sum_{i=0}^K s_i(t_i)^2} \quad (3)$$

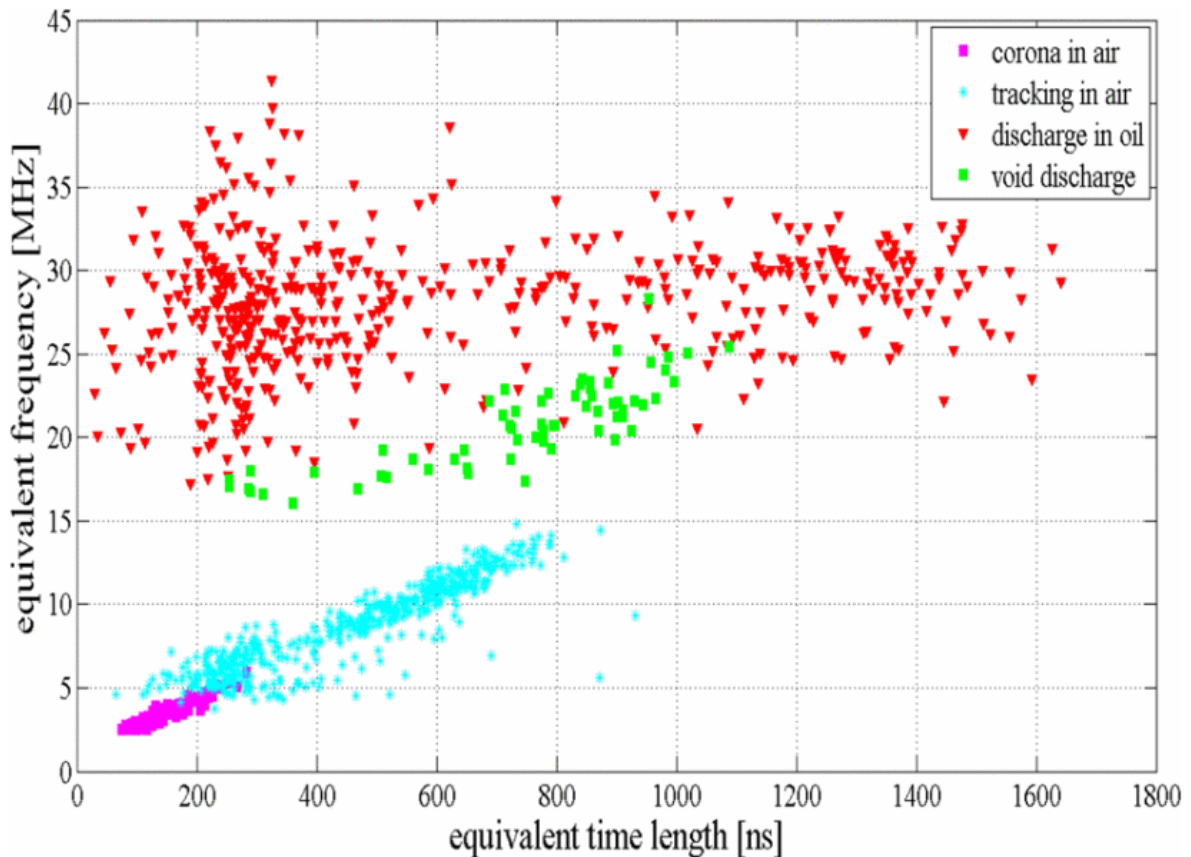
Ekvivalentní časová délka je poté definována vztahem:

$$T^2 = \frac{\sum_{i=0}^K (t_i - t_0)^2 \cdot s_i(t_i)^2}{\sum_{i=0}^K s_i(t_i)^2} \quad (4)$$

Ekvivalentní spektrum lze spočítat následujícím vztahem. Výrazem $X_i(f_i)$ je zde vyjádřena velikost harmonické složky po rozkladu FFT.

$$W^2 = \frac{\sum_{i=0}^K f_i^2 \cdot |X_i(f_i)|^2}{\sum_{i=0}^K |X_i(f_i)|^2} \quad (5)$$

Při velkém počtu změřených signálů lze poté vytvořit pro každý typ částečného výboje jeho obraz v rovině T^2 , W^2 . Touto transformací dochází ke ztrátě informací o průběhu v časové rovině, nicméně jde o vcelku rychlý výpočet, který lze provádět v reálném čase. [1,10,12]



Obrázek 20: Zobrazení výbojů v TW rovině [12]

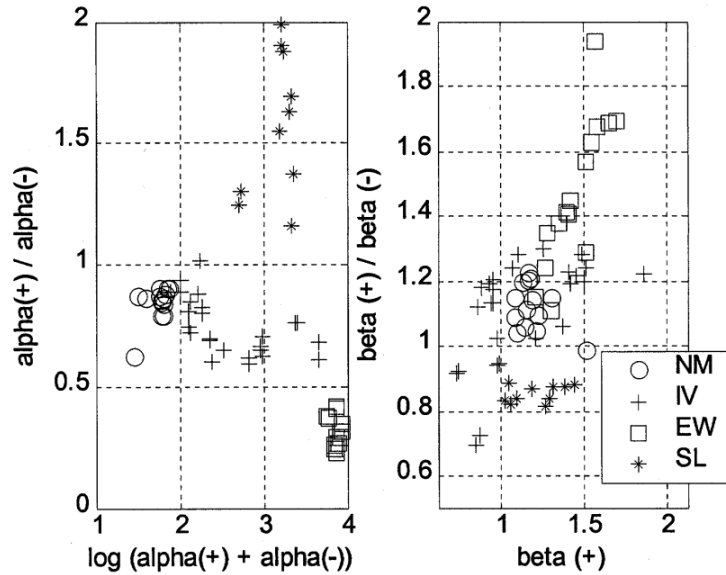
5.4 Weibullové rozdělení nad PHA křivkami

Za předpokladu, že lze pravděpodobnostní rozdělení četnosti výbojů F v závislosti na velikosti ekvivalentního náboje q aproximovat Weibullovou funkcí

$$F(q; \alpha, \beta) = 1 - \exp \left[- \left(\frac{q}{\alpha} \right)^\beta \right], \quad (6)$$

lze pro každou křivku pulzně výškové analýzy (PHA) spočítat parametry α^+ , β^+ , α^- a β^- .

Tyto parametry lze použít k analýze za předpokladu, že se pro každý typ výboje vytvoří jeho otisk. Příklad takového zobrazení lze pozorovat na obrázku 21, kde pro α hodnoty symbolizuje osa x celkovou velikost náboje a osa y reprezentuje vztah kladného a záporného pulzu. Zde NM značí normální částečný výboj na tyči, IV výboj v dutině izolace, EW koronu na konci delaminace a SL samotnou delaminaci izolace statorové tyče. [11]



Obrázek 21: Charakteristické zobrazení parametrů Weibullové funkce [11]

5.5 Cross wavelet spectrum

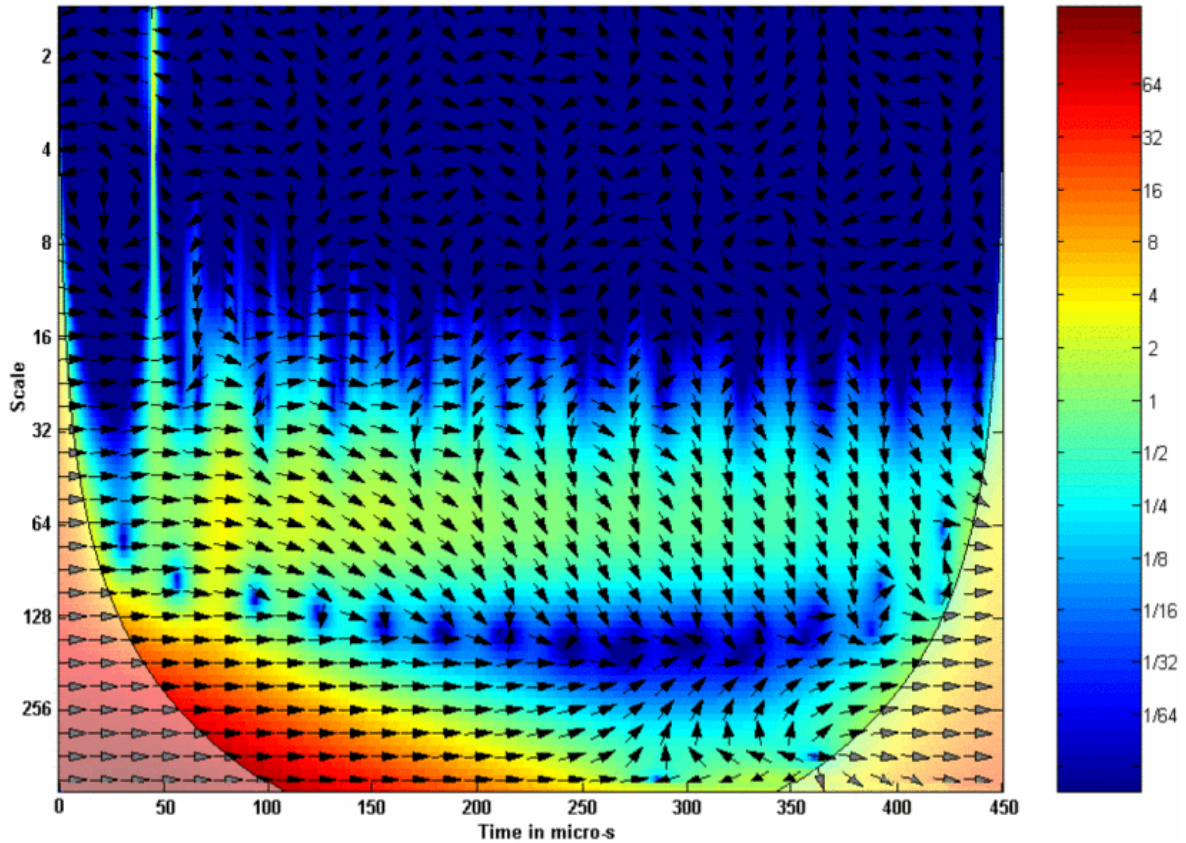
Toto spektrum vyjadřuje míru vzájemné korelace mezi dvěma signály a zvýrazňuje místa s obdobnou energií v časově frekvenční rovině. Samotné spektrum je vyjádřeno vztahem:

$$W^{xy}(s, \tau) = \frac{1}{k_{\psi} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} W^x(a, b) W^{y*} \left(\frac{a}{s}, \frac{b - \tau}{s} \right) \frac{dad b}{a^2}} \quad (7)$$

kde $W^x(s, \tau)$ a $W^y(s, \tau)$ vyjadřují wavelet transformaci $x(t)$ a $y(t)$ s ohledem na mateřskou funkci $\psi(t)$. Funkce se volí podle povahy analyzovaného signálu, přičemž se ukázalo, že pro analýzu částečných výbojů se nejvíce hodí Morletova funkce. Konstanta k_{ψ} je poté definována následovně:

$$k_{\psi} = \int_{-\infty}^{+\infty} \frac{|\psi(\omega)|^2}{|\omega|} d\omega < \infty \quad (8)$$

Příklad takového spektra je možné vidět na obrázku 22, kde osa x reprezentuje čas a osa y reprezentuje měřítko, které je úzce spjaté s frekvencí. Obarvení poté odpovídá velikosti $|W^{xy}|$ v této rovině. Křivka ve tvaru písmene U poté vyznačuje oblast největší korelace signálů.



Obrázek 22: Příklad cross wavelet spektra [12]

Z tohoto spektra lze poté určit některé z parametrů, které se dají využít v pozdější klasifikaci výboje. Jedná se o koeficienty spektrální:

$$F_1 = \frac{\sum_s \sum_\tau s \tau |W^{xy}(s, \tau)|}{\sum_s \sum_\tau |W^{xy}(s, \tau)|} \quad (9)$$

$$F_2 = \sqrt{\frac{\sum_s \sum_\tau s^2 \tau^2 |W^{xy}(s, \tau)|}{\sum_s \sum_\tau |W^{xy}(s, \tau)|}} \quad (10)$$

$$F_3 = \frac{\sum_s \sum_\tau |W^{xy}(s, \tau)|}{|W^{xy}(s, \tau)|_{peak}} \quad (11)$$

$$F_4 = \frac{\sum_s \sum_\tau |W^{xy}(s, \tau)|}{(s_{max} - s_{min})(\tau_{max} - \tau_{min})} \quad (12)$$

$$F_5 = \sqrt{\frac{\sum_s \sum_\tau (F_4 - |W^{xy}(s, \tau)|)^2}{(s_{max} - s_{min})(\tau_{max} - \tau_{min})}} \quad (13)$$

$$F_6 = |W^{xy}(s, \tau)|_{s-peak} \quad (14)$$

$$F_7 = |W^{xy}(s, \tau)|_{\tau-peak} \quad (15)$$

a fázové:

$$F_8 = \frac{\sum_s \sum_\tau s \tau |\phi(s, \tau)|}{\sum_s \sum_\tau |\phi(s, \tau)|} \quad (16)$$

$$F_9 = \sqrt{\frac{\sum_s \sum_\tau s^2 \tau^2 |\phi(s, \tau)|}{\sum_s \sum_\tau |\phi(s, \tau)|}} \quad (17)$$

$$F_{10} = \frac{\sum_s \sum_\tau |\phi(s, \tau)|}{\phi(s, \tau)|_{peak}} \quad (18)$$

$$F_{11} = \frac{\sum_s \sum_\tau |\phi(s, \tau)|}{(s_{max} s_{min})(\tau_{max} - \tau_{min})} \quad (19)$$

$$F_{12} = \sqrt{\frac{\sum_s \sum_\tau (F_{11} - |\phi(s, \tau)|)^2}{(s_{max} - s_{min})(\tau_{max} - \tau_{min})}} \quad (20)$$

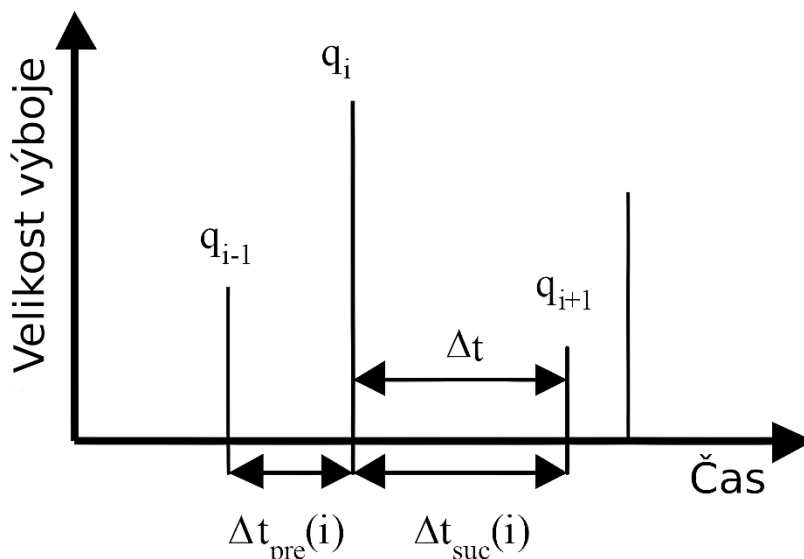
Tyto koeficienty se považují za dostatečné, nicméně mohou být doplněny například lokálními extrémy pro lepší kvantifikaci. [1,12]

5.6 Využití paměťového efektu izolace

Následující postup se používá při kvantifikaci výbojů na stejnosměrném napětí. Využívá se toho, že izolace nemá nekonečný izolační odpor a tím se přes ní může v omezenou rychlostí přesouvat náboj. Vzhledem k tomu že zde chybí vztažná hodnota fázového úhlu napětí, používá se namísto tohoto ukazatele kombinace časů a velikostí předchozích a následujících impulzů.

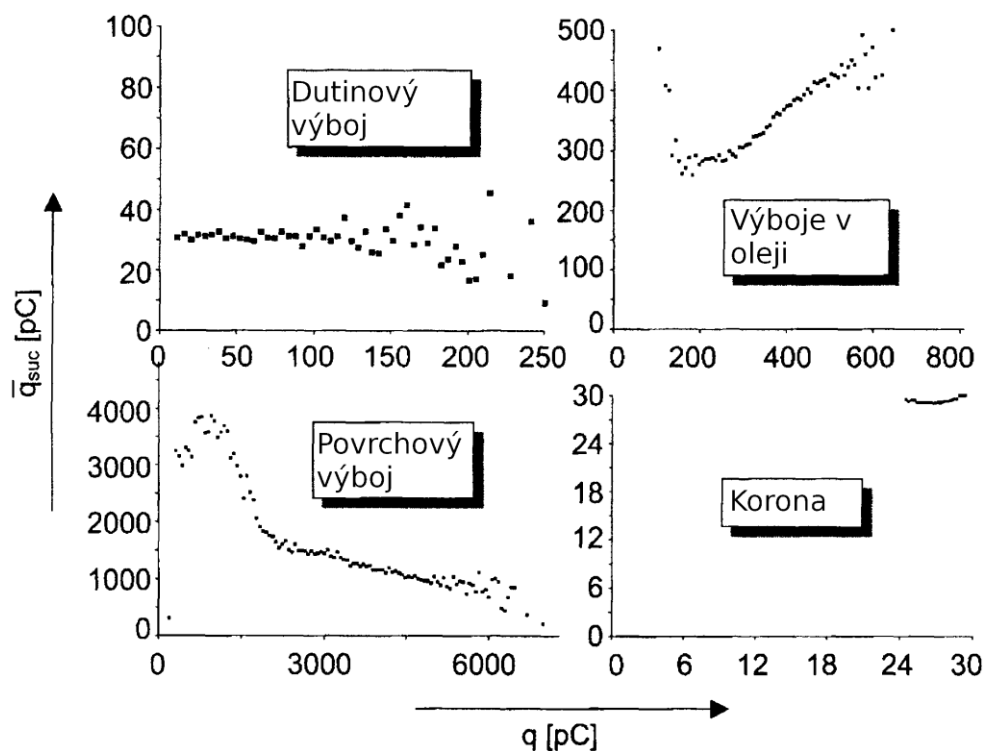
Parametr q_{suc} , který se využívá pro specifikaci výbojové činnosti při DC napětí, se spočte podle následující rovnice:

$$q_{suc} = \frac{1}{n} \sum_{i=1}^n q_{t_{i+1}} \quad (21)$$



Obrázek 23: Znázornění veličin pro výpočet q_{suc} [14]

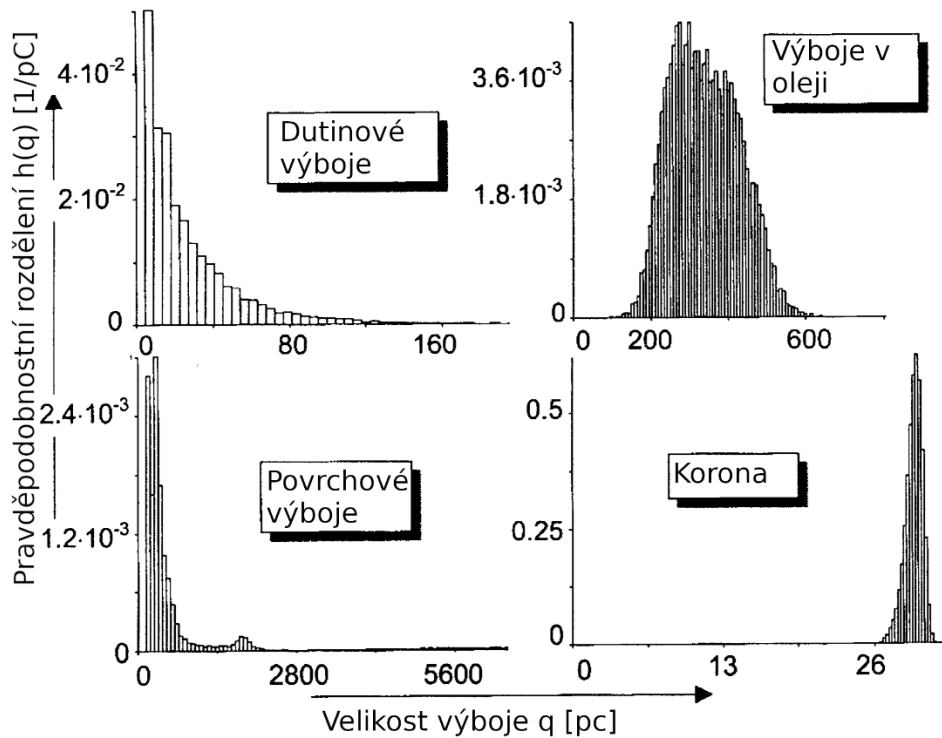
V této rovnici se počítá průměr velikosti náboje, který se nachází v určitém intervalu velikosti (například mezi 9 a 10 pC), na dohodnutém časovém intervalu. Ze závislosti q_{suc} a prvního impulzu náboje q pak lze soudit o povaze výboje. Na následujícím obrázku jsou znázorněny tyto závislosti, přičemž je patrné, že velikost q_{suc} na q je u dutinového výboje vcelku neměnná, v oleji a při povrchovém výboji vykazuje přímou a nepřímou závislost a u korony je konstantní. [12,13,14]



Obrázek 24: Závislosti q_{suc} a q pro různé typy výbojů [6]

5.7 Histogramy pravděpodobnostního rozdělení velikosti výboje

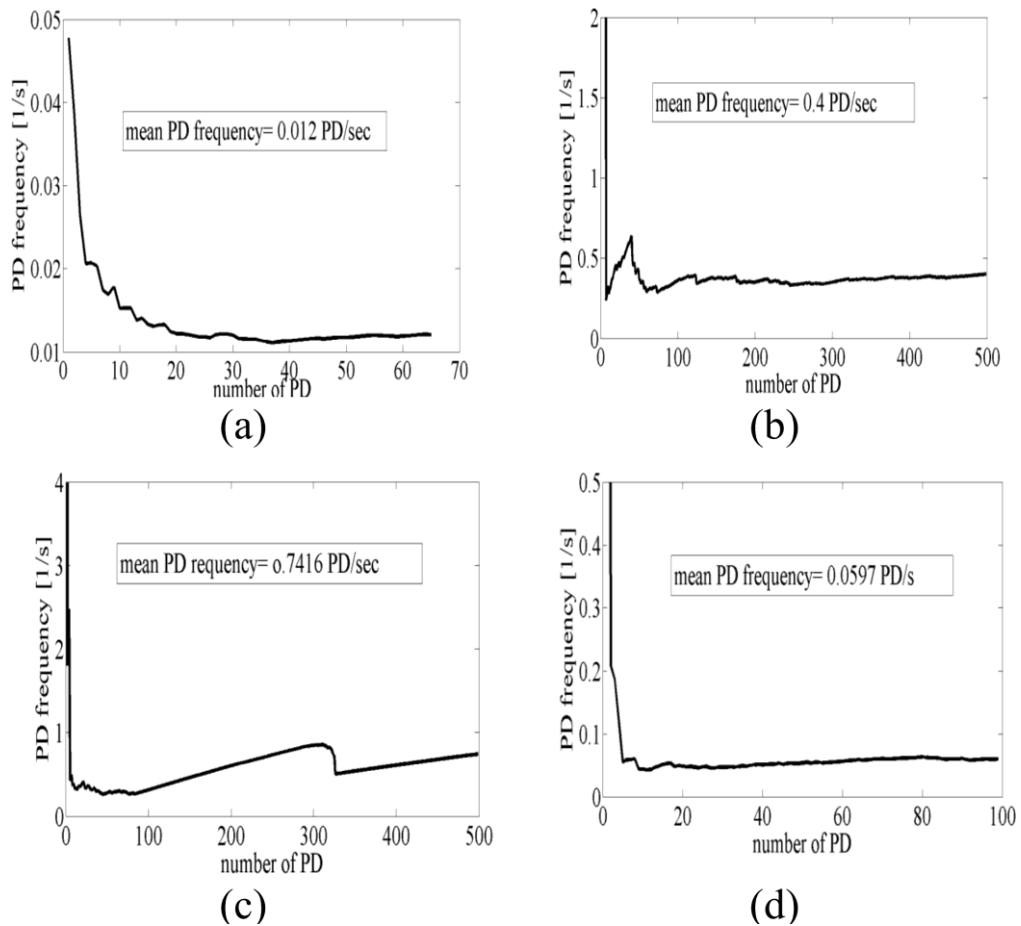
Z DC měření lze vyhodnotit histogram pravděpodobnosti výskytu konkrétní velikosti náboje a ten poté tvarově porovnat. U dutiny odpovídá průběh klesající exponenciále, povrchový výboje je poté charakterizován vysokým rozptylem hodnot a korónový výboj má výboje rozdělené ve velmi úzkém pásu. [13]



Obrázek 25: Pravděpodobnostní rozdělení velikosti výboje podle typu [6]

5.8 Průměrná frekvence výboje

Další vyhodnocování lze provádět jako měření frekvence, s jakou se výboje projevují, oproti jejich počtu v měřeném úseku. Z následující charakteristiky je zřejmé, že frekvence u všech typů klesá s jejich počtem. Zde je možné si všimnout velmi rychlé stabilizace u korony a u dutiny. U povrchového výboje je velmi znatelný postupný nárůst křivky, která má uprostřed znatelný zlom. Tento zlom byl v tomto případě dán konstrukčním uspořádáním při měření. [13]



Obrázek 26: Křivky průměrné frekvence PD proti jejich počtu pro: a) dutina, b) výboj v oleji, c) povrchový výboj, d) korona [13]

6. Klasifikace

Výše uvedené možnosti různých způsobů extrakce charakteristických veličin lze využít k počítačovému rozpoznání klíčových detailů o zdroji částečných výbojů. Obecně lze říci, že se tímto procesem rozdělují vstupní data do několika tříd, na základě kterých poté celý proces klasifikace ohodnocuje předkládaná vstupní data. Klasifikaci lze rozdělit na řízenou (s učitelem) a neřízenou (bez učitele). V případě neřízené klasifikace dochází k učení pomocí shlukové analýzy, kdy se vyhodnocují jednotlivé podobné znaky parametrů. V případě řízené klasifikace je dopředu známý počet tříd a trénovací algoritmus se snaží najít vztah, mezi vstupem a výstupem klasifikátoru.

6.1 Neuronové sítě

Neuronové sítě mají široké pole působnosti počínaje jednoduchým zpracováním signálů, vyhodnocení trendů, přes systémy řízení, jako jsou autopiloti do letadel, až po výpočty chemických sloučenin pro použití v lékařství.

V základu se síť skládá z neuronů, kdy každý neuron reprezentuje funkci, jež zpracovává jemu přidělené vstupy. Tato funkce je popsána v rovnici 22.

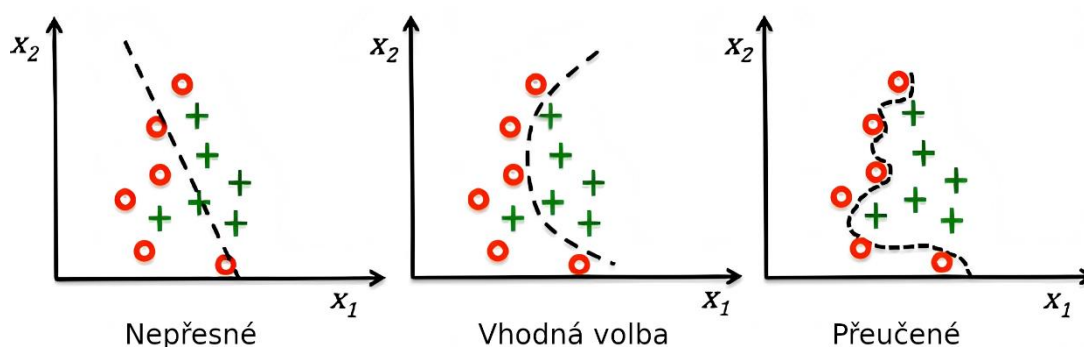
$$a^1 = \sigma(\mathbf{w} \cdot \mathbf{a}^0 + b) \quad (22)$$

Vektor vstupů \mathbf{a} neuronu je přenásoben vektorem vah \mathbf{w} a k výsledku je přičten bias b . Bias si lze představit jako určité posunutí vybuzení neuronu do jiného pásma citlivosti, například aby nebyl citlivý na oblast kolem nuly, ale kolem 10. Nakonec je tento součet ještě dán na vstup aktivační funkci neuronu, jež zajistí saturaci výstupních hodnot. Z aktivačních funkcí se používají často sigmoida, hyperbolický tangens, ReLU, softmax a softplus. Výsledná hodnota poté odpovídá velikosti vybuzení neuronu, typicky v mezích -1 či 0 až 1, v závislosti na zvolené aktivační funkci.

Neurony je poté možné spojit vedle sebe do jedné vrstvy a tyto vrstvy spojit za sebe, kdy každý neuron má spojení s každým neuronem v sousedních vrstvách, ale nemá spojení v rámci své vrstvy.

Návrhu počtu neuronů ve vrstvě a počtu vrstev je nutné určit experimentálně, nicméně je obecně známo, že jednovrstvé sítě jsou sice schopné rychlého učení, ale také s nimi nelze řešit všechny problémy. Čím více je poté řešený problém nelineárnější, komplikovanější nebo pokud výsledná funkce obsahuje nespojitost, tím více je vhodnější použít síť s dvěma a více vrstvami. Pro použití více než dvou vrstev typicky není důvod, pouze může být v některých případech vhodnější použít více vrstev o méně neuronech

z důvodu lepších výsledků pro konkrétní problém. Více vrstev lze také použít, pokud se příliš malá síť zastaví v lokálním minimu hledané funkce, k tomu, aby byl zaručen větší stupeň volnosti a větší pravděpodobnost nalezení globálního minima chybové funkce (nejvyšší přesnost sítě). Při příliš velkém počtu neuronů však může docházet k přeučení sítě, kdy se síť naučí trénovací data velmi přesně a není schopna najít řešení při velmi podobných vstupních datech. Příklad takového přeučení lze vidět na obrázku 27. Okamžik přeučení se dá zjistit, máme-li k dispozici nejen trénovací, ale i testovací množinu dat. V průběhu učení sítě se zmenšuje chyba jak na trénovacích, tak testovacích datech, ale v okamžiku, kdy se síť začíná přeučovat, se začne chyba na testovacích datech zvětšovat, zatímco bude chyba velmi nízká na trénovací množině.



Obrázek 27: Příklady výsledků málo naučené, dobře naučené a přeučené neuronové sítě [19]

6.1.1 Backpropagation

Pro učení neuronových sítí je nejčastěji používaným algoritmem adaptační algoritmus Backpropagation. Ten je určený pro učení zejména vícevrstvých sítí. V rámci učení sítě se upravují jednotlivé váhy a biasy neuronů tak, aby výsledný celkový přenos sítě odpovídal požadované hodnotě výstupu, z čehož plyne požadavek na to, že musíme mít připravené vstupy a k nim stanovené požadované výstupy trénovacích dat před začátkem celého procesu. Algoritmus Backpropagation se skládá z dílčích kroků, mezi které patří spočtení výsledku sítě, určení chyby, výpočet korekcí a aplikace korekcí na příslušné parametry. Tato část se opakuje tak dlouho, dokud není dosaženo požadované přesnosti.

Prvotní nastavení sítě probíhá nejlépe s nastavenými náhodně zvolenými hodnotami vah $\mathbf{w}^{(0)}$, kdy je zřejmé, že výsledná chybovost v tomto nastavení bude veliká. Následující popis odpovídá krokům prováděným v rámci jedné epochy. V prvním kroku se spočte výsledek a určí se jeho celková chyba E :

$$E(w) = \sum_{l=0}^q \frac{1}{2} \sum_{k=0}^n (y_{l,k} - t_{l,k})^2 \quad (23)$$

Tento vztah poté sčítá parciální chyby přes všechny tréninkové vzory l a všechny výstupy k sítě tak, že vyhodnocuje rozdíly požadovaných a skutečných hodnot výstupů sítě. Cílem tréninku je tuto chybu zmenšit na minimální možnou velikost. K tomu se využívá gradientu chyby E , jakožto vektoru parciálních derivací E podle složek vektoru hodnot w . Jednotlivými iteracemi adaptace lze najít minimum (nulový gradient) hledané funkce, ačkoliv není zaručeno, že je toto minimum minimum globálním. Při mělkém lokálním minimu tak lze do sítě vnášet šum s očekáváním, že dojde k posunutí w do oblasti, která konverguje k jinému minimu.

Pomocí parciálních derivací vzorce pro výstupní chybu a přenosové funkce každého jednotlivého neuronu lze spočítat chybové faktory pro daný neuron. Pro neurony ve výstupní vrstvě se jeho přenos vynásobí rozdílem požadované a spočtené hodnoty následovně:

$$\delta_k^n = (t_k - y_k) \cdot f' \left(w_{0,k}^n + \sum_{i=1}^m y_i^{n-1} \cdot w_{i,k}^n \right) \quad (24)$$

Pro neurony skryté vrstvy se chybový faktor spočítá obdobně, jen je místo chyby počítáno s vazbami na neurony v následující vrstvě.

$$\delta_k^n = \sum_{i=1}^q (\delta_i^{n+1} \cdot w_{k,i}^{n+1}) \cdot f' \left(w_{0,k}^n + \sum_{i=1}^m y_i^{n-1} \cdot w_{i,k}^n \right) \quad (25)$$

Oba vzorce si jsou podobné, přičemž symbolika je následující: t_k je požadovaný a y_k skutečný výstup sítě, m je počet propojení s nižší vrstvou a q s vyšší vrstvou.

Pomocí těchto faktorů lze poté určit korekci váhy každé vazby

$$\Delta w_{i,k}^n = \alpha \cdot \delta_k^n \cdot y_i^{n-1} \quad (26)$$

a biasu

$$\Delta w_{0,k}^n = \alpha \cdot \delta_k^n \quad (27)$$

Posledním krokem každé epochy je poté přičtení každé korekce k odpovídající hodnotě

$$w_{i,k}^{n, \text{nové}} = w_{i,k}^{n, \text{původní}} + \Delta w_{i,k}^n \quad (28)$$

a určení, zdali má skončit výpočet. Typickými podmínkami ukončení je jednak počet iterací, kdy proces učení může trvat stovky až tisíce iterací, a jednak se využívá toho, že se sleduje trend velikosti chyby, kdy za předpokladu, že se chyba snižuje, výpočet pokračuje a výsledek s každou další iterací konverguje a pokud se již nemění nijak zásadně, tak se předpokládá,

že síť našla minimum chybové funkce a výpočet se zastaví.

Ve vzorcích 26 a 27 si lze povšimnout nadále koeficientu učení α , kterým lze proces učení urychlit, nebo zpomalit. Typicky se volí v rozmezí 0,0001 až 0,01, kdy příliš nízká hodnota znamená, že se síť bude učit po malých krocích, a tím i delší dobu, a vysoká hodnota proces učení urychlí. V případě vyšších hodnot také může docházet k tomu, že chyba sítě opustí oblast minima a nezkonverguje.

Na tento algoritmus je navíc kladena podmínka na diferencovatelnost aktivační funkce neuronů, plynoucí ze vzorců 24 a 25, spojitost aktivační funkce a funkce navíc musí být monotónně neklesající. Mezi nejpoužívanější funkce, splňující tyto podmínky, se řadí sigmoida a hyperbolický tangens.

Při klasifikaci částečných výbojů se doporučuje použít minimálně dva různé zdroje vstupních hodnot, například statistické parametry a image processing, aby byl dosažen rozumný předpoklad k tomu, že síť bude konvergovat.

Pro klasifikaci částečných výbojů se používají i další, pokročilejší, druhy neuronových sítí, které budou krátce popsány dále. [1,15,16,17,18,19]

6.1.2 Self-Organizing Maps

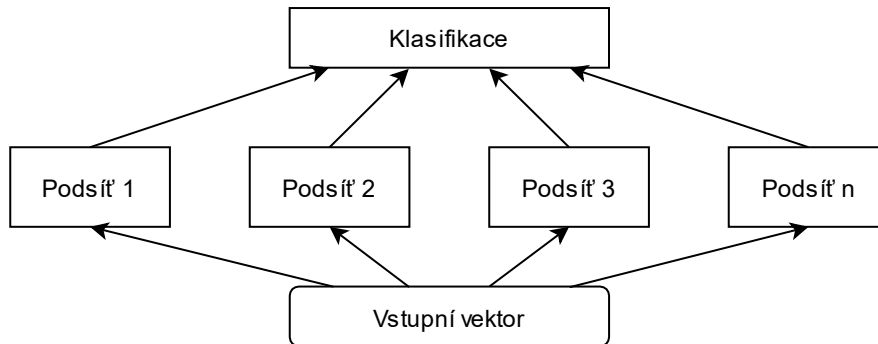
Self-Organizing mapy, také známé jako Kohonenovy mapy, je název pro neuronovou síť, která se skládá z Kohonenových vrstev. Kohonenova vrstva poté převádí vstupní vícedimenzionální prostor na prostor o typicky dvou rozměrech. Učící algoritmus využívá funkce sousedství, kdy se pro každý neuron určí jeho vazby na okolní neurony. Pro každý neuron se určí euklidovská vzdálenost od učícího vzoru a nejbližšímu neuronu jsou upraveny váhy tak, aby se učícímu vzoru více přiblížil. Podle příslušnosti do okolí tohoto neuronu se odpovídající měrou přizpůsobí i váhy sousedících neuronů. Tento algoritmus patří do kategorie učení bez učitele, neboť není znám předem očekávaný výsledek. Výsledné rozřazení na jednotlivé určované kategorie poté probíhá pomocí standardní výstupní vrstvy, která je na kohonenovu vrstvu plně napojena. Zde se již využívá učení s učitelem.[34, 35]

6.1.3 Counterpropagation síť

Tento typ sítí se vyznačuje tím, že hledá vzájemnou obousměrnou funkci mezi vstupními a výstupními daty. Pro klasifikaci částečných výbojů nelze ale předpokládat, že by vstupní data byla závislá na výstupních přímým vztahem. Z těchto důvodů lze využít pouze její dopřednou (forward) variantu. Učící algoritmus pracuje bez učitele a je podobný tomu, jaký je využíván u Kohonenových map.

6.1.4 Modulární síť

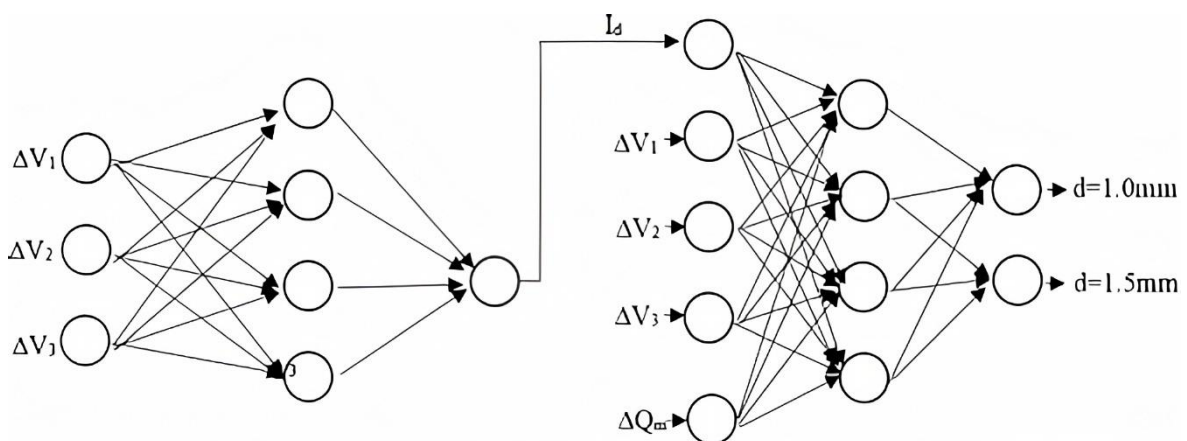
Principem modulárních neuronových sítí je rozdělení komplexního problému na jednotlivé podproblémy. Největší výhodou tohoto uspořádání je, že jednotlivé podsítě lze uzpůsobit pro individuální úkoly snáze než uzpůsobit jednu velkou síť na komplexní úkol. Na následujícím obrázku je příklad takového uspořádání znázorněn. Vyobrazené podsítě lze v případě klasifikace částečných výbojů vyčlenit například pro specifické znaky v signálu, či podle typu částečného výboje.



Obrázek 28: Topologie modulární sítě [34]

6.1.5 Kaskádní síť

Kaskádních sítí je dosaženo pomocí sériového zapojení backpropagation sítí. Standardně se u klasifikace částečných výbojů využívá dvou stupňového zapojení. Každý stupeň poté zpracovává různé vstupní informace a druhý stupeň taktéž zpracovává výstup prvního stupně, který tvoří velmi dobře zpracovatelný vstup. Tím je dosaženo vysoké míry přesnosti klasifikace. [34]

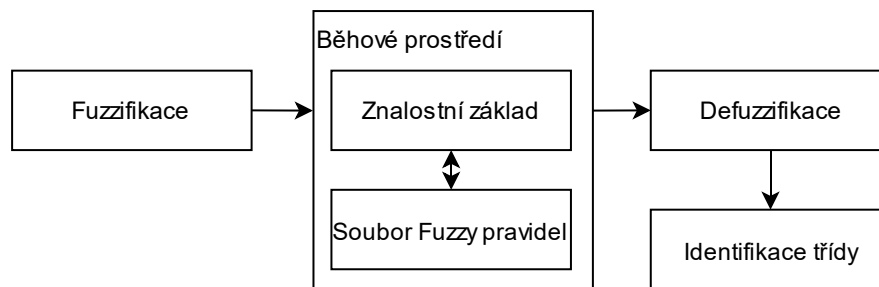


Obrázek 29: Topologie kaskádní sítě [1]

6.2 Klasifikace pomocí Fuzzy logiky

Při klasifikaci fuzzy logikou se využívá toho, že hodnoty některých parametrů částečných výbojů trpí na vysokou variabilitu, a jsou zde tedy rozřazeny do několika fuzzy hodnot podle velikosti na velmi malé, malé, střední, velké a velmi velké. V rámci fuzzifikace poté dojde také k určení míry náležitosti konkrétní hodnoty do fuzzy hodnoty. Výsledek je poté dán výstupem klasifikátoru a jeho defuzzifikací.

K vyhodnocování poté může být použita například neuro-fuzzy síť založená na Min-Max modelu, který dovoluje určení přesného typu výboje. [34]



Obrázek 30: Schéma Fuzzy klasifikace [34]

6.3 Klasifikace pomocí výpočtu vzdáleností

Tento princip klasifikace spočívá v určení vzdálenosti zjišťovaného vektoru parametrů od příslušného referenčního bodu. Předpokladem jsou změřená referenční data pro každou řešenou třídu. Ze zdrojových referenčních dat jednotlivých určovaných tříd se pomocí K-means algoritmu určí jednotlivé clustery a klasifikace poté probíhá jednou z následujících metod.

6.3.1 Klasifikace pomocí minimální vzdálenosti

Zde se využívá toho, že se ke každému zdrojovému clusteru určí průměrný vektor jeho hodnot m_i . Následně se určuje vzdálenost mezi určovaným vektorem a všemi vektory m_i . Vzdálenosti lze poté počítat pomocí několik základních způsobů.

Pravouhlá (Manhattan) vzdálenost:

$$d_i(x) = \frac{1}{N} \sum_{j=1}^N |x_j - m_{ij}|, i = 1, \dots, M \quad (29)$$

Určuje se vzdálenost přes N parametrů vstupního vektoru vůči každému z M clusterů.

Zobecněná L_z vzdálenost:

$$d_i^z(x) = \sqrt[z]{\frac{1}{N} \sum_{j=1}^N w_{ij} |x_j - m_{ij}|^z}, i = 1, \dots, M \quad (30)$$

Jedná se o obecné vyjádření vzdálenosti v prostoru. Pravoúhlá a Euklidovská vzdálenost jsou poté speciálními případy pro $z=1$ a $z=2$. Váhový faktor w_{ij} bývá standardně roven 1, ale za předpokladu že známe standardní odchylky zdrojových clusterů, lze s výhodou přesnějšího řešení položit tento faktor roven $1/\sigma_{ij}^2$.

6.3.2 Klasifikace pomocí nejbližšího souseda

Zde se výsledná třída výboje neurčuje pomocí vzdálenosti vůči středu celého clusteru, ale vůči jednotlivým změřeným samplům vyjádřených vektorem s_l . Výsledek poté odpovídá stejné třídě, jako je třída jeho nejbližšího sousedního samplu. Pro určení těchto vzdáleností lze využít výše zmíněné způsoby určení vzdálenosti, nebo normalizovaného skalárního součinu sledovaného vektoru a porovnávaného vektoru. Tento součin poté odpovídá:

$$0 \leq \frac{x' \cdot s_l}{\|x\| \|s_l\|} \leq 1 \quad (31)$$

Zatímco při určování náležitosti pomocí vzdálenosti se vyhledává nejmenší vzdálenost, tak v případě tohoto součinu se hledá hodnota nejvyšší blížká jedné. Pro zvýšení přesnosti této metody lze také využít určení několika nejbližších sousedů, kteří jsou bližší než specifická vzdálenost, a poté vybrat nejzastoupenější třídu. Je ovšem důležité spočítat zastoupení této třídy a druhé nejbližší, neboť v některých případech mohou být zastoupeny přibližně stejnou měrou a poté je nutné výsledek zneplatnit.

6.3.3 Klasifikace polynomem

Klasifikace polynomem využívá převodu vstupního vektoru parametrů na skalár pomocí souboru vah w , které tvoří jednotlivé konstanty. Počítaný polynom lze vyjádřit následovně:

$$d_i(x) = w_{i0} + w_{i1}x_1 + w_{i2}x_2 + \dots + w_{iN}x_N \quad (32)$$

Soubor těchto polynomů pro všechny řešené třídy lze poté zapsat v maticovém tvaru

$$\mathbf{d}_i(x) = \mathbf{w}_i' \mathbf{x}_a \rightarrow \mathbf{D}(x) = \mathbf{W}\mathbf{X}, \quad (33)$$

kde matice \mathbf{W} je maticí vah, \mathbf{X} je maticí řešených vektorů a \mathbf{D} je matice odpovídajících $d_i(x)$. Matice prvky matice \mathbf{W} jsou určovány iteračně tak, aby splňovaly podmínku pro vektor x ,

který patří do třídy i :

$$d_{ij}(x) > d_{kj}(x) \text{ pro } k \neq i \quad (34)$$

6.4 Statistická klasifikace

Pro tento způsob klasifikace se využívá především statistických hodnot zdrojových clusterů dat, jako jsou parametry jejich normálního rozdělení, nebo standardních odchylek veličin.

6.4.1 Bayesův klasifikátor

Tento klasifikátor využívá statistických parametrů jednotlivých clusterů vstupních dat. Především se zde využívá parametrů více rozměrového normálního rozdělení. Výsledná rozhodovací funkce může být pro případ částečných výbojů převedena na vzdálenostní ve tvaru:

$$d_i(x) = \exp \left[-0,5 \sum_{k=1}^N \left(\frac{x_k - m_k}{\sigma_{ik}} \right)^2 \right] / \left(M \sqrt{(2\pi)^N \prod_{k=1}^N \sigma_{ik}^2} \right) \quad (35)$$

Tato rovnice je již vyjádřena ve tvaru vzdálenost, tudíž k ní lze přistupovat stejně jako u předchozích klasifikátorů, a hledat minimální hodnotu parametru $d_i(x)$. Výsledná třída je pak rovna třídě nejbližšího clusteru.

6.4.2 Klasifikace pomocí intervalu shody

Jednotlivé prvky porovnávaných vektorů parametrů se vyznačují určitou mírou odchylky. Při tomto způsobu klasifikace se pomocí této odchylky určí intervaly každého z prvků vektorů zdrojových dat a pro zjišťovaný vektor se poté posuzuje míra shody s jednotlivými intervaly.

Interval CI , který je stanoven zvlášť pro každý prvek vektorů každé třídy, odpovídá má následující hranice:

$$CI_1 = M_{SO} - \frac{\lambda\sigma}{\sqrt{N}}; \quad CI_2 = M_{SO} + \frac{\lambda\sigma}{\sqrt{N}} \quad (34)$$

kde M_{SO} je průměrná hodnota z N měření stejného typu výboje, λ je statistickým parametrem závislým na N . Celý interval se potom nastavuje tak, aby pokryl 95 % všech hodnot. To zabrání velmi krajním hodnotám jeho ovlivňování. Při určování třídy se následně určí, do jakého intervalu daný prvek vektoru přísluší a vyhodnotí se nejzastoupenější třída. [34]

7. Využití možností klasifikace v laboratoři

Pro praktickou část práce byly změřeny pulzy jednotlivých částečných výbojů pomocí osciloskopu. Pulzy byly měřeny pomocí osciloskopu LeCroy WavePro 7300A. Osciloskop je schopný snímat s vzorkovací frekvencí až 10 GS/s na všech kanálech, čehož bylo při měření využito. Výstupní data lze poté uložit do souboru pro pozdější zpracování pomocí PC. Výboje jsou všechny korónového typu.

Měření proběhlo pro osm konfigurací zkoušeného objektu, přičemž bylo měřeno při dvou různých hodnotách vzdálenosti elektrod, dvou různých vzdálenostech stranového vysunutí a dvou různých tlacích. Jednotlivé konfigurace jsou poté označeny jako n0 až n7.

7.1 Možnosti pulse-shape analýzy

Co se týče možností analýzy tvaru průběhu pulzu částečného výboje změřeného pomocí osciloskopu, nabízí se využití především následujících metod zpracování.

První metoda vychází z článku [36], ve kterém se autoři zabývají pulse-shape analýzou signálu pro rozhodování mezi tím, zdali se jedná o pulz bubliny v izolaci, či elektrický stromeček. Dále vytvořili neuronovou síť pro každý typ výboje, přičemž u bublin má síť za úkol ohodnotit velikost bubliny a předpovědět další vývoj a u elektrického stromečku má předpovědět jeho délku. Z hlediska popisu pulzu využívají parametry jako je doba náběhu, doba týlu, plocha křivky, šířka a maximální velikost amplitudy. Těchto pět parametrů poté přivádí na vstup feed-forward sítě, která má jeden výstupní neuron, který je roven 0 pro stromeček a 1 pro bublinu. V závislosti na tomto typu jsou poté parametry předány další síti, vždy specifické pro daný typ. Pro elektrické stromečky bylo výstupem osm neuronů, které odpovídaly konkrétní délce stromečku. Skutečná velikost stromečku poté byla spočtena z míry vybuzení dvou sousedních neuronů. Pro určení míry degradace u bublin byly poté změřeny vzorky s osmi různými stupni degradace, na kterých se síť učila a které následně klasifikovala. V případě délky stromečků se tímto způsobem podařilo klasifikovat správně 94 % případů a pro bubliny 91 % případů.

Jako další metoda, která by pro tento účel byla v rozumné míře použitelná se jeví metoda použita v článku [37]. Autoři zde porovnávali charakteristiky korónových a povrchových výbojů při AC a DC namáhání, kde spočítali dobu náběhu, týlu a také šířku pulzu v polovině amplitudy. Tyto parametry poté zobrazili do histogramů podle polarity výboje a typu namáhání. Výsledky byly zpracovány jen vizuálně, ale plyne z nich, že zatímco pro povrchový výboj jsou parametry velmi odlišné a vizuálně dobře rozlišitelné, pro

korónový výboj se parametry svými rozsahy překrývají. Z důvodu nejistých závěrů pro různé konfigurace námi změřených průběhů tak nad touto metodou není uvažováno.

V článku [38] byla naopak pro klasifikaci použita konvoluční neuronová síť. Tyto sítě jsou známy pro svoje pokročilé schopnosti strojového učení. V tomto případě ze změřených pulzů extrahovali podobné parametry, jako jsou použity v prvním z výše zmíněných článků, které doplnili o další, například hodnotu výboje v pC, střední hodnotu průběhu, efektivní hodnotu a několik dalších. V článku použili konvoluční síť se třemi konvolučními vrstvami, třemi pooling vrstvami a jednu plně napojenou vrstvu. Výsledná přesnost této sítě se poté pohybovala na úrovni 92,6 %, přičemž byla tatáž data analyzována také pomocí feed-forward sítě trénované pomocí backpropagation algoritmu. Oproti této síti si konvoluční vedla lépe přibližně o 6,5 %

7.2 Příprava dat pro neuronovou síť

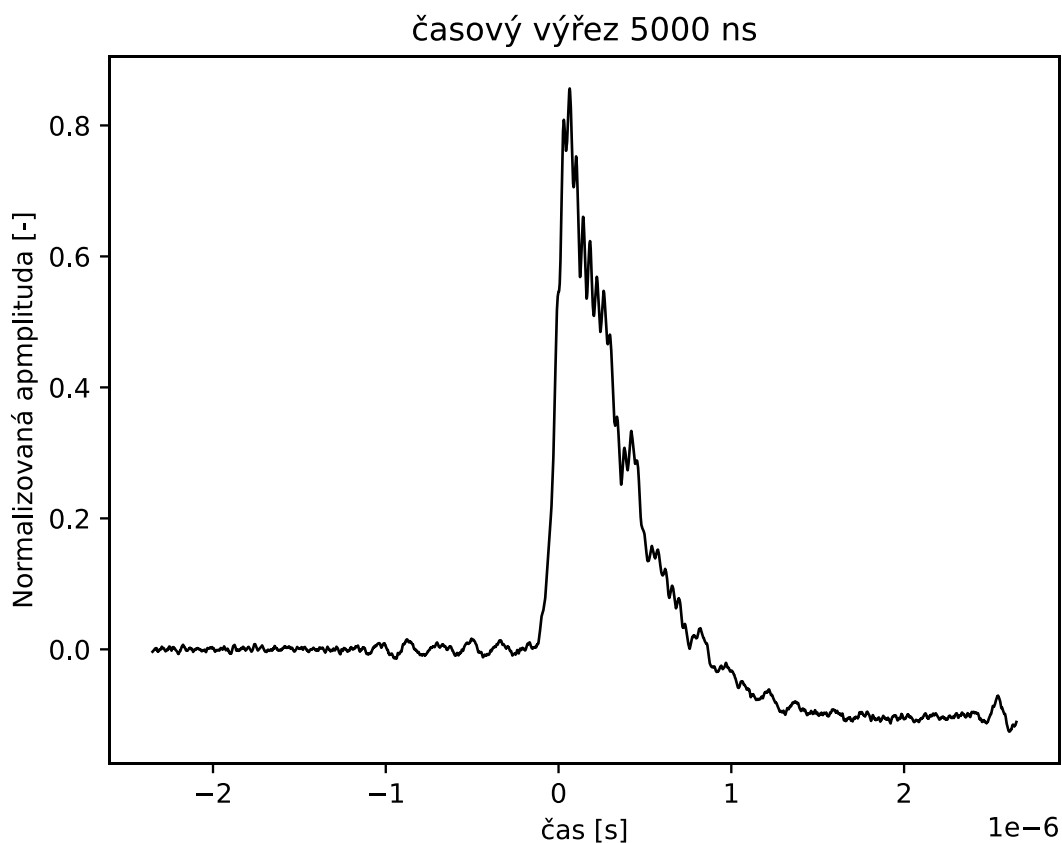
Pro praktickou část byl vytvořen skript *prepar.py* ve skriptovacím jazyce Python ve verzi 3.8, který o jednotlivých pulzech získává potřebné informace, a jednoduchá feed-forward neuronová síť *NN.py*, která provádí rozpoznávání jednotlivých konfigurací.

Data byla změřena pro každou konfiguraci po 200 opakováních. Vždy 100 vzorků pro kladnou polaritu a 100 vzorků pro zápornou polaritu. Celkově jde tedy o 1600 změřených pulzů. Vzorky byly poté pro každou konfiguraci rozděleny po 140 měřeních pro potřeby tréninku, 40 měřeních pro validaci a kontrolu míry přeučení sítě a 20 měřeních na finální testování schopnosti kategorizace.

Jako zdroj surových dat jsou k dispozici přímá měření v podobě TRC souborů, které jsou poskytovány přímo osciloskopem. Pro potřeby následné klasifikace jsou zjišťovány parametry shodné s [36], které byly doplněny o další, buďto po vzoru [39] nebo vlastní, jež se ukázaly, že by mohly přispět k lepší schopnosti naučení neuronové sítě.

Ve skriptu se proto převádí tyto zdrojové soubory na přímé hodnoty vektorů změřené

výchylky y a času osciloskopu x . Data jsou poté zbavena vysokofrekvenčního šumu pomocí výpočtu klouzavého průměru přes 81 samplů. Tím je dosaženo relativně hladkého průběhu při zachování přesnosti. Průměr z 81 samplů při dané samplovací frekvenci na měřeném časovém okně 5000 ns odpovídá přibližně 0,16 % z celkové doby měření pulzu, což způsobuje pouze zanedbatelnou deformaci původního signálu.



Obrázek 31: Ukázka normalizovaného pulzu (kategorie – n4)

Následně dochází k normalizaci amplitudy do rozsahu 0 až 1, odečtení stejnosměrné složky, která se detekuje z průměru první μs , a otočení pulzu tak, aby měl kladnou výchylku. Tím je dosaženo toho, že má průběh přibližně nulovou hodnotu až do svého začátku. Jak vypadá průběh po těchto krocích je možné vidět na obrázku 31.

Nyní je možné spočítat parametry pulzů, které budou později vyhodnocovány.

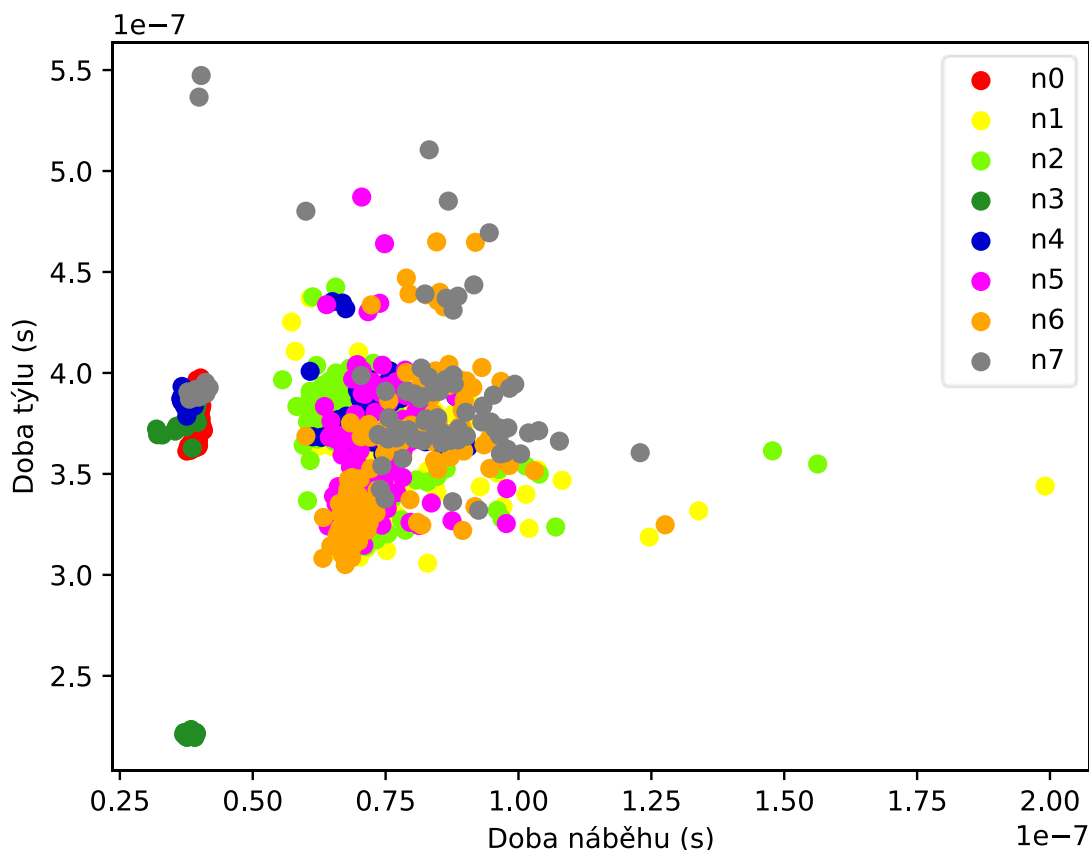
7.2.1 Doba náběhu

Doba náběhu je počítána jako rozdíl časů mezi body, ve kterých pulz vytíná pásmo 20 a 80 % své maximální amplitudy při náběhu z nulové hodnoty. Tato doba je u korónového výboje závislá především na rychlosti ionizace plynu v místě výboje a je silně ovlivněna schopností měřicího systému přenášet vysoké frekvence na svorky osciloskopu.

7.2.2 Doba týlu

Doba týlu je počítána ve stejném smyslu jako doba náběhu jen s tím rozdílem, že se určuje v sestupné části pulzu čili jako rozdíl při přechodu z 80% amplitudy na 20% hodnoty.

Vzájemné zobrazení doby náběhu a doby týlu je vidět na následujícím obrázku:



Obrázek 32: Zobrazení závislosti doby náběhu pulzů na jejich době týlu

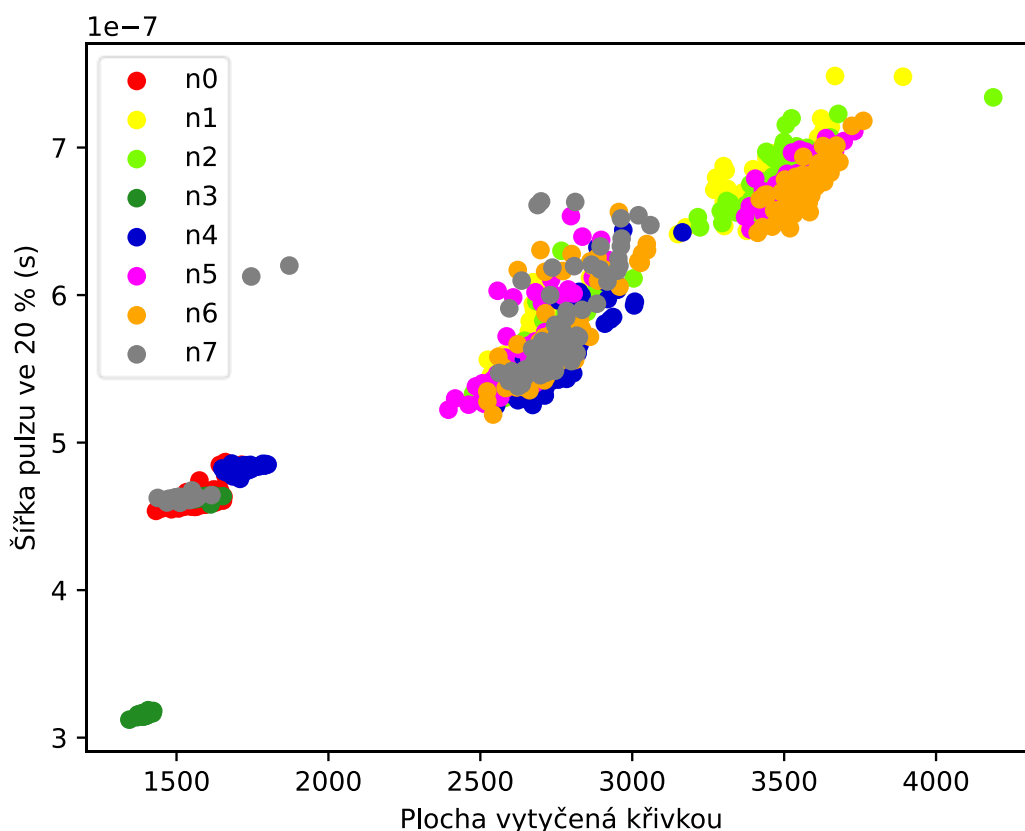
Z obrázku je dobré si povšimnout clusterů hodnot v oblasti doby náběhu 39 ns, která je společná pro konfigurace n0, n3 a záporné pulzy n4 a n7. V této oblasti si jsou spočtené hodnoty pro jednotlivé konfigurace sice velmi blízké, v porovnání s ostatními hodnotami, ale hodnoty se nepřekrývají, což značí, že by se mohly vyhodnocovat spolehlivěji.

7.2.3 Plocha křivky

Dalším parametrem, který je u pulzů počítán je plocha, která je vytyčená křivkou pulzu a nulovou hodnotou osy y. Tato plocha se počítá mezi prvním a posledním průchodem 20% amplitudy pulzu. K výpočtu je využito lichoběžníkové pravidlo, které je implementováno v knihovně *numpy* funkcí *trapz*.

7.2.4 Šířka pulzu

Šířka pulzu je ve skriptu počítána ve dvou úrovních. Jednak je počítána jako rozdíl prvního a posledního průchodu 20 % amplitudy a jednak v polovině amplitudy. Pro přidání výpočtu v polovině jsem se rozhodl, neboť některé změřené konfigurace vykazují vysokou míru zvlnění a docházelo k tomu, že poslední průchod 20 % nebyl vždy v přibližně stejném časovém okně, ale byl ovlivněn amplitudou tohoto superponovaného zvlnění. To je patrné například z předchozího obrázku z konfigurace n7, kde se doba týlu pohybovala ve dvou intervalech



Obrázek 33: Závislost šířky pulzu na ploše pulzu pod křivkou

Z vyobrazené závislosti opět vyplývá, že konfigurace n0, n3 a záporné pulzy n4 a n7 mají přesně vymezené oblasti, a tedy vyšší pravděpodobnost správnosti následné klasifikace naučenou sítí. Všechny ostatní konfigurace se ve velké míře v hodnotách překrývají.

7.2.5 Maximální výchylka

Jedním z parametrů, podle kterých se síť orientuje, je také maximální amplituda výchylky. Ta je počítána z nenormalizovaných hodnot, což by mělo zajistit rozdílnější hodnoty. Výsledné hodnoty jsou ale v některých konfiguracích ovlivněny výše zmíněným superponovaným zvlněním. To je patrné zvláště v kladných pulzech konfigurace n3, kdy se

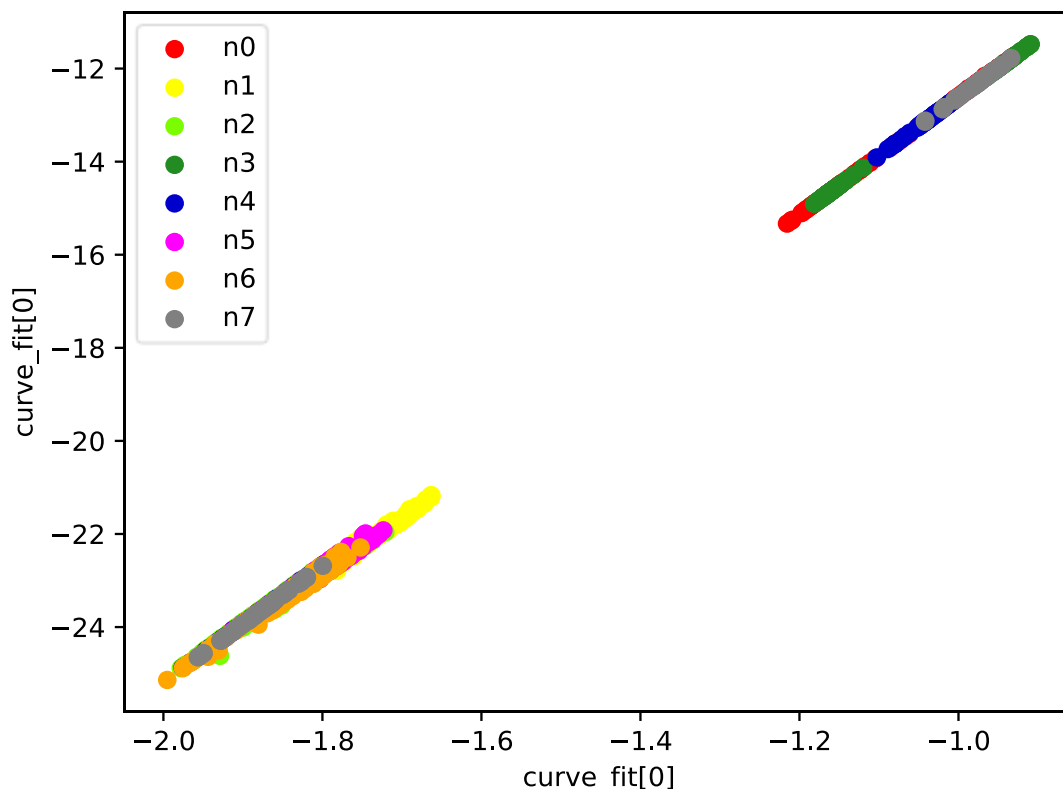
tato výchylka pohybuje okolo hodnoty 11,5 μV , zatímco všechny ostatní konfigurace jsou přibližně v intervalu 0,8 až 1,6 μV .

konf.	amp. (μV)	konf.	amp. (μV)	konf.	amp. (μV)	konf.	amp. (μV)
n0 +	1,96	n2 +	1,24	n4 +	1,19	n6 +	0,92
n0 -	1,59	n2 -	0,76	n4 -	1,09	n6 -	0,83
n1 +	1,03	n3 +	11,54	n5 +	1,28	n7 +	0,85
n1 -	0,69	n3 -	1,84	n5 -	0,70	n7 -	1,19

Tabulka 4: Průměrné maximální výchylky jednotlivých konfigurací podle polarity

7.2.6 Proložení

Parametry proložení se počítají na sestupné straně pulzu od hodnoty, kdy je výchylka naposledy rovna 80 % maximální amplitudy, přes následujících 10 000 samplů, což při 10 GS/s odpovídá 1 μs trvání. Hodnoty jsou ale před proložením posunuty o 2,5 μs později a ve výchylce o 1 nahoru. Původně jsem chtěl těmito parametry vyjádřit proložení exponenciálou, ale při proložení přímkou hodnoty $\log(y)$ a x se ukázalo, že síť nezlepší schopnost klasifikování. Při pokusech se ukázalo, že pokud se přímkou proloží hodnoty $\log(y)$ a $\log(x)$, pak se síť je schopna naučit mnohem lépe. Výstupem jsou poté dva parametry, které toto proložení respektují. Tyto parametry jsou zde zobrazeny.



Obrázek 34: Parametry proložení

7.2.7 Polarita

Posledním parametrem, určeným při zpracování změřených průběhů, je polarita pulzu. Síť se polarita předává jako parametr, který je roven nule, pokud se jednalo o kladný pulz, nebo jedničku, pokud se jednalo o záporný pulz.

Těchto devět parametrů a číslo konfigurace se poté uloží jako pole pomocí *numpy* knihovny a je tímto připraven soubor, na kterém lze naučit neuronovou síť.

7.3 Využití neuronové sítě pro klasifikaci

Základ klasifikace je postaven na knihovně Keras, což je open-source knihovna pro snadné a přehledné vytváření neuronových sítí. Keras je nadstavbou nad knihovnou TensorFlow. TensorFlow je knihovna vyvíjená společností Google pro potřeby pokročilého strojového učení.

Keras je v základu možno použít pro tvorbu jednoduchých feed-forward sítí a také dvou druhů pokročilých neuronových sítí a sice rekurentních a konvolučních sítí. Výpočty je poté možno provádět na standardních procesorech, či na CUDA® platformě pro grafické karty NVIDIA, která se vyznačuje mnohem rychlejším výpočtem sítě.

7.3.1 Možnosti pracovního prostředí

Pro účely pozdějšího popisu konkrétní sítě jsou na následujících řádcích uvedeny možnosti Keras knihovny, jakými lze specifikovat model neuronové sítě a jakými lze tyto sítě upravovat a učit.

Model

Model je třídou, která zahrnuje metody pro tvorbu sítě a práce s ní. V rámci třídy *model* je poté definována třída *Sequential*, která zahrnuje následující metody:

- Add Přidání nové vrstvy hned za předchozí
- Pop Odstranění poslední vrstvy modelu

V rámci třídy *model* jsou to potom především tyto metody:

- Compile Metoda pro stanovení optimizera, ztrátové funkce a metrik sítě
- Fit Slouží pro naučení sítě. Stanovují se v ní vstupní vektory a k nim příslušné výstupní vektory dat, validující data, případně kroky, které je nutné provést po konci každé epochy

- Evaluate Vrací přesnost a chybovost pro zvolenou ztrátovou funkci pro množinu dat
- Predict Vrací třídu jednoho vstupu, která je ohodnocena největší pravděpodobností

Vrstvy

Tato třída definuje možnosti, jakými lze specifikovat konkrétní přidávanou vrstvu sítě. Pro nás, z hlediska možností použití klasifikace devíti číselných parametrů pomocí feed-forward sítě, jsou zajímavé především následující třídy, které definují standardní a redukční vrstvy:

- Dense Definuje jednoduchou, plně propojenou vrstvu
- Activation Vrací hodnotu specifikované aktivační funkce pro daný vstup
- Dropout V závislosti na daném parametru nastavuje výstupy předchozí vrstvy na nulovou hodnotu. Tím se snaží zabránit přeučení sítě. Tato vrstva se poté využívá pouze při učení.
- GaussianNoise Přidává mezi vrstvy šum, čímž se snaží zabránit přeučení a síť se tím snaží generalizovat pro širší rozptyl možností vstupů [40]

7.3.2 Charakteristika použité NN

Pro účely klasifikace byla ve skriptu definována feed-forward neuronová síť skládající se ze čtyř *dense* vrstev. Vrstvy mají postupně 1024, 256, 64 a 8 neuronů, přičemž poslední vrstva je výstupní čili počet jejich neuronů odpovídá počtu konfigurací změřených průběhů. Počty neuronů byly zvoleny takto experimentálně a při snížení jejich počtu docházelo ke zvýšení chybovosti klasifikace.

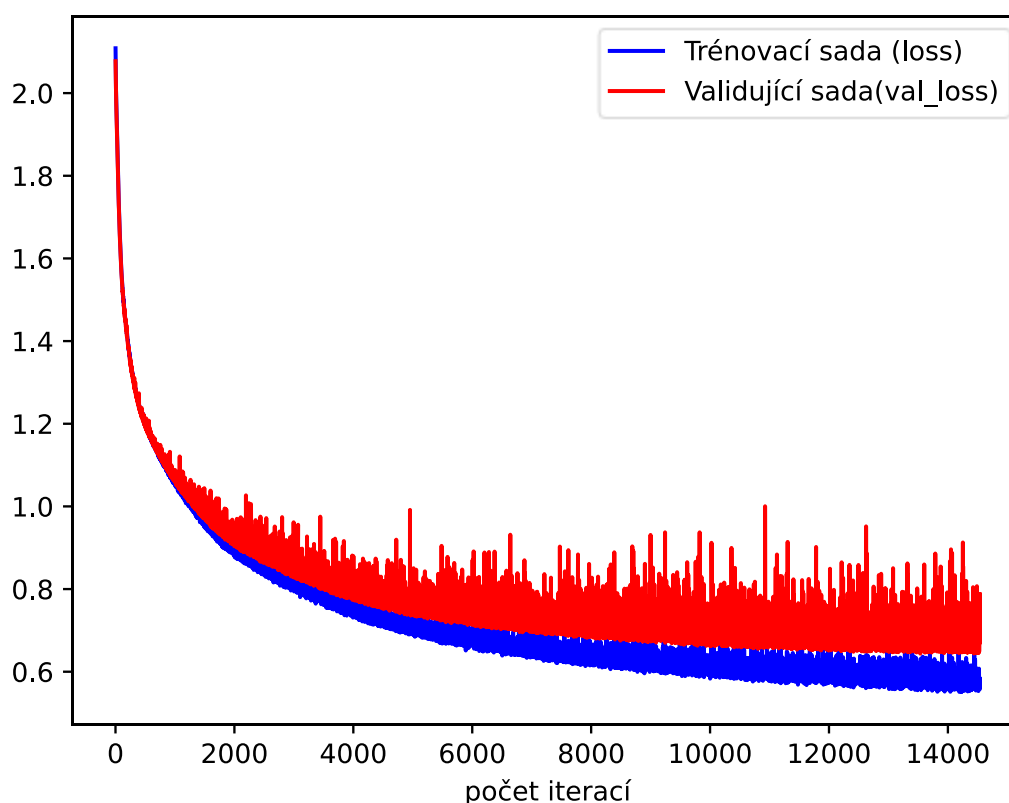
Vnitřní skryté vrstvy používají v tomto případě jako aktivační funkci hyperbolický tangens. Poslední vrstva používá funkci softmax. Tento typ aktivační funkce je výhodný pro použití ve výstupních vrstvách, neboť převádí výstupy do intervalu 0 až 1, přičemž zachovává součet celého vstupního vektoru roven 1. Z těchto důvodů lze výstup této aktivační funkce, potažmo celé vrstvy, považovat za pravděpodobnostní ohodnocení výstupních parametrů. Iniclace těchto vrstev pak byla nastavena na náhodné rozdělení s průměrem hodnot rovným nule a standardní odchylkou 0,1. S tímto rozdělením lze předpokládat, že se během učení nalezne snáze globální minimum ztrátové funkce.

Jako optimizer celého modelu je využito Adamax optimizeru, který se stará o celý proces učení. Learning rate bylo zvoleno rovno 0,00003, což způsobí pomalou konvergenci, ale zaručí, že výsledek bude přesný a při učení nebudou výsledky divergovat.

Batch hodnota, hodnota počtu vstupů, po nichž se aktualizují parametry sítě, byla nastavena na 35 a limitní počet epoch na 20 000. Vysoký počet epoch je ale ne vždy dosažen, neboť je v algoritmu nastavena zastavovací podmínka, která kontroluje přeučení sítě pomocí ztrátové funkce, a pokud se začne tato chyba zvětšovat, je proces učení zastaven a do modelu nastaveny hodnoty vah a biasů takové, jež odpovídaly minimální chybě pro validující data. To zajistí, že síť bude mít po konci učícího procesu vždy nejpřesnější výsledky, jakých při něm bylo dosaženo. Při konečném testování je pak jako správný výsledek označena ta varianta, která má nejvyšší pravděpodobnostní ohodnocení.

7.3.3 Vyhodnocení výsledků pro kompletní sadu dat

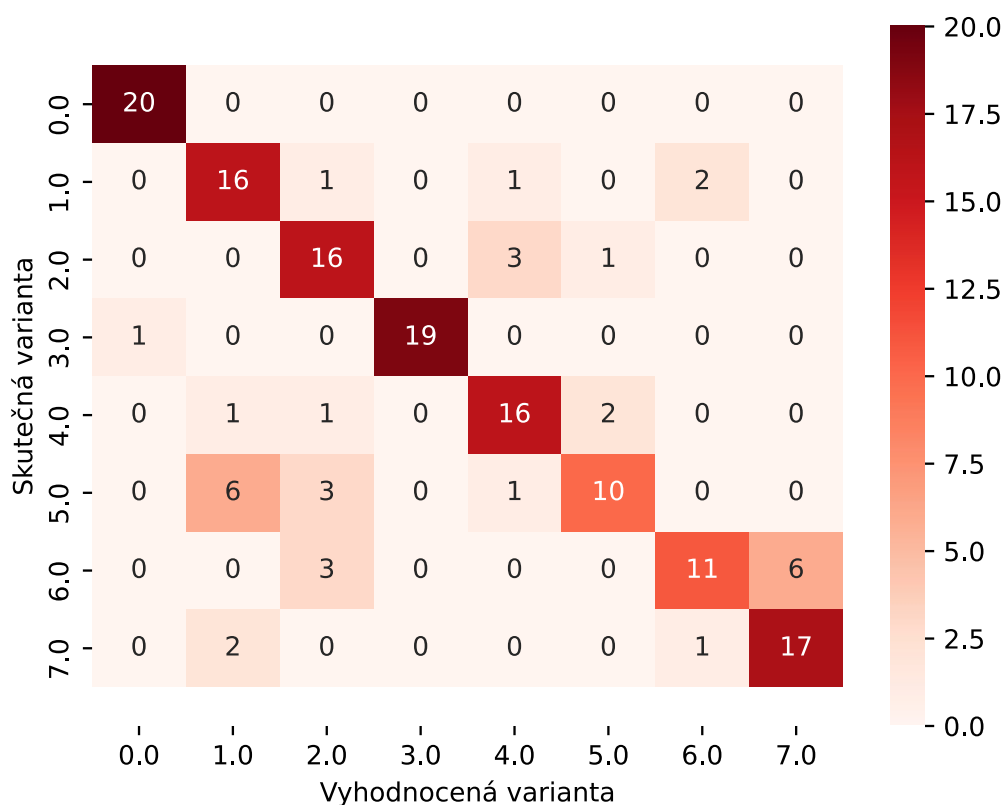
Při učení sítě na všech osm variant došlo k ukončení učícího procesu přibližně po 14 500 epochách. V tomto okamžiku je ztrátová funkce rovna přibližně 0,7, díky čemuž se výsledná síť nedá považovat za přesnou.



Obrázek 35: Vývoj ztrátové funkce během učení sítě

Schopnost správně určit konfiguraci změřeného pulzu byla po naučení testována na poslední skupině změřených pulzů, které nejsou obsaženy ani ve skupině, na které se síť

trénuje, ani ve validující sadě. Počet těchto pulzů je 20 pro každou konfiguraci. Výsledek



testování na této sadě je znázorněn na následujícím obrázku.

Z předchozího obrázku je jasně patrné, že nejsprávněji určené konfigurace jsou n0 a n3. Naopak u konfigurací n6 a n5 dochází k velké chybovosti, kdy jsou pulzy často vyhodnocovány jako pulzy konfigurace n1 a n2. Celková přesnost vyhodnocování je v tomto případě rovna 78 %, což nelze považovat za ideální.

7.3.4 Vyhodnocení výsledků pro rozdělená vstupní data

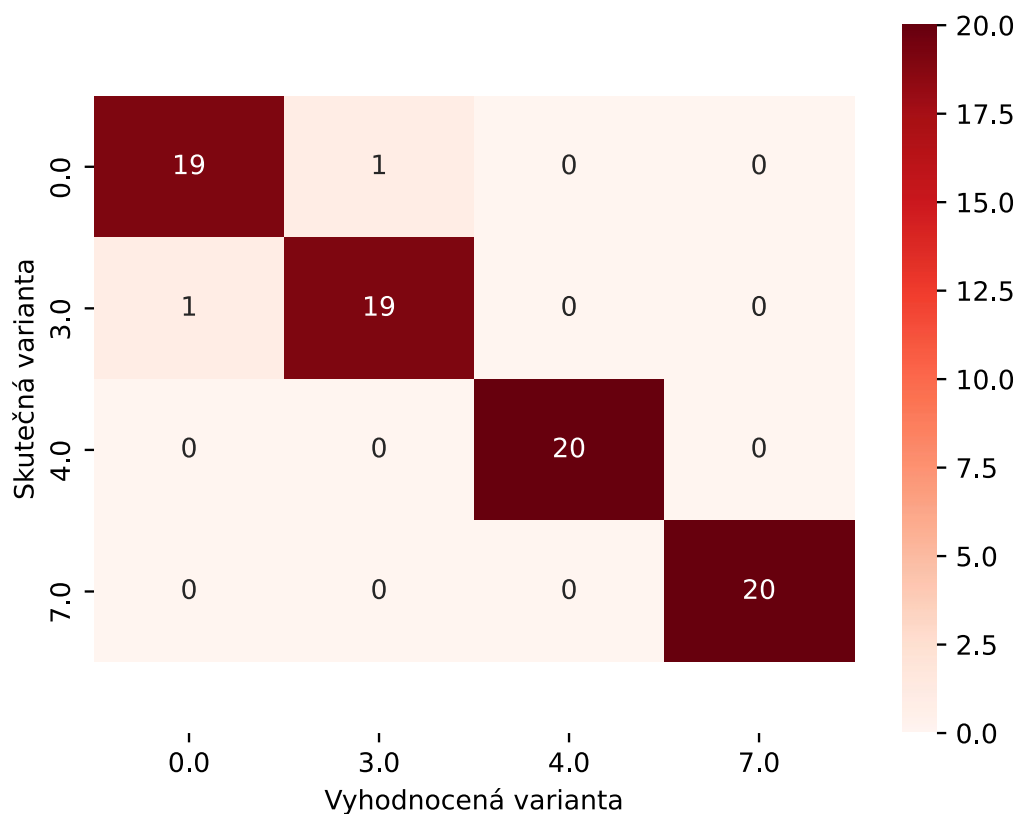
Zlepšení schopnosti klasifikace je v tomto případě možné dosáhnout například zmenšením počtu variant, které jsou vyhodnocovány. V tomto případě došlo k rozdělení podle délky stranového vysunutí elektrod na dvě skupiny. První sestává z konfigurací n0, n3, n4 a n7, druhá skupina je poté složena ze zbývajících n1, n2, n5 a n6. V každé s těchto skupin jsou tedy zastoupeny dvě hodnoty tlaků a dvě hodnoty vzdálenosti elektrod.

Předpokladem pro toto vyhodnocení je, na základě předchozího obrázku, že varianty z první skupiny budou vyhodnocovány s vysokou přesností. U druhé skupiny, která v předchozím vyhodnocení trpěla na nízkou přesnost, by však také mělo dojít k mírnému zlepšení, z důvodu méně možných řešení.

Obrázek 36: Porovnání skutečné varianty s výsledkem neuronové sítě

První skupina

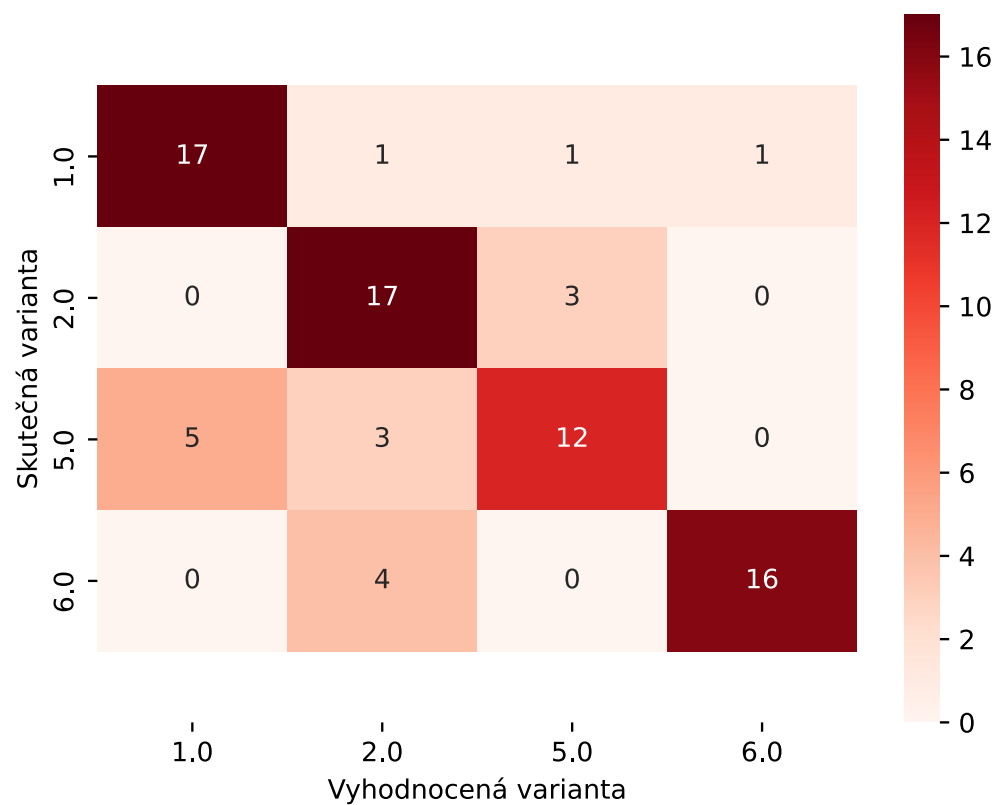
Pro vyhodnocení první skupiny byla použita stejná neuronová síť, jako byla použita při vyhodnocování všech osmi konfigurací. Celý výpočet je ukončen zhruba po 10 000 epochách, což spolu s polovičním počtem konfigurací dává přibližně čtvrtinovou dobu výpočtu sítě. Výsledky klasifikace jsou v tomto případě velmi přesné a k chybnému ohodnocení dochází pouze u dvou variant, což odpovídá přesnosti 97,5 %.



Obrázek 37: Vyhodnocení pro první skupinu

Druhá skupina

Také druhá skupina doznala zlepšení. Při vyšetřování kompletní sady změřených průběhů bylo z variant n1, n2, n5 a n6 správně vyhodnoceno na 66 % případů, zatímco při takovémto rozdělení byla přesnost 77,5 %. Trénovací proces sítě byl ukončen po přibližně 11 000 epochách. Konečné ohodnocení lze vidět na následujícím obrázku. Je stále patrné, že i zde dochází k poměrně špatnému vyhodnocení konfigurace n5, která bývá často vyhodnocována jako n1.

**Obrázek 38: Vyhodnocení pro druhou skupinu**

8. Závěr

V práci jsou shrnuty moderní možnosti rozpoznávání částečných výbojů. Tyto metody se v poslední době prosazují hlavně z důvodu nutnosti rychlé a přesné detekce činnosti částečných výbojů.

V základu jsou metody rozděleny na dvě na sebe navazující skupiny zpracování, jež na sebe navazují. V oblasti kvantifikace je kladen důraz na popis možností měření jak na střídavém napětí, kde se objevují nové algoritmy pro zpracování, tak na stejnosměrném napětí, které je perspektivou pro vysokonapěťový přenos na dlouhé vzdálenosti. Za zmínku stojí také to, že tyto kvantifikační metody se zaměřují i na více než jeden aktivně působící zdroj částečných výbojů.

U kvantifikace se klade hlavní důraz na univerzálnost metod. Jsou zde především popsány neuronové sítě, jejichž základní principy jsou sice již několik desetiletí známy, ale ve kterých stále dochází k jejich masivnímu vylepšování s tím, jak rostou výpočetní možnosti softwarového zpracování a strojového učení.

Závěrečná část se zabývá možnostmi pulse-shape analýzy, kdy pro jednotlivé pulzy částečných výbojů byly určeny charakteristické parametry. Na této množině parametrů byla poté vytrénována feed-forward neuronová síť, která prováděla klasifikaci jednotlivých variant průběhů. Výsledná neuronová síť dokázala úspěšně určovat jednotlivé varianty změřených pulzů se 78% úspěšností.

Seznam literatury a informačních zdrojů

[1] RAYMOND, Wong Jee Keen, Hazlee Azil ILLIAS, Ab Halim Abu BAKAR a Hazlie MOKHLIS. Partial discharge classifications: Review of recent progress. *Measurement*. 2015, 68, 164-181. DOI: 10.1016/j.measurement.2015.02.032. ISSN 02632241. Dostupné také z: <https://linkinghub.elsevier.com/retrieve/pii/S0263224115000901>

[2] ČSN EN 60270 Technika zkoušek vysokým napětím – Měření částečných výbojů. Praha: Český normalizační institut, 2001.

[3] PIHERA, Josef, Jaroslav HORNAK, Ales VOBORNIK, Lukas KUPKA, Svatoslav CHLADEK a Rainer HALLER. Partial Discharges Pulse Shape Analysis at AC and DC. *Proceedings of the 21st International Symposium on High Voltage Engineering*. Cham: Springer International Publishing, 2020, 2020-11-28, , 549-559. *Lecture Notes in Electrical Engineering*. DOI: 10.1007/978-3-030-31676-1_52. ISBN 978-3-030-31675-4. Dostupné také z: http://link.springer.com/10.1007/978-3-030-31676-1_52

[4] BARTNIKAS, R. Partial discharges. Their mechanism, detection and measurement. *IEEE Transactions on Dielectrics and Electrical Insulation*. 2002, 9(5), 763-808. DOI: 10.1109/TDEI.2002.1038663. ISSN 1070-9878. Dostupné také z: <http://ieeexplore.ieee.org/document/1038663/>

[5] THAYOOB, Yasmin Hanum Md, Syed Khaleel AHMED, Chua Chun PIAU, Chong Yu PING a Yogendra BALASUBRAMANIAM. Characterization of Phase Resolved Partial Discharge waveforms from instrument transformer using statistical signal processing technique. *2015 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*. IEEE, 2015, 2015, , 355-360. DOI: 10.1109/ICSIPA.2015.7412217. ISBN 978-1-4799-8996-6. Dostupné také z: <http://ieeexplore.ieee.org/document/7412217/>

[6] MORSHUIS, P.H.F. a J.J. SMIT. Partial discharges at dc voltage: their mechanism, detection and analysis. *IEEE Transactions on Dielectrics and Electrical Insulation*. 2005, 12(2), 328-340. DOI: 10.1109/TDEI.2005.1430401. ISSN 1070-9878. Dostupné také z: <http://ieeexplore.ieee.org/document/1430401/>

[7] LALITHA, E.M. a L. SATISH. Wavelet analysis for classification of multi-source PD patterns. *IEEE Transactions on Dielectrics and Electrical Insulation* [online]. 2000, 7(1), 40-47 [cit. 2020-02-04]. DOI: 10.1109/94.839339. ISSN 1070-9878. Dostupné z:

<http://ieeexplore.ieee.org/document/839339/>

[8] MA, Zhuo, Yang YANG, Martin KEARNS, Kevin COWAN, Huajie YI, Donald M. HEPBURN a Chengke ZHOU. Fractal-based autonomous partial discharge pattern recognition method for MV motors. High Voltage [online]. 2018, 3(2), 103-114 [cit. 2020-02-05]. DOI: 10.1049/hve.2017.0109. ISSN 2397-7264. Dostupné z: <https://digital-library.theiet.org/content/journals/10.1049/hve.2017.0109>

[9] Wavelet Toolbox in MATLAB — Part 1 - Ashkan Abbasi - Medium. Medium – Get smarter about what matters to you. [online]. Dostupné z: <https://medium.com/@ashkan.abbasi/wavelet-toolbox-in-matlab-part-1-dfcabc68725b>

[10] CONTIN, A., A. CAVALLINI, G.C. MONTANARI, G. PASINI a F. PULETTI. Digital detection and fuzzy classification of partial discharge signals. IEEE Transactions on Dielectrics and Electrical Insulation [online]. 2002, 9(3), 335-348 [cit. 2020-02-11]. DOI: 10.1109/TDEI.2002.1007695. ISSN 1070-9878. Dostupné z: <http://ieeexplore.ieee.org/document/1007695/>

[11] YU HAN a Y.H. SONG. Using improved self-organizing map for partial discharge diagnosis of large turbogenerators. IEEE Transactions on Energy Conversion [online]. 2003, 18(3), 392-399 [cit. 2020-02-12]. DOI: 10.1109/TEC.2003.815834. ISSN 0885-8969. Dostupné z: <http://ieeexplore.ieee.org/document/1223607/>

[12] DEY, D., B. CHATTERJEE, S. CHAKRAVORTI a S. MUNSHI. Rough-granular approach for impulse fault classification of transformers using cross-wavelet transform. IEEE Transactions on Dielectrics and Electrical Insulation [online]. 2008, 15(5), 1297-1304 [cit. 2020-02-13]. DOI: 10.1109/TDEI.2008.4656237. ISSN 1070-9878. Dostupné z: <http://ieeexplore.ieee.org/document/4656237/>

[13] RAHIMI, M. R., R. JAVADINEZHAD a Mehdi VAKILIAN. DC partial discharge characteristics for corona, surface and void discharges. In: 2015 IEEE 11th International Conference on the Properties and Applications of Dielectric Materials (ICPADM) [online]. IEEE, 2015, 2015, s. 260-263 [cit. 2020-03-08]. DOI: 10.1109/ICPADM.2015.7295258. ISBN 978-1-4799-8903-4. Dostupné z: <http://ieeexplore.ieee.org/document/7295258/>

- [14] HOOGENRAAD, G. a J. BEYER. Digital HVDC partial discharge testing. In: Conference Record of the 2000 IEEE International Symposium on Electrical Insulation (Cat. No.00CH37075) [online]. IEEE, 2000, s. 448-451 [cit. 2020-03-15]. DOI: 10.1109/ELINSL.2000.845545. ISBN 0-7803-5931-3. Dostupné z: <http://ieeexplore.ieee.org/document/845545/>
- [15] RNDr. PaedDr. Eva Volná, PhD. Neuronové sítě 1. 2008. Ostravská univerzita v Ostravě: Ostravská univerzita v Ostravě, 2008, [cit. 2020-04-10]. Dostupné z: https://www1.osu.cz/~volna/Neuronove_site_skripta.pdf
- [16] GRANT SANDERSON, Neural networks, Youtube [online], Zveřejněno: 5.10.2017, [cit. 2020-04-10]. Dostupné z: https://www.youtube.com/playlist?list=PLZHQObOWTQDNU6R1_67000Dx_ZCJB-3pi
- [17] BRAVENEC, PETR, Neuronové sítě a rozeznávání obličejů – teoretické základy, Hobrosoft.cz [online], Zveřejněno: 10.12.2019, [cit. 2020-04-10]. Dostupné z: <https://www.hobrosoft.cz/cs/blog/bravenec/neuronove-site-zaklad>
- [18] KAČER, PETR. Vícevrstvá neuronová síť. Brno, 2013. Bakalářská práce. Vysoké učení technické v Brně. Vedoucí práce Doc. Ing. VÁCLAV JIRSÍK, CSc., [cit. 2020-04-11]. Dostupné z: https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=64938
- [19] K HONG, Ph.D., ARTIFICIAL NEURAL NETWORK (ANN) 7 - OVERFITTING & REGULARIZATION, [cit. 2020-04-11]. Dostupné z: <https://www.bogotobogo.com/python/scikit-learn/Artificial-Neural-Network-ANN-7-Overfitting-Regularization.php>
- [20] ASHLIEGH, Dustin. A Guide to Understanding Partial Discharge Sensor Applications [online]. 7.11.2015 [cit. 2020-05-17]. Dostupné z: <https://www.linkedin.com/pulse/guide-understanding-partial-discharge-sensor-dustin-ashliegh>
- [21] HOEK, Stefan M a Udo RANNINGER. Practical experiences with and without the help of UHF measurement technology. Transformers Magazine [online]. 2015 [cit. 2020-05-17]. Dostupné z: <https://www.semanticscholar.org/paper/Practical-experiences-with-and-without-the-help-of-Hoek-Ranninger/3ccef8c23056c3e0982d89bcc7c994b6fef7305>

[22] SARATHI, R., Prathap SINGH a Michael DANIKAS. Characterization of partial discharges in transformer oil insulation under AC and DC voltage using acoustic emission technique. *Journal of Electrical Engineering* [online]. 2007, 58 [cit. 2020-05-17]. Dostupné z: http://iris.elf.stuba.sk/JEEEC/data/pdf/2_107-05.pdf

[23] LI, Junhao, Xutao HAN, Zehui LIU a Xiu YAO. A Novel GIS Partial Discharge Detection Sensor With Integrated Optical and UHF Methods. *IEEE Transactions on Power Delivery* [online]. 2018, 33(4), 2047-2049 [cit. 2020-05-17]. DOI: 10.1109/TPWRD.2016.2635382. ISSN 0885-8977. Dostupné z: <https://ieeexplore.ieee.org/document/7776945/>

[24] ŘEHÁK, Martin. Photomultiplier Tubes See the Light. *Photonics* [online]. 1996 [cit. 2020-05-17]. Dostupné z: <http://rayer.g6.cz/elektro/semopto/opte-pmt.htm>

[25] KHAN, Md. Tawhidul Islam. Structural Health Monitoring by Acoustic Emission Technique. WAHAB, Magd Abdel, Yun Lai ZHOU a Nuno Manuel Mendes MAIA, ed. *Structural Health Monitoring from Sensing to Processing* [online]. InTech, 2018, 2018-09-26 [cit. 2020-05-17]. DOI: 10.5772/intechopen.79483. ISBN 978-1-78923-787-0. Dostupné z: <http://www.intechopen.com/books/structural-health-monitoring-from-sensing-to-processing/structural-health-monitoring-by-acoustic-emission-technique>

[26] LOPEZ-ROLDAN, J., T. TANG a M. GASKIN. Optimisation of a sensor for onsite detection of partial discharges in power transformers by the UHF method. *IEEE Transactions on Dielectrics and Electrical Insulation* [online]. 2008, 15(6), 1634-1639 [cit. 2020-05-17]. DOI: 10.1109/TDEI.2008.4712667. ISSN 1070-9878. Dostupné z: <http://ieeexplore.ieee.org/document/4712667/>

[27] ČÁP, Martin. Detekce a prostorová lokalizace částečných výbojů ve výkonových transformátorech metodou UHF [online]. Brno, 2017 [cit. 2020-05-17]. Dostupné z: <http://hdl.handle.net/11012/65091>. Disertační práce. Vysoké učení technické v Brně. Fakulta elektrotechniky a komunikačních technologií. Ústav teoretické a experimentální elektrotechniky. Vedoucí práce Petr Drexler.

[28] PD measurements [online]. 2019 [cit. 2020-05-17]. Dostupné z: <https://www.energo-complex.pl/en/offer/pd-measurements/>

[29] Využívání vymezených rádiových kmitočtů. Český telekomunikační úřad [online]. [cit. 2020-05-17]. Dostupné z: <https://www.ctu.cz/vyuzivani-vymezenyh-radiovyh-kmitoctu>

[30] TAO, Shiyang, Fengyuan YANG, Dapeng DUAN, Wenshan WANG, Xu CHENG, Zhigang REN, Hui SONG a Jing CUI. Transient Earth Voltage detection technique for switchgears in distribution network. In: 2016 International Conference on Condition Monitoring and Diagnosis (CMD) [online]. IEEE, 2016, 2016, s. 542-545 [cit. 2020-05-17]. DOI: 10.1109/CMD.2016.7757882. ISBN 978-1-5090-3398-0. Dostupné z: <http://ieeexplore.ieee.org/document/7757882/>

[31] EATECHNOLOGY. UltraTEV. In: *Twitter* [online]. [cit. 2020-05-17]. Dostupné z: <https://twitter.com/eatechnology/status/970275276797169664>

[32] COLIN a F.C. CHEN. *Detection and Location of PD in MV Cables in Electrically Noisy Industrial Environments*. [online]. 2011 [cit. 2020-05-17]. Dostupné z: https://www.researchgate.net/publication/233612387_Detection_and_Location_of_PD_in_MV_Cables_in_Electrically_Noisy_Industrial_Environments

[33] MUHAMAD, N.A., B.T. PHUNG a T.R. BLACKBURN. Dissolved gas analysis (DGA) of partial discharge fault in bio-degradable transformer insulation oil. In: 2007 Australasian Universities Power Engineering Conference [online]. IEEE, 2007, 2007, s. 1-6 [cit. 2020-05-17]. DOI: 10.1109/AUPEC.2007.4548072. ISBN 978-0-646-49488-3. Dostupné z: <http://ieeexplore.ieee.org/document/4548072/>

[34] SAHOO, N.C., M.M.A. SALAMA a R. BARTNIKAS. Trends in partial discharge pattern classification: a survey. *IEEE Transactions on Dielectrics and Electrical Insulation* [online]. 2005, **12**(2), 248-264 [cit. 2020-06-9]. DOI: 10.1109/TDEI.2005.1430395. ISSN 1070-9878. Dostupné z:

<http://ieeexplore.ieee.org/document/1430395/>

[35] ING. ŽÁČEK, Viktor. *Kohonenova samoorganizační mapa* [online]. 2012 [cit. 2020-06-9]. Dostupné z: https://www.vutbr.cz/studenti/zav-prace?action=detail&zp_id=51742. Diplomová práce. VYSOKÉ UČENÍTECHNICKÉ V BRNĚ. Vedoucí práce Doc. Ing. Václav Jirsík, CSc.

[36] MAZROUA, A.A., R. BARTNIKAS a M.M.A. SALAMA. Neural network system using the multi-layer perceptron technique for the recognition of PD pulse shapes due to cavities and electrical trees. *IEEE Transactions on Power Delivery* [online]. **10**(1), 92-96 [cit. 2020-06-10]. DOI: 10.1109/61.368411. ISSN 08858977. Dostupné z: <http://ieeexplore.ieee.org/document/368411/>

[37] LUHRING, Ulrich, Daniel WIENOLD a Frank JENAU. Comparative investigation on pulse shape parameters of partial discharges in air under AC and DC voltage stress. In: 2016 51st International Universities Power Engineering Conference (UPEC) [online]. IEEE, 2016, 2016, s. 1-4 [cit. 2020-06-10]. DOI: 10.1109/UPEC.2016.8114056. ISBN 978-1-5090-4650-8. Dostupné z: <http://ieeexplore.ieee.org/document/8114056/>

[38] PENG, Xiaosheng, Fan YANG, Ganjun WANG, et al. A Convolutional Neural Network-Based Deep Learning Methodology for Recognition of Partial Discharge Patterns from High-Voltage Cables. *IEEE Transactions on Power Delivery* [online]. 2019, 34(4), 1460-1469 [cit. 2020-06-10]. DOI: 10.1109/TPWRD.2019.2906086. ISSN 0885-8977. Dostupné z: <https://ieeexplore.ieee.org/document/8669832/>

[39] LUHRING, Ulrich, Daniel WIENOLD a Frank JENAU. Applicability of coefficients for the defect identification of partial discharges in air under AC voltage stress based on the pulse shape. In: *2016 IEEE 16th International Conference on Environment and Electrical Engineering (EEEIC)* [online]. IEEE, 2016, 2016, s. 1-5 [cit. 2020-06-10]. DOI: 10.1109/EEEIC.2016.7555561. ISBN 978-1-5090-2320-2. Dostupné z: <http://ieeexplore.ieee.org/document/7555561/>

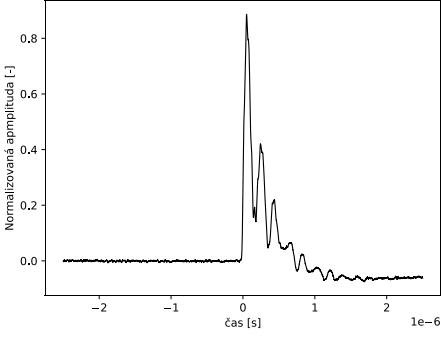
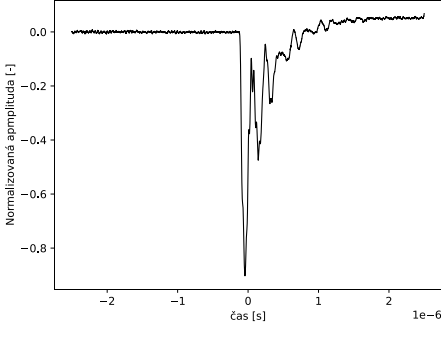
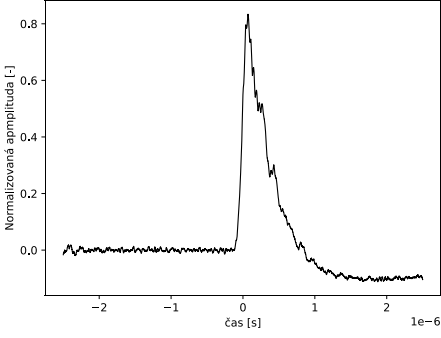
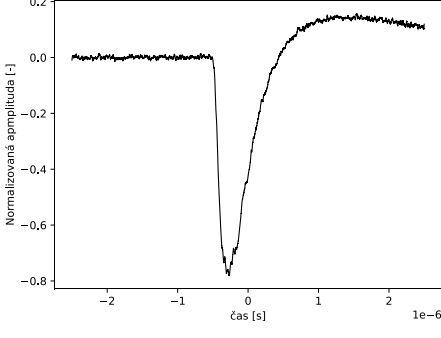
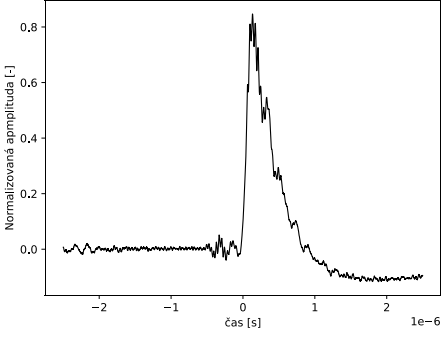
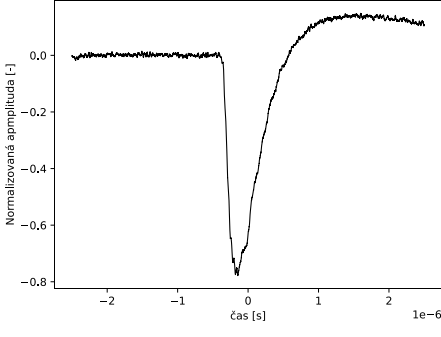
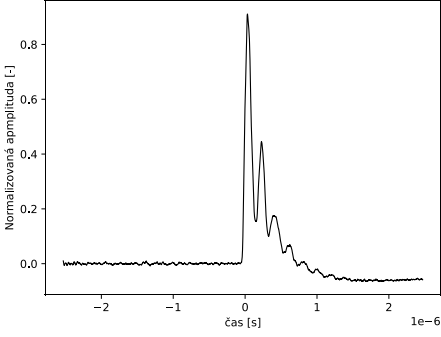
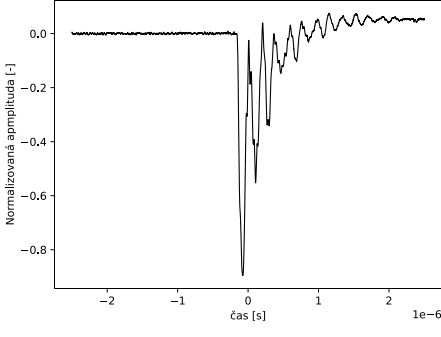
[40] Chollet, F., & others. , 2015, Keras. <https://keras.io>.

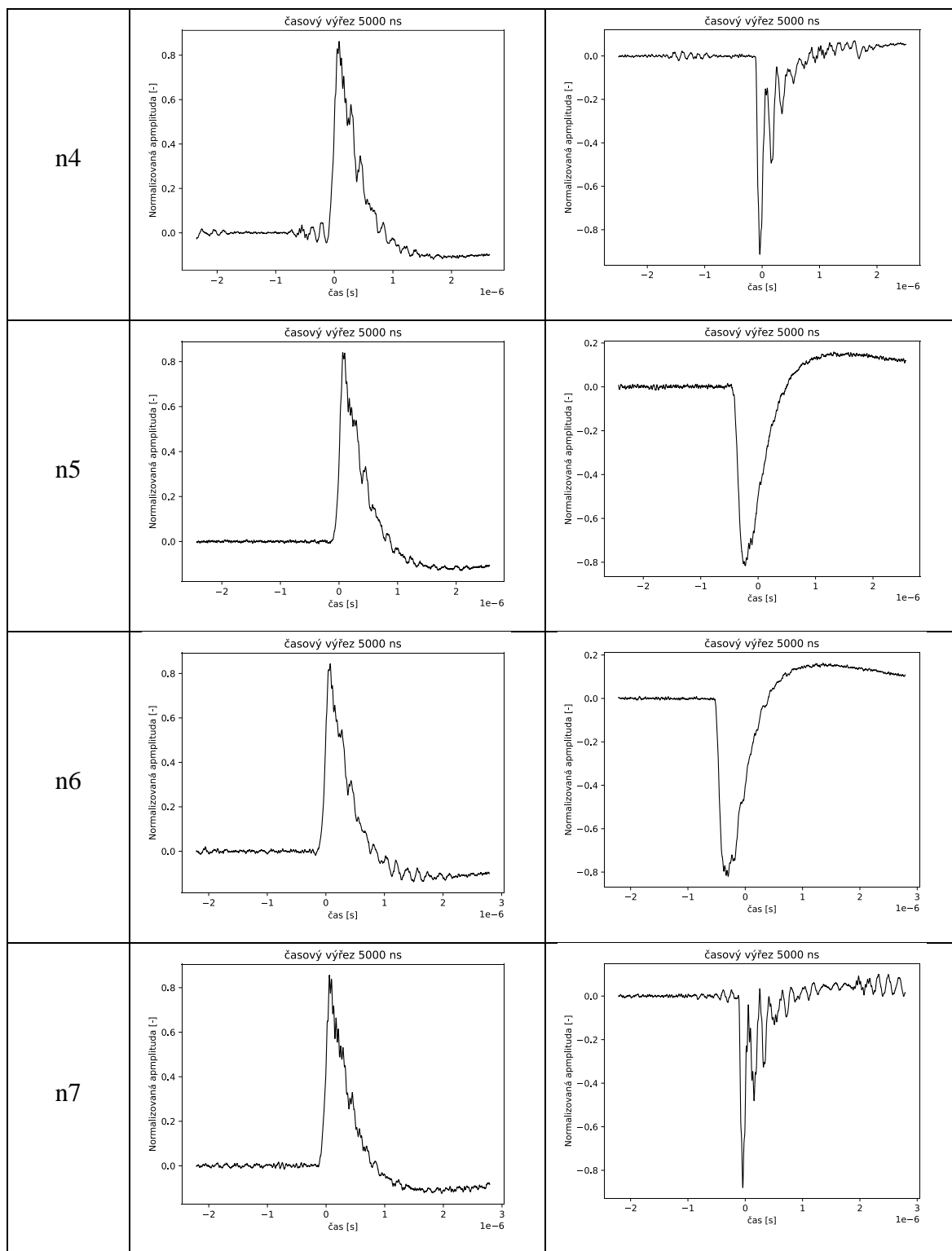
[41] WIENOLD, Daniel, Ulrich LUHRING a Frank JENAU. Detection and distinction of partial discharges in air at DC voltage by using a non-conventional approach in the high-frequency range. In: 2017 IEEE International Conference on Environment and

Electrical Engineering and 2017 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I&CPS Europe) [online]. IEEE, 2017, 2017, s. 1-5 [cit. 2020-06-17]. DOI: 10.1109/EEEIC.2017.7977477. ISBN 978-1-5386-3917-7. Dostupné z: <http://ieeexplore.ieee.org/document/7977477/>

Přílohy

Příloha 1 - Normalizované průběhy každé kategorie

kategorie	Kladný pulz	Záporný pulz
n0	<p>časový výřez 5000 ns</p> 	<p>časový výřez 5000 ns</p> 
n1	<p>časový výřez 5000 ns</p> 	<p>časový výřez 5000 ns</p> 
n2	<p>časový výřez 5000 ns</p> 	<p>časový výřez 5000 ns</p> 
n3	<p>časový výřez 5000 ns</p> 	<p>časový výřez 5000 ns</p> 



Příloha 2 - Průměrné vektory parametrů pro obě polarity každé kategorie

kat	doba náběhu [s]	doba týlu [s]	plocha	šířka 20 [s]	šířka 50 [s]	max výchylka	proložení 1	proložení 2
n0+	3.893E-08	3.643E-07	1563.485	4.580E-07	1.060E-07	1.985	-1.155	-14.576
n0-	3.990E-08	3.750E-07	1580.340	4.634E-07	2.407E-07	1.596	-0.985	-12.420
n1+	7.339E-08	3.833E-07	2673.347	5.688E-07	3.289E-07	1.031	-1.837	-23.155
n1-	7.927E-08	3.331E-07	3449.063	6.767E-07	4.552E-07	0.718	-1.704	-21.715
n2+	6.678E-08	3.834E-07	2679.228	5.620E-07	3.286E-07	1.241	-1.893	-23.822
n2-	7.941E-08	3.326E-07	3462.034	6.750E-07	4.542E-07	0.766	-1.788	-22.721
n3+	3.824E-08	2.239E-07	1399.840	3.186E-07	1.141E-07	11.547	-1.161	-14.655
n3-	3.801E-08	3.788E-07	1568.909	4.626E-07	2.492E-07	1.844	-0.938	-11.842
n4+	7.157E-08	3.776E-07	2742.945	5.579E-07	3.303E-07	1.197	-1.876	-23.639
n4-	3.742E-08	3.866E-07	1711.397	4.814E-07	2.738E-07	1.095	-1.038	-13.093
n5+	7.403E-08	3.862E-07	2654.226	5.635E-07	3.289E-07	1.287	-1.853	-23.354
n5-	7.393E-08	3.362E-07	3542.621	6.777E-07	4.548E-07	0.713	-1.774	-22.571
n6+	8.293E-08	3.783E-07	2744.768	5.715E-07	3.325E-07	0.922	-1.886	-23.779
n6-	7.194E-08	3.294E-07	3551.602	6.712E-07	4.555E-07	0.839	-1.824	-23.176
n7+	8.699E-08	3.855E-07	2767.041	5.818E-07	3.330E-07	0.853	-1.885	-23.760
n7-	3.979E-08	3.929E-07	1529.085	4.650E-07	2.517E-07	1.204	-0.974	-12.292

Příloha 3 - Prepar.py

```
#!/usr/bin/python3
import os
import matplotlib.pyplot as plt
from readTrc import Trc
import numpy as np
from numpy import trapz
import sys
trc = Trc()

def smooth(y_to_smooth, window_len):
    if window_len == 0:
        return y_to_smooth
    else:
        s1 = np.r_[2 * y_to_smooth[0] - y_to_smooth[window_len - 1::-1], y_to_smooth, 2 *
y_to_smooth[-1] - y_to_smooth[-1:-window_len:-1]]
        w1 = np.ones(window_len, 'd')
        y1 = np.convolve(w1 / w1.sum(), s1, mode='same')
        y_smoothed = y1[window_len:-window_len + 1]
        return y_smoothed

index = 'n'
window_len = 81
size = 781
classes = 8

for name in ['train', 'validation', 'test']:
    for i in range(classes):
        trida = str(i)
        path = './TRC/' + name + '/' + index + trida + '/'
        path2 = './SVG/' + name + '_dir/' + index + trida + '/'

        if not os.path.exists(path2):
            os.makedirs(path2)

        listdir = os.listdir(path)
        if '.DS_Store' in listdir:
            listdir.remove('.DS_Store')
        mylist = ""
        with open('filelist.txt', 'w', encoding='utf-8') as file:
            for eachfile in listdir:
                mylist += path + eachfile + "\n"
            file.write(mylist)

        #polarity,risetime,fallime,area,width80, width50,magnitude,curvefit
        params_out = np.zeros(shape=(len(listdir), 9), dtype='float32')
        #classes
        classes_out = np.zeros(shape=(len(listdir), 1), dtype="float32")

        for k in range(len(listdir)):
```

```

aa = path + listdir[k]
print(aa)
x52000, y52000, d = trc.open(aa) # d - dictionary with data, print(d['TIMEBASE'])
y51000 = np.delete(y52000, 50001) # y_50002 to 50001 to 50000
y50000 = np.delete(y51000, 50000)
x50000 = np.delete(x52000,[50001,50000])

y_mean10000 = np.average(y50000[0:10000])
y50000_moved = y50000 - y_mean10000

ymax, ymin = max(y50000), min(y50000) # normalization Volts axis (y) to (0, 1)
range
y50000n_offset = (y50000 - ymin) / (ymax - ymin)

y_mean10000 = np.average(y50000n_offset[0:10000])
y50000n = y50000n_offset - y_mean10000

y_to_smooth = y50000n
y_smoothed = smooth(y_to_smooth, window_len)

y1 = y_smoothed

sample_scale=0.1 # 10GS/s -> ns
polarity=0. #0 = kladný, 1 = záporný
y_smoothed_max=max(abs(y_smoothed))

if y_mean10000>0.5:
    y_smoothed=-1*y_smoothed
    polarity=1.
#pocita casy signalu z 20% na 80%
r20 = next(x50000 for x50000, val in enumerate(y_smoothed)
           if val > 0.2*y_smoothed_max)
#print ("r20: "+ str(r20))
r80 = next(x50000 for x50000, val in enumerate(y_smoothed)
           if val > 0.8*y_smoothed_max)
risetime=x50000[r80]-x50000[r20]
#print ("r80: "+ str(r80))
#print ("risetime ", risetime , " ns") #

#falltime
y_rev=y_smoothed[::-1]
x_rev=x50000[::-1]
f20 = next(x_rev for x_rev, val in enumerate(y_rev)
           if val > 0.2*y_smoothed_max)
f80 = next(x_rev for x_rev, val in enumerate(y_rev)
           if val > 0.8*y_smoothed_max)
falltime=x50000[len(x50000)-f20]-x50000[len(x50000)-f80]

mintime_index = r20
maxtime_index = len(x50000)-f20

```

```
#AREA
    area = trapz(y_smoothed[mintime_index:maxtime_index], dx=1)

#WIDTH
    width80=x50000[maxtime_index]-x50000[mintime_index]
    r50 = next(x50000 for x50000, val in enumerate(y_smoothed)
              if val > 0.5*y_smoothed_max)
    f50 = next(x_rev for x_rev, val in enumerate(y_rev)
              if val > 0.5*y_smoothed_max)
    width50 = x50000[len(x50000)-f50]-x50000[r50]

#MAGNITUDE
    mag=max(abs(y50000))

#prolozeni
    x_tofit=x50000[50000-f80:50000-f80+10000]+2.5E-6
    y_tofit=y_smoothed[50000-f80:50000-f80+10000]+1

    x_tofit_log=np.log(x_tofit)
    y_tofit_log=np.log(y_tofit)
    curve_fit = np.polyfit(x_tofit_log, y_tofit_log, 1)

    plt.plot(x50000,y1, 'k', linewidth=1)
    plt.title('časový výřez 5000 ns')
    bb = path2 + listdir[k] + '.svg'
    plt.xlabel('čas [s]')
    plt.ylabel('Normalizovaná amplituda [-]')
    plt.savefig(bb, bbox_inches='tight')
    #plt.show()
    plt.clf()

    params_out[k, 0] = polarity
    params_out[k, 1] = risetime
    params_out[k, 2] = falltime
    params_out[k, 3] = area
    params_out[k, 4] = width80
    params_out[k, 5] = width50
    params_out[k, 6] = mag
    params_out[k, 7] = curve_fit[0]
    params_out[k, 8] = curve_fit[1]

    classes_out[:, 0] = i

save_output = './out/'
if not os.path.exists(save_output):
    os.mkdir(save_output)
np.save(save_output + 'x_' + name + index + trida + '.npy', params_out)

np.save(save_output + 'y_' + name + index + trida + '.npy', classes_out)
```

```
def get_name_x(save_output, name, index, cla):
    out_name = save_output+ 'x_' + name + index + str(cla) + '.npy'
    return out_name
def get_name_y(save_output, name, index, cla):
    out_name = save_output+ 'y_' + name + index + str(cla) + '.npy'
    return out_name

if classes == 8:

    x0i = np.load(get_name_x(save_output, name, index, 0))
    x1i = np.load(get_name_x(save_output, name, index, 1))
    x2i = np.load(get_name_x(save_output, name, index, 2))
    x3i = np.load(get_name_x(save_output, name, index, 3))
    x4i = np.load(get_name_x(save_output, name, index, 4))
    x5i = np.load(get_name_x(save_output, name, index, 5))
    x6i = np.load(get_name_x(save_output, name, index, 6))
    x7i = np.load(get_name_x(save_output, name, index, 7))

    y0i = np.load(get_name_y(save_output, name, index, 0))
    y1i = np.load(get_name_y(save_output, name, index, 1))
    y2i = np.load(get_name_y(save_output, name, index, 2))
    y3i = np.load(get_name_y(save_output, name, index, 3))
    y4i = np.load(get_name_y(save_output, name, index, 4))
    y5i = np.load(get_name_y(save_output, name, index, 5))
    y6i = np.load(get_name_y(save_output, name, index, 6))
    y7i = np.load(get_name_y(save_output, name, index, 7))

    x_cnt = np.concatenate((x0i, x1i, x2i, x3i, x4i, x5i, x6i, x7i))
    np.save(save_output + 'x_' + name + '.npy', x_cnt)
    print('x_' + name + '.shape', x_cnt.shape)

    y_cnt = np.concatenate((y0i, y1i, y2i, y3i, y4i, y5i, y6i, y7i))
    np.save(save_output + 'y_' + name + '.npy', y_cnt)
    print('y_' + name + '.shape', y_cnt.shape)
```

Příloha 4 - ReadTrc.py

```
"""
```

```
Little helper class to load data from a .trc binary file.
```

```
This is the file format used by LeCroy oscilloscopes.
```

```
M. Betz 09/2015
```

```
"""
```

```
import datetime
```

```
import numpy as np
```

```
import struct
```

```
class Trc:
```

```
    _recTypes = (
```

```
        "single_sweep", "interleaved", "histogram", "graph",
```

```
        "filter_coefficient", "complex", "extrema",
```

```
        "sequence_obsolete", "centered_RIS", "peak_detect"
```

```
    )
```

```
    _processings = (
```

```
        "no_processing", "fir_filter", "interpolated", "sparsed",
```

```
        "autoscaled", "no_result", "rolling", "cumulative"
```

```
    )
```

```
    _timebases = (
```

```
        '1_ps/div', '2_ps/div', '5_ps/div', '10_ps/div', '20_ps/div',
```

```
        '50_ps/div', '100_ps/div', '200_ps/div', '500_ps/div', '1_ns/div',
```

```
        '2_ns/div', '5_ns/div', '10_ns/div', '20_ns/div', '50_ns/div',
```

```
        '100_ns/div', '200_ns/div', '500_ns/div', '1_us/div', '2_us/div',
```

```
        '5_us/div', '10_us/div', '20_us/div', '50_us/div', '100_us/div',
```

```
        '200_us/div', '500_us/div', '1_ms/div', '2_ms/div', '5_ms/div',
```

```
        '10_ms/div', '20_ms/div', '50_ms/div', '100_ms/div', '200_ms/div',
```

```
        '500_ms/div', '1_s/div', '2_s/div', '5_s/div', '10_s/div',
```

```
        '20_s/div', '50_s/div', '100_s/div', '200_s/div', '500_s/div',
```

```
        '1_ks/div', '2_ks/div', '5_ks/div', 'EXTERNAL'
```

```
    )
```

```
    _vCouplings = ('DC_50_Ohms', 'ground', 'DC_1MOhm', 'ground', 'AC_1MOhm')
```

```
    _vGains = (
```

```
        '1_uV/div', '2_uV/div', '5_uV/div', '10_uV/div', '20_uV/div',
```

```
        '50_uV/div', '100_uV/div', '200_uV/div', '500_uV/div', '1_mV/div',
```

```
        '2_mV/div', '5_mV/div', '10_mV/div', '20_mV/div', '50_mV/div',
```

```
        '100_mV/div', '200_mV/div', '500_mV/div', '1_V/div', '2_V/div',
```

```
        '5_V/div', '10_V/div', '20_V/div', '50_V/div', '100_V/div',
```

```
        '200_V/div', '500_V/div', '1_kV/div'
```

```
    )
```

```
    def __init__(self):
```

```
        """
```

```
        use trc.open(fName) to open a Le Croy .trc file
```

```
        """
```

```
        self._f = None
```

```
        # offset to start of WAVEDESC block
```

```
        self._offs = 0
```

```
self._smplFmt = "int16"
self._endi = ""
```

```
def open(self, fName):
```

```
    """
    _readS .trc binary files from LeCroy Oscilloscopes.
    Decoding is based on LECROY_2_3 template.
    [More info]
```

(http://forums.ni.com/attachments/ni/60/4652/2/LeCroyWaveformTemplate_2_3.pdf)

```
    Parameters
```

```
    -----
```

```
    fName = filename of the .trc file
```

```
    Returns
```

```
    -----
```

```
    a tuple (x, y, d)
```

```
    x: array with sample times [s],
```

```
    y: array with sample values [V],
```

```
    d: dictionary with metadata
```

```
    M. Betz 09/2015
```

```
    """
```

```
    with open(fName, "rb") as f:
```

```
        # Binary file handle
```

```
        self._f = f
```

```
        self._endi = ""
```

```
        temp = f.read(64)
```

```
        # offset to start of WAVEDESC block
```

```
        self._offs = temp.find(b'WAVEDESC')
```

```
        # -----
```

```
        # Read WAVEDESC block
```

```
        # -----
```

```
        # Template name
```

```
        self._TEMPLATE_NAME = self._readS("16s", 16)
```

```
        if self._TEMPLATE_NAME != "LECROY_2_3":
```

```
            print(
```

```
                "Warning, unsupported file template:",
```

```
                self._TEMPLATE_NAME,
```

```
                "... trying anyway"
```

```
            )
```

```
        # 16 or 8 bit sample format?
```

```
        if self._readX('H', 32):
```

```
            self._smplFmt = "int16"
```

```
        else:
```

```
            self._smplFmt = "int8"
```

```
        # Endian-ness ("<" or ">")
```

```
        if self._readX('H', 34):
```

```
            self._endi = "<"
```

```
        else:
```

```
            self._endi = ">"
```

```
# Get length of blocks and arrays
self._IWAVE_DESCRIPTOR = self._readX("I", 36)
self._USER_TEXT = self._readX("I", 40)
self._TRIGTIME_ARRAY = self._readX("I", 48)
self._IRIS_TIME_ARRAY = self._readX("I", 52)
self._IWAVE_ARRAY_1 = self._readX("I", 60)
self._IWAVE_ARRAY_2 = self._readX("I", 64)

d = dict() # Will store all the extracted Metadata

# -----
# Get Instrument info
# -----
d["INSTRUMENT_NAME"] = self._readS("16s", 76)
d["INSTRUMENT_NUMBER"] = self._readX("I", 92)
d["TRACE_LABEL"] = self._readS("16s", 96)

# -----
# Get Waveform info
# -----
d["WAVE_ARRAY_COUNT"] = self._readX("I", 116)
d["PNTS_PER_SCREEN"] = self._readX("I", 120)
d["FIRST_VALID_PNT"] = self._readX("I", 124)
d["LAST_VALID_PNT"] = self._readX("I", 128)
d["FIRST_POINT"] = self._readX("I", 132)
d["SPARSING_FACTOR"] = self._readX("I", 136)
d["SEGMENT_INDEX"] = self._readX("I", 140)
d["SUBARRAY_COUNT"] = self._readX("I", 144)
d["SWEEPS_PER_ACQ"] = self._readX("I", 148)
d["POINTS_PER_PAIR"] = self._readX("h", 152)
d["PAIR_OFFSET"] = self._readX("h", 154)
d["VERTICAL_GAIN"] = self._readX("f", 156)
d["VERTICAL_OFFSET"] = self._readX("f", 160)
# to get floating values from raw data:
# VERTICAL_GAIN * data - VERTICAL_OFFSET
d["MAX_VALUE"] = self._readX("f", 164)
d["MIN_VALUE"] = self._readX("f", 168)
d["NOMINAL_BITS"] = self._readX("h", 172)
d["NOM_SUBARRAY_COUNT"] = self._readX("h", 174)
# sampling interval for time domain waveforms
d["HORIZ_INTERVAL"] = self._readX("f", 176)
# trigger offset for the first sweep of the trigger,
# seconds between the trigger and the first data point
d["HORIZ_OFFSET"] = self._readX("d", 180)
d["PIXEL_OFFSET"] = self._readX("d", 188)
d["VERTUNIT"] = self._readS("48s", 196)
d["HORUNIT"] = self._readS("48s", 244)
d["HORIZ_UNCERTAINTY"] = self._readX("f", 292)
d["TRIGGER_TIME"] = self._getTimeStamp(296)
d["ACQ_DURATION"] = self._readX("f", 312)
```



```

d["RECORD_TYPE"] = Trc._recTypes[
    self._readX("H", 316)
]
d["PROCESSING_DONE"] = Trc._processings[
    self._readX("H", 318)
]
d["RIS_SWEEPS"] = self._readX("h", 322)
d["TIMEBASE"] = Trc._timebases[self._readX("H", 324)]
d["VERT_COUPLING"] = Trc._vCouplings[
    self._readX("H", 326)
]
d["PROBE_ATT"] = self._readX("f", 328)
d["FIXED_VERT_GAIN"] = Trc._vGains[
    self._readX("H", 332)
]
d["BANDWIDTH_LIMIT"] = bool(self._readX("H", 334))
d["VERTICAL_VERNIER"] = self._readX("f", 336)
d["ACQ_VERT_OFFSET"] = self._readX("f", 340)
d["WAVE_SOURCE"] = self._readX("H", 344)
d["USER_TEXT"] = self._readS(
    "{0}s".format(self._USER_TEXT),
    self._IWAVE_DESCRIPTOR
)

y = self._readSamples()
y = d["VERTICAL_GAIN"] * y - d["VERTICAL_OFFSET"]
x = np.arange(1, len(y) + 1, dtype=float)
x *= d["HORIZ_INTERVAL"]
x += d["HORIZ_OFFSET"]
self.f = None
self.x = x
self.y = y
self.d = d
return x, y, d

def _readX(self, fmt, adr=None):
    """ extract a byte / word / float / double from the binary file f """
    fmt = self._endi + fmt
    nBytes = struct.calcsize(fmt)
    if adr is not None:
        self._f.seek(adr + self._offs)
    s = struct.unpack(fmt, self._f.read(nBytes))
    if (type(s) == tuple):
        return s[0]
    else:
        return s

def _readS(self, fmt="16s", adr=None):
    """ read (and decode) a fixed length string """
    temp = self._readX(fmt, adr).split(b'\x00')[0]

```

```
return temp.decode()

def _readSamples(self):
    # -----
    # Get main sample data with the help of numpys .fromfile(
    # -----
    # Seek to WAVE_ARRAY_1
    self._f.seek(
        self._offs + self._IWAVE_DESCRIPTOR +
        self._USER_TEXT + self._TRIGTIME_ARRAY +
        self._IRIS_TIME_ARRAY
    )
    y = np.fromfile(self._f, self._smpFmt, self._IWAVE_ARRAY_1)
    if self._endi == ">":
        y.byteswap(True)
    return y

def _getTimeStamp(self, adr):
    """ extract a timestamp from the binary file """
    s = self._readX("d", adr)
    m = self._readX("b")
    h = self._readX("b")
    D = self._readX("b")
    M = self._readX("b")
    Y = self._readX("h")
    trigTs = datetime.datetime(
        Y, M, D, h, m, int(s), int((s - int(s)) * 1e6)
    )
    return trigTs
```

Příloha 5 - NN.py

```
#!/usr/bin/python3
from keras import layers
from keras import models
from keras import regularizers
from keras import initializers
from keras import callbacks
import keras
from keras.utils import np_utils
import matplotlib.pyplot as plt
import numpy as np
from keras.models import load_model
import time
import pandas as pd
import seaborn as sn

classes = 8
path_to = './out'

train = 'train'
validation = 'validation'
x_train1 = np.load(path_to+'x_' + train + '.npy')

y_train0 = np.load(path_to+'y_' + train + '.npy')
y_train = np_utils.to_categorical(y_train0, classes)

x_val1 = np.load(path_to+'x_' + validation + '.npy')

y_val0 = np.load(path_to+'y_' + validation + '.npy')
y_val = np_utils.to_categorical(y_val0, classes)

batch = 35
epochs1 = 20000

model = models.Sequential()
initializer = initializers.RandomNormal(mean=0., stddev=0.1, seed=19)
model.add(layers.Dense(1024, kernel_initializer=initializer, activation='tanh')) #,
kernel_initializer=initializer #2048
#model.add(layers.GaussianNoise(0.1))
#model.add(layers.Dropout(0.05))
model.add(layers.Dense(256, kernel_initializer=initializer, activation='tanh'))#zapnuto
model.add(layers.Dense(64, kernel_initializer=initializer, activation='tanh'))
model.add(layers.Dense(classes, kernel_initializer=initializer, activation='softmax'))

model.compile(optimizer=keras.optimizers.adamax(learning_rate=0.00003), #sgd slibné-
adamax #lr=0.00003
              loss='categorical_crossentropy',
              metrics=['categorical_accuracy'])

history = model.fit(x_train1,
```

```
        y_train,
        batch_size=batch,
        epochs=epochs1,
        validation_data=(x_val1, y_val),
        verbose=1,
        callbacks = [callbacks.EarlyStopping(monitor='val_loss', mode='auto',
        verbose=1, patience=1000,restore_best_weights=True)]
        )#callbacks = [callbacks.EarlyStopping(monitor='val_categorical_accuracy',
        mode='max', verbose=1, patience=100,restore_best_weights=True)]

model.save(path_to+'/NN.model')

train_acc = history.history['categorical_accuracy']
val_acc = history.history['val_categorical_accuracy']

train_loss = history.history['loss']
val_loss = history.history['val_loss']

epochs = range(1, len(train_acc) + 1)
plt.plot(epochs, train_acc, 'b', label='Trénovací sada (categorical_accuracy)')
plt.plot(epochs, val_acc, 'r', label='Validující sada (val_categorical_accuracy)')
plt.title('Přesnost')
plt.xlabel('počet iterací')
plt.legend()
axes = plt.gca()
#axes.set_ylim([0, 1])
plt.savefig('./vysledky/' + time.strftime('%Y%m%d%H%M%S') + '_acc_.svg',
bbox_inches='tight')
plt.clf()

plt.figure()
plt.plot(epochs, train_loss, 'b', label='Trénovací sada (loss)')
plt.plot(epochs, val_loss, 'r', label='Validující sada(val_loss)')
plt.xlabel('počet iterací')
plt.legend()
plt.savefig(
    './vysledky/' + time.strftime('%Y%m%d%H%M%S') + '_loss_.svg',
bbox_inches='tight')
plt.clf()
print(model.summary())

x_test1 = np.load(path_to+'/x_test.npy')
actual = np.load(path_to+'/y_test.npy')

model = load_model(path_to + '/NN.model')

a = model.predict(x_test1, batch_size=None, verbose=0)

predict = np.zeros(shape=(1, actual.size), dtype='float32')
```

```
for row in range(a.shape[0]):
    #position = np.argmax(a[row, :])
    predict[0, row] = np.argmax(a[row, :])
    #plt.bar(a[row, :], height=1, width=0.8)
plt.figure()
plt.show()
plt.clf()

predict_squeezed = np.squeeze(predict)
actual_T = np.squeeze(np.transpose(actual))
y_actu = pd.Series(actual_T, name='Skutečná varianta')
y_pred = pd.Series(predict_squeezed, name='Vyhodnocená varianta')
df_confusion = pd.crosstab(y_actu, y_pred)
plt.figure()
sn.heatmap(df_confusion, annot=True, cmap='RdPu')
# fix for mpl bug that cuts off top/bottom of seaborn viz
b, t = plt.ylim() # discover the values for bottom and top
b += 0.5 # Add 0.5 to the bottom
t -= 0.5 # Subtract 0.5 from the top
plt.ylim(b, t) # update the ylim(bottom, top) values
plt.savefig('./vysledky/' + time.strftime(
    '%Y%m%d%H%M%S') + '_matrix_.svg', bbox_inches='tight')
# plt.show()
plt.clf()
```

Příloha 6 - params.py

```
#!/usr/bin/python3
import os
import matplotlib.pyplot as plt
from readTrc import Trc
import numpy as np
from numpy import trapz
import sys
trc = Trc()
#https://github.com/yetifrisstlama/readTrc
#https://scipy-cookbook.readthedocs.io/items/SignalSmooth.html
#https://stackoverflow.com/questions/5515720/python-smooth-time-series-data
def smooth(y_to_smooth, window_len):
    if window_len == 0:
        return y_to_smooth
    else:
        s1 = np.r_[2 * y_to_smooth[0] - y_to_smooth[window_len - 1::-1], y_to_smooth, 2 *
y_to_smooth[-1] - y_to_smooth[-1:-window_len:-1]]
        w1 = np.ones(window_len, 'd')
        y1 = np.convolve(w1 / w1.sum(), s1, mode='same')
        y_smoothed = y1[window_len:-window_len + 1]
        return y_smoothed

name = 'train'
#name = 'validation'
#name = 'test'
index = 'n' # n, dp, dpr, pp, epp
window_len = 81 # if 0 - without smoothing
size = 781 # 50000, 25000, 12500, 6250, 3125, 1562, 781, 390, 195, 97, 48, 24
classes = 8 #def=8

for name in ['train', 'validation', 'test']:
    for i in range(classes):
        trida = str(i)
        path = './TRC/' + name + '/' + index + trida + '/'

        path2 = './SVG/' + name + '_dir/' + index + trida + '/'
        path3 = './'
        if not os.path.exists(path2):
            os.makedirs(path2)

        listdir = os.listdir(path)
        if '.DS_Store' in listdir:
            listdir.remove('.DS_Store')
        mylist = ""
        with open('filelist.txt', 'w', encoding='utf-8') as file:
            for eachfile in listdir:
                mylist += path + eachfile + '\n'
            file.write(mylist)
```

```

#polarity,risetime,falltime,area,width,magnitude
x_class_N = np.zeros(shape=(len(listdir), 6), dtype='float32')
#classes
y_class_N = np.zeros(shape=(len(listdir), 1), dtype="float32")

for k in range(len(listdir)):
    aa = path + listdir[k]
    print(aa, "DONE")
    x52000, y52000, d = trc.open(aa) # d - dictionary with data, print(d['TIMEBASE'])
    y51000 = np.delete(y52000, 50001) # y_50002 to 50001 to 50000
    y50000 = np.delete(y51000, 50000)
    x50000 = np.delete(x52000,[50001,50000])

    y_mean4000 = np.average(y50000[0:4000])
    y50000_moved = y50000 - y_mean4000

    ymax, ymin = max(y50000), min(y50000) # normalization Volts axis (y) to (0, 1)
range
    y50000n_offset = (y50000 - ymin) / (ymax - ymin)

    y_mean4000 = np.average(y50000n_offset[0:4000])
    y50000n = y50000n_offset - y_mean4000

    y_to_smooth = y50000n
    y_smoothed = smooth(y_to_smooth, window_len)

    y1 = y_smoothed

    sample_scale=0.1 # 10GS/s -> ns
    polarity=0. #0 = kladný, 1 = záporný
    y_smoothed_max=max(abs(y_smoothed))

    if y_mean4000>0.5:
        y_smoothed=-1*y_smoothed
        polarity=1.
#pocita casy signalu z 20% na 80%
    r20 = next(x50000 for x50000, val in enumerate(y_smoothed)
                if val > 0.2*y_smoothed_max)
    #print ("r20: "+ str(r20))
    r80 = next(x50000 for x50000, val in enumerate(y_smoothed)
                if val > 0.8*y_smoothed_max)
    risetime=x50000[r80]-x50000[r20]
    #print ("r80: "+ str(r80))
    #print ("risetime ", risetime , " ns") #

#falltime
#pocita casy signalu z 80% na 20%
    y_rev=y_smoothed[::-1]
    x_rev=x50000[::-1]
    f20 = next(x_rev for x_rev, val in enumerate(y_rev)

```

```

        if val > 0.2*y_smoothed_max)
#print ("f20: "+ str(f20))
f80 = next(x_rev for x_rev, val in enumerate(y_rev)
        if val > 0.8*y_smoothed_max)
#f80=y_rev.index(max(y_rev)) --nefunguje
falltime=x50000[len(x50000)-f20]-x50000[len(x50000)-f80]
#print ("f80: "+ str(f80))
#   print ("falltime ", falltime , " ns")

    mintime_index = r20
    maxtime_index = len(x50000)-f20

#AREA
#pokus o vypocteni plochy pod krivkou - pozor! pocita od minimalni hodnoty - sumu
#SOURCE - https://stackoverflow.com/questions/13320262/calculating-the-area-under-a-curve-given-a-set-of-coordinates-without-knowing-t
    area = trapz(y_smoothed[mintime_index:maxtime_index], dx=1)
#   print("area =", area)

#WIDTH
    width80=x50000[maxtime_index]-x50000[mintime_index]
#   print("width =", width, " ns")
    r50 = next(x50000 for x50000, val in enumerate(y_smoothed)
        if val > 0.5*y_smoothed_max)
    f50 = next(x_rev for x_rev, val in enumerate(y_rev)
        if val > 0.5*y_smoothed_max)
    width50 = x50000[len(x50000)-f50]-x50000[r50]

#MAGNITUDE
#predpokladem je mereni vzdy pri stejných uV/div
    mag=max(abs(y50000))
#   print("magnitude =", mag)

#prolozeni
    x_tofit=x50000[50000-f80:50000-f80+10000]+2.5E-6
    y_tofit=y_smoothed[50000-f80:50000-f80+10000]+1

    x_tofit_log=np.log(x_tofit)
    y_tofit_log=np.log(y_tofit)
    curve_fit = np.polyfit(x_tofit_log, y_tofit_log, 1)
    #print(curve_fit)

    f=open(path3+"params.csv", "ab") #(polarity, risetime, falltime, area, width, mag,
i)
    arr = np.array((polarity, risetime, falltime, area, width80, width50, mag, curve_fit[0],
curve_fit[1], i ))
    with open(path3+'params.csv', 'ab') as abc:
        np.savetxt(abc, [arr], delimiter=',', fmt='%0.10f')

```


Příloha 7 - plot.py

```

#!/usr/bin/python3
import os
import matplotlib.pyplot as plt
from readTrc import Trc
import numpy as np
from numpy import trapz
import sys
trc = Trc()
#https://github.com/yetifrisstlama/readTrc
#https://scipy-cookbook.readthedocs.io/items/SignalSmooth.html
#https://stackoverflow.com/questions/5515720/python-smooth-time-series-data
def smooth(y_to_smooth, window_len):
    if window_len == 0:
        return y_to_smooth
    else:
        s1 = np.r_[2 * y_to_smooth[0] - y_to_smooth[window_len - 1::-1], y_to_smooth, 2 *
y_to_smooth[-1] - y_to_smooth[-1:-window_len:-1]]
        w1 = np.ones(window_len, 'd')
        y1 = np.convolve(w1 / w1.sum(), s1, mode='same')
        y_smoothed = y1[window_len:-window_len + 1]
        return y_smoothed

name = 'train'#pro korektnost nechat train -> 140 bodu v plotu na konci
#name = 'validation'
#name = 'test'
index = 'n' # n, dp, dpr, pp, epp
window_len = 81 # if 0 - without smoothing
size = 781 # 50000, 25000, 12500, 6250, 3125, 1562, 781, 390, 195, 97, 48, 24
classes = 8 #def=8

rt=np.zeros((1120,2))

for i in range(classes):
    trida = str(i)
    path = './TRC/' + name + '/' + index + trida + '/'

    listdir = os.listdir(path)
    if '.DS_Store' in listdir:
        listdir.remove('.DS_Store')
    mylist = ""
    with open('filelist.txt', 'w', encoding='utf-8') as file:
        for eachfile in listdir:
            mylist += path + eachfile + '\n'
        file.write(mylist)

#polarity,risetime,fallime,area,width,magnitude
x_class_N = np.zeros(shape=(len(listdir), 6), dtype='float32')
#classes

```

```
y_class_N = np.zeros(shape=(len(listdir), 1), dtype="float32")

for k in range(len(listdir)):
    aa = path + listdir[k]
    print(aa, "DONE")
    x52000, y52000, d = trc.open(aa) # d - dictionary with data, print(d['TIMEBASE'])
    y51000 = np.delete(y52000, 50001) # y_50002 to 50001 to 50000
    y50000 = np.delete(y51000, 50000)
    x50000 = np.delete(x52000,[50001,50000])

    y_mean4000 = np.average(y50000[0:4000])
    y50000_moved = y50000 - y_mean4000

    ymax, ymin = max(y50000), min(y50000) # normalization Volts axis (y) to (0, 1) range
    y50000n_offset = (y50000 - ymin) / (ymax - ymin)

    y_mean4000 = np.average(y50000n_offset[0:4000])
    y50000n = y50000n_offset - y_mean4000

    y_to_smooth = y50000n
    y_smoothed = smooth(y_to_smooth, window_len)

    y1 = y_smoothed

    sample_scale=0.1 # 10GS/s -> ns
    polarity=0. #0 = kladný, 1 = záporný
    y_smoothed_max=max(abs(y_smoothed))

    if y_mean4000>0.5:
        y_smoothed=-1*y_smoothed
        polarity=1.
#pocita casy signalu z 20% na 80%
    r20 = next(x50000 for x50000, val in enumerate(y_smoothed)
               if val > 0.2*y_smoothed_max)
    #print ("r20: "+ str(r20))
    r80 = next(x50000 for x50000, val in enumerate(y_smoothed)
               if val > 0.8*y_smoothed_max)
    risetime=x50000[r80]-x50000[r20]
    #print ("r80: "+ str(r80))
    #print ("risetime ", risetime , " ns") #

#falltime
#pocita casy signalu z 80% na 20%
    y_rev=y_smoothed[::-1]
    x_rev=x50000[::-1]
    f20 = next(x_rev for x_rev, val in enumerate(y_rev)
               if val > 0.2*y_smoothed_max)
    #print ("f20: "+ str(f20))
    f80 = next(x_rev for x_rev, val in enumerate(y_rev)
               if val > 0.8*y_smoothed_max)
```

```

#f80=y_rev.index(max(y_rev)) --nefunguje
falltime=x50000[len(x50000)-f20]-x50000[len(x50000)-f80]
#print ("f80: "+ str(f80))
# print ("falltime ", falltime , " ns")

mintime_index = r20
maxtime_index = len(x50000)-f20

#AREA
#pokus o vypocteni plochy pod krivkou - pozor! pocita od minimalni hodnoty - sumu
#SOURCE - https://stackoverflow.com/questions/13320262/calculating-the-area-under-a-curve-given-a-set-of-coordinates-without-knowing-t
area = trapz(y_smoothed[mintime_index:maxtime_index], dx=1)
# print("area =", area)

#WIDTH
width80=x50000[maxtime_index]-x50000[mintime_index]
# print("width =", width, " ns")
r50 = next(x50000 for x50000, val in enumerate(y_smoothed)
           if val > 0.5*y_smoothed_max)
f50 = next(x_rev for x_rev, val in enumerate(y_rev)
           if val > 0.5*y_smoothed_max)
width50 = x50000[len(x50000)-f50]-x50000[r50]

#MAGNITUDE
#predpokladem je mereni vzdy pri stejných uV/div
mag=max(abs(y50000))
# print("magnitude =", mag)

#prolozeni
x_tofit=x50000[50000-f80:50000-f80+10000]+2.5E-6
y_tofit=y_smoothed[50000-f80:50000-f80+10000]+1

x_tofit_log=np.log(x_tofit)
y_tofit_log=np.log(y_tofit)
curve_fit = np.polyfit(x_tofit_log, y_tofit_log, 1)
#print(curve_fit)

rt[i*140+k,0]=curve_fit[0]#prepis osy
rt[i*140+k,1]=curve_fit[1]

plt.scatter(rt[0*140:1*140-1,0],rt[0*140:1*140-1,1],color='red', label='n0')
plt.scatter(rt[1*140:2*140-1,0],rt[1*140:2*140-1,1],color='yellow', label='n1')
plt.scatter(rt[2*140:3*140-1,0],rt[2*140:3*140-1,1],color='lawngreen', label='n2')
plt.scatter(rt[3*140:4*140-1,0],rt[3*140:4*140-1,1],color='forestgreen', label='n3')
plt.scatter(rt[4*140:5*140-1,0],rt[4*140:5*140-1,1],color='mediumblue', label='n4')
plt.scatter(rt[5*140:6*140-1,0],rt[5*140:6*140-1,1],color='fuchsia', label='n5')

```

```
plt.scatter(rt[6*140:7*140-1,0],rt[6*140:7*140-1,1],color='orange', label='n6')
plt.scatter(rt[7*140:8*140-1,0],rt[7*140:8*140-1,1],color='gray', label='n7')
plt.xlabel('curve_fit[0]')
plt.ylabel('curve_fit[0]')

plt.legend()
plt.savefig("./"+"curve_fit.svg", bbox_inches='tight')
#plt.title('časový výřez 5000 ns')
#bb = "./out/" + listdir[k] + '.svg'
#plt.xlabel('čas [s]')
#plt.ylabel('Normalizovaná amplituda [-]')
#plt.savefig(bb, bbox_inches='tight')
plt.show()
#plt.clf()
```