SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA CHEMICKEJ A POTRAVINÁRSKEJ TECHNOLÓGIE

EVIDENČNÉ ČÍSLO: FCHPT-113709-82054

VYUŽITIE STROJOVÉHO UČENIA A UMELEJ INTELIGENCIE NA ODHAD PROCESNÝCH VELIČÍN

BAKALÁRSKA PRÁCA

2018 **Samuel Hrstka**

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA CHEMICKEJ A POTRAVINÁRSKEJ TECHNOLÓGIE

Evidenčné číslo: FCHPT-113709-82054

VYUŽITIE STROJOVÉHO UČENIA A UMELEJ INTELIGENCIE NA ODHAD PROCESNÝCH VELIČÍN

BAKALÁRSKA PRÁCA

Študijný program: automatizácia, informatizácia a manažment v chémii a potravinárstve
Študijný odbor: 5.2.14. automatizácia, 5.2.52. priemyselné inžinierstvo
Školiace pracovisko: Ústav informatizácie, automatizácie a matematiky
Vedúci záverečnej práce/školiteľ: doc. Ing. Michal Kvasnica, PhD.

Bratislava 2018 **Samuel Hrstka**

Slovenská technická univerzita v Bratislave
Ústav informatizácie, automatizácie a matematiky

Fakulta chemickej a potravinárskej technológie
Akademický rok: 2017/2018
Evidenčné číslo: FCHPT-113709-82054

STU FCHPT

ZADANIE BAKALÁRSKEJ PRÁCE

Študent:	**Samuel Hrstka**
ID študenta:	82054
Študijný program:	automatizácia, informatizácia a manažment v chémii a potravinárstve
Kombinácia študijných odborov:	5.2.14. automatizácia, 5.2.52. priemyselné inžinierstvo
Vedúci práce:	doc. Ing. Michal Kvasnica, PhD.

Názov práce: **Využitie strojového učenia a umelej inteligencie na odhad procesných veličín**

Jazyk, v ktorom sa práca vypracuje: slovenský jazyk

Špecifikácia zadania:

Cieľom práce je navrhnúť, implementovať a experimentálne overiť automatizovaný systém na odhad procesných veličín pomocou zvukových a vibračných odtlačkov. Systém využije prvky strojového učenia a umelej inteligencie, pomocou ktorých bude zo známych zvukových a vibračných dát dedukovať ostatné procesné veličiny, napr. výkon čerpadla, otáčky motora, prietok, atď. Navrhnutý systém bude implementovaný v prostredí Matlab a bude experimentálne overený na laboratórnom procese chladiča s ventilátorom.

Rozsah práce: 30

Riešenie zadania práce od: 12. 02. 2018

Dátum odovzdania práce: 06. 05. 2018

Samuel Hrstka
študent

prof. Ing. Miroslav Fikar, DrSc.
vedúci pracoviska

prof. Ing. Miroslav Fikar, DrSc.
garant študijného programu

Abstrakt

V súčasnej dobe sme zaznamenali výrazný posun v technológiách. Výrazný pokrok zaznamenala najmä umelá inteligencia. Cieľom tejto práce je navrhnúť systém ktorý využíva prvky strojového učenia a umelej inteligencie. Takýto systém sa pokúsime implementovať v prostredí Matlab a experimentálne overiť pomocou zariadenia flexy. Pomocou flexy meriame zvukové a vibračné signály. Takéto signály prejdú reťazcom algoritmov, ktoré extrahujú užitočný signál. Na základe nameraných hodnôt fyzikálnych veličín by mal systém zostaviť tabuľku, ktorá bude obsahovať rozdiely medzi odhadovanými hodnotami a reálnymi hodnotami.

Z experimentálnej práce možno dôjsť k záveru, že systém pracuje spoľahlivo a efektívne. Miernou chybou je, že pri príliš pomalej manipulácií s potenciometrom dochádza k skresleniu signálu, čo spôsobuje výrazný nárast odchýlky, až dokým sa výkon opäť neustáli. Ďalší problém, s ktorým by sa mohol tento systém v praxi stretnúť je hluk prostredia, ktorý by svojou dynamikou presahoval užitočný signál, alebo niekoľko takýchto zariadení pracujúcich vedľa seba, ktoré by si zároveň indikovali sebe neprislúchajúce impulzy.

Výstupom tejto práce by mal byť automatizovaný systém, ktorý by dokázal dostatočne presne odhadovať veličiny na základe analýzy fyzikálnych vlastností. Tento princíp by mohol byť nie len efektívnou náhradou za finančne nákladné prietokomery, no môže slúžiť aj ako bezpečnostný systém, ktorý chráni užívateľa pred útokmi tretej strany v prostredí bezdrôtových sietí. Na základe miery odchýlky by mal byť poučený užívateľ schopný odhadnúť, či zariadenie nepracuje chybne

Kľúčové slová: analýza frekvenčného spektra; flexy; Fourierová transformácia; Feature extraction

Abstract

The contemporary advance of technology is primarily due to significant development of artificial intelligence. The aim of this work is to design a system, which uses the aspects of both machine learning and artificial intelligence. Such a system will be implemented in MATLAB environment and experimentally verified on a device called Flexy. Using aforementioned device, we are measuring sound and vibrational signals, which further undergo a series of algorithms to extract only useful parts of the signal. Based on measured values of physical parameters, the system creates a table of differences between the real values and the ones estimated by our system itself.

Concluding from the experimental part of this work, one can say that the system works reliably and effectively. However, too slow manipulation with a potentiometer causes a signal distortion that is reflected on temporary increase of amplitude, until the power reaches steady state. Another challenge to be overcome is the background noise of the environment that could overlap the desired signal source, or several of these devices working in vicinity, which could lead to indication of wrong impulses.

The outcome of this thesis is an automatized system that can estimate the parameters based on the analysis of their physical properties accurately enough. The principle can be used not only to replace expensive flowrate meters, but also as a security system against third party attacks via wireless networks. Based on the extent of deviation, a trained user should be able to detect a faulty operation of the device.

Key words: frequency spectrum analysis, Flexy, Fourier transformation, feature extraction

Obsah

Úvod

V posledných rokoch zaznamenala významný pokrok a má stále širšie využitie v rôznych oblastiach ľudskej činnosti. Počas celého 20. storočia sa vedci a inžinieri snažili vytvoriť stroje, ktoré by na základe rôznych algoritmov a modelov dokázali riešiť zložité problémy namiesto ľudí. V mnohých oblastiach umelá inteligencia dokonca prekonala ľudí, čo dokazuje napríklad víťazstvo umelej inteligencie nad šachovým veľmajstrom Gary Kasparovom, a to len vďaka základným znalostiam šachu a obrovskej rýchlosti počítania kombinácií jednotlivých ťahov. Umelá inteligencia má samozrejme uplatnenie aj v chemickom a potravinárskom priemysle, na ktoré sme sa v tejto práci zamerali.

Systém, ktorý sa pokúsime navrhnúť a dôkladne popísať zahŕňa niekoľko procesov, ktoré v tejto práci bližšie rozoberáme z hľadiska metodiky práce. Ide o signálovú cestu, ktorá obsahuje procesy ako zaznamenanie zvuku - premena akustického na elektrický signál, Fourierová transformácia – proces premeny elektrického signálu na frekvenčné spektrum , zobrazenie frekvenčného spektra – funkcia závislosti amplitúdy a frekvencie, „feature extraction“ (detekcia dát) – selektuje dáta a takto oddelí užitočný signál a šum a regresný strom – triedi hodnoty nameraného signálu do segmentov. Tento výpočtový reťazec nám pomohol zozbierať, spracovať a následne vyhodnotiť dáta, ktoré sú kľúčové pre túto prácu. Každý z týchto algoritmov sme do hĺbky rozobrali v teoretickej časti, kde sa zaoberáme princípom ich fungovania. Na záver sme dáta vyhodnotili, a zhodnotili sme, do akej miery je zariadenie efektívne a využiteľné v praxi.

Cieľom tejto práce je navrhnúť, implementovať a na záver aj experimentálne overiť systém, ktorý pomocou zvukových a vibračných odtlačkov dokáže odhadnúť procesné veličiny.

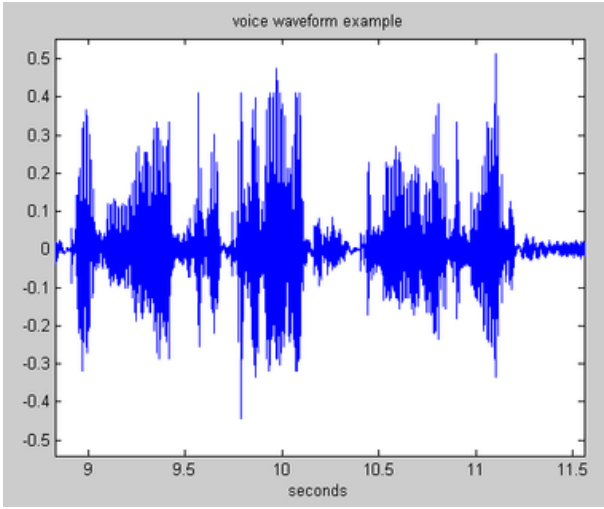
1 Teoretická časť

Základným pilierom tejto práce je zariadenie flexy (ďalej ako zariadenie). Ide o mechanizmus, ktorý pracuje s akustickým signálom. Obsahuje potenciometer, ktorým regulujeme otáčky vstavaného ventilátora. Viac sa o tomto zariadení dozvieme v stati 2.1 Zariadenie flexy. V teoretickej časti rozoberieme pojmy ako Fourierova transformácia, či „feature extraction“. Oba tieto procesy sú kľúčové pri spracovaní údajov získaných zo zariadenia flexy. Taktiež sa budeme zaoberať jednotkami amplitúdy, regresným stromom či frekvenčným spektrom.

1.1 Frekvenčná identifikácia a rýchla Fourierova transformácia

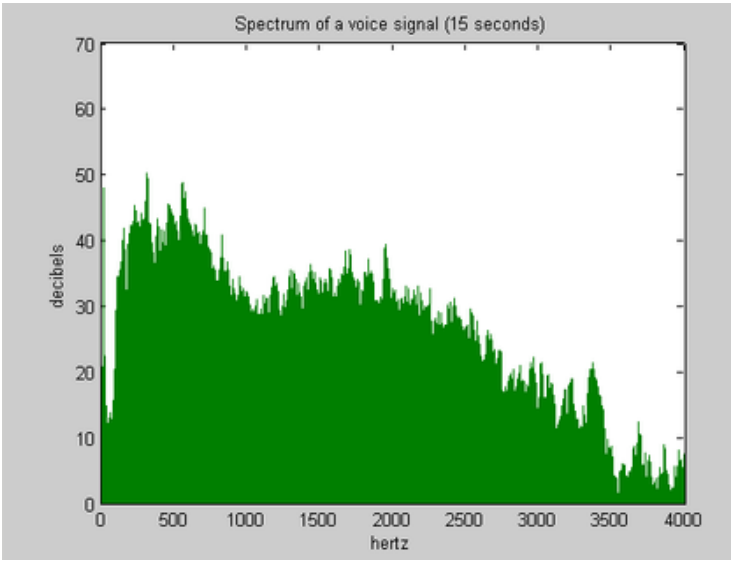
V práci budeme hodnoty procesných veličín získavať nepriamo pomocou analýzy fyzikálnych charakteristík procesu, akými sú napríklad jeho zvukové alebo vibračné odtlačky. Zvukové aj vibračné signály sa pritom vyznačujú tým, že to sú kmitavé signály, zložené z harmonických funkcií rôznych frekvencií a amplitúd. Na spätný rozklad časového signálu na jednotlivé frekvencie využijeme Fourierovu transformáciu.

Fourierova transformácia (FT) nesie meno po francúzskom vedcovi J.B.J. Fourierovi. Na začiatku 19. storočia zverejnil výsledky svojej práce, v ktorej uviedol, že každá harmonická funkcia môže byť vyjadrená, ako súčet sínusových vĺn a ich harmonických násobkov. Fourierova transformácia pracuje s časom v diskrétnej, nespojitej rovine. Vďaka FT môžme takto rozložiť vstupný signál na jednotlivé frekvencie. Oscilogram nám ukazuje priebeh zvukového signálu na grafe, na ktorom X-ová os predstavuje čas t. Vďaka FT transformujeme túto podobu zvuku do grafického zobrazenia, kde X-ová os je charakterizovaná frekvenciou f [1].



Obrázok 1 - časové zobrazenie zvukového signálu - oscilogram

Ako som už spomínal, na prevod signálu z oscilogramu, teda zloženého zvukového signálu, na jednoduchší signál nám slúži rýchla Fourierova transformácia. Táto transformácia predstavuje veľmi efektívny algoritmus, ktorý premieňa časové zobrazenie zvukového signálu na takzvané frekvenčné spektrum.



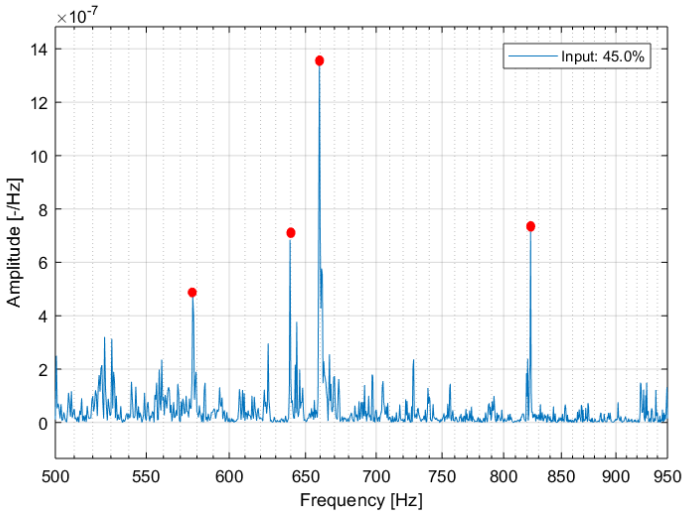
Obrázok 2 - frekvenčné zobrazenie zvukového signálu - frekvenčné spektrum

Rýchla Fourierova transformácia je veľmi dôležitá a využívaná v mnohých oblastiach, či už na digitálne spracovanie signálu, riešenie parciálnych diferenciálnych rovníc alebo napríklad aj na rýchle násobenie veľkých celých čísel. Rýchla Fourierova transformácia vytvára frekvenčné spektrum, ktoré obsahuje všetky informácie o pôvodnom signále, ale v inej forme. To znamená,

že pôvodnú funkciu možno obnoviť naspäť pomocou inverznej Fourierovej transformácie. Pre dokonalú rekonštrukciu musí spektrálny analyzátor zachovať amplitúdu aj fázu každej frekvenčnej zložky.

1.2 Frekvenčné spektrum

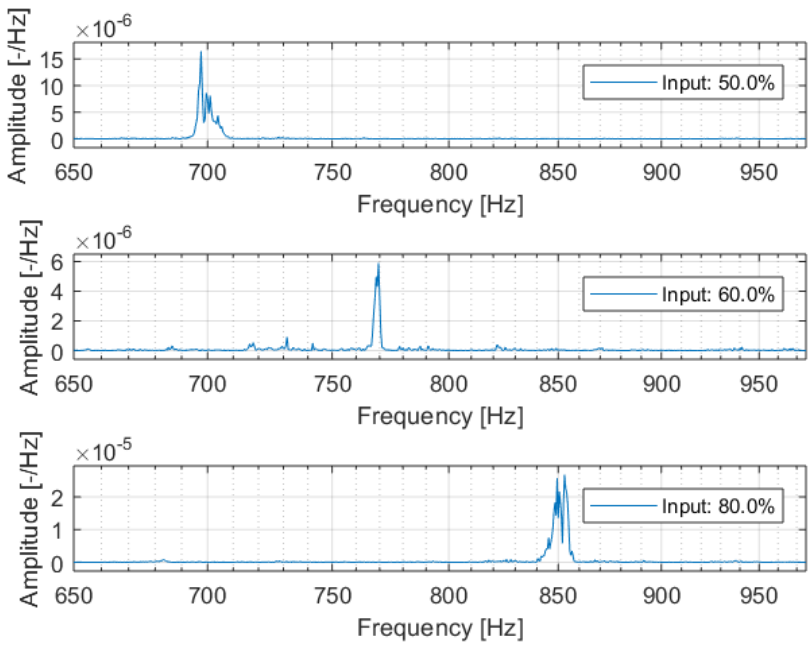
Frekvenčné spektrum je závislosť amplitúdy meranej veličiny od frekvencie a zobrazujeme ho pomocou spektrálneho analyzátora. Naše namerané vzorky už teda nie sú závislé od času, ale od frekvencie. Z frekvenčného spektra sú potom viditeľné tie frekvencie jednoduchých kmitavých pohybov, ktoré majú na výsledný zložený priebeh najväčší vplyv. Našim zámerom bude teda vyhľadať určité lokálne maximá, ktoré predstavujú významné vlastnosti danej vzorky a nazývajú sa píky.



Obrázok 3 - zobrazenie významných píkov frekvenčného spektra pri 45%-nom výkone ventilátora

Na obrázku 3 možno vidieť frekvenčné spektrum nášho zariadenia pri výkone 45% spolu s vyznačenými štyrmi hlavnými píkmi. Pre ďalší procesing je dôležitá vzdialenosť jednotlivých píkov, a odstup užitočného signálu od šumu. Môžeme si to všimnúť aj na obrázku 3 , kde sa zdá, že hneď za tretím píkom sa nachádza ešte jeden, no s veľkou pravdepodobnosťou sa jedná len o šum z predchádzajúceho píku. Na obrázku 4 možno vidieť, že pri rôznych výkonoch nášho zariadenia, sa poloha jednotlivých píkov líši, a teda jednotlivé významné vlastnosti sa nenachádzajú na rovnakej frekvencií ale sú poposúvané. Táto informácia nám indikuje, že existuje závislosť medzi výkonmi ventilátora a danými frekvenciami a má zmysel spracovať dané

údaje. Ak by sa pri rôznych výkonoch frekvencia respektíve amplitúda nemenila, nemalo by zmysel spracovávať dané údaje a celá naša procedúra by bola zbytočná.

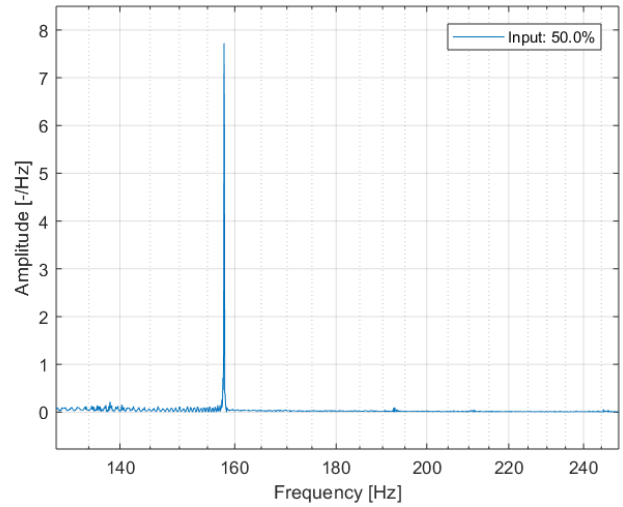


Obrázok 4 - frekvenčné spektrá pri rôznych výkonoch ventilátora

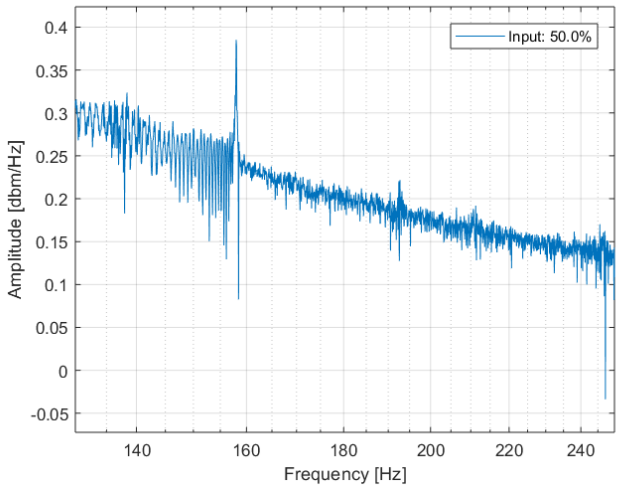
Na spracovanie daných údajov sa používajú rôzne metódy, či už pomocou regresného stromu (ku ktorému sa dostaneme v časti 1.3), klasifikačného stromu, poprípade pomocou neurónových sietí, s ktorými by sme chceli neskôr pracovať. Všetky tieto typy strojového učenia sa dajú považovať za umelú inteligenciu.

1.2.1 Možnosti vyjadrenia amplitúdy

Amplitúdu vieme vyjadriť v rôznych jednotkách, ako napríklad dBm (decibel-milliwatt), ktorý zodpovedá pomeru výkonu v decibeloch vzhľadom na jeden milliwatt, -/Hz, ktorý odstraňuje prebytočný šum okolo píku, v dB (decibel) a v mnoho ďalších jednotkách. Každá jednotka ovplyvní frekvenčné spektrum inak, avšak frekvencia významného píku ostane nezmenená. Na obrázku 5 (respektíve 6) môžeme vidieť frekvenčné spektrum pri použití jednotky -/Hz a dBm/Hz.



Obrázok 5 - frekvenčné spektrum v jednotkách amplitúdy -/Hz



Obrázok 6 - frekvenčné spektrum v jednotkách amplitúdy dBm/Hz

Ako môžeme vidieť, frekvenčné spektrá sa pri použití rôznych jednotiek amplitúdy zmenili, avšak hodnota frekvencie významného píku je rovnaká.

1.3 Detekcia vlastností – „feature extraction“

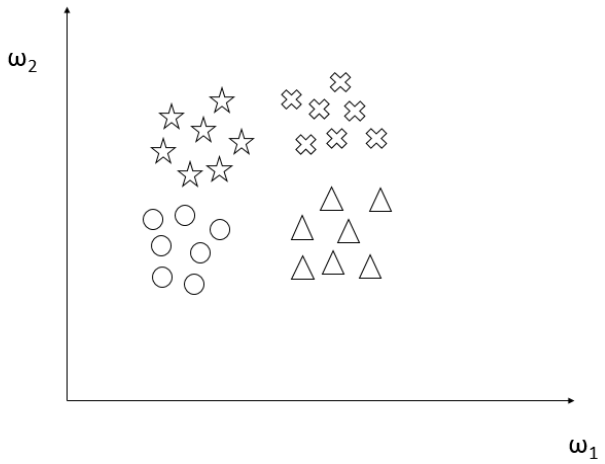
„Feature extraction“ alebo teda detekcia vlastností je proces, ktorý sa využíva vo viacerých odvetviach počítačového procesingu. Je založený na princípe detekovania výrazných merateľných prvkov a následne k ich vyhodnocovaniu. Tento proces sa využíva ako v digitalizácií grafického (obrazového), tak aj zvukového materiálu. Mnohé algoritmy sú dokonca obsiahnuté v príkazoch programátorských prostredí, ako je napríklad Matlab, v ktorom sme pracovali. Vďaka tejto detekcií dokážeme obsiahnuť väčšie množstvo informácií v menšom množstve dát. To je v dnešnej dobe informácií veľmi dôležitý atribút, nakoľko z veľkého počtu vzorkovacích dát získame iba tie najdôležitejšie a tým pádom nezahlcujeme pamäť počítača prebytočným množstvom nepotrebných dát [2].

Vďaka Fourierovej transformácii sme dostali frekvenčné spektrum. Toto frekvenčné spektrum obsahuje množstvo neužitočného signálu, ako je vlastný šum jednotlivých článkov v signálovom reťazci. Aj tieto, ale prebehli procesom digitalizácie. Ak chceme teda so signálom ďalej pracovať, je nutná selekcia dát. Feature extraction pracuje na základe redukcie počtu dimenzií, v ktorých sa kód nachádza. Nakoľko sa nám v čase menila hodnota frekvencie aj hodnota amplitúdy, a to bez danej konštantnej závislosti, extrahovali sme tieto parametre oba zvlášť [3]. Ako najefektívnejší postup sa ukázalo vyhodnocovať z ôsmich najväčších píkov pri zvuku (4 hodnoty amplitúdy a 4 hodnoty frekvencie) a 4 pri vibráciách (2 hodnoty amplitúdy a 2 hodnoty frekvencie). Pri zvolení väčšieho počtu významných píkov sa nám do vyhodnocovanie dostal neužitočný signál, ktorý zníži kvalitu výslednej regresie. Naopak, ak by sme použili príliš málo významných píkov, nemali by sme dostatok údajov na zostavenie kvalitného regresného stromu a znížila by sa opäť kvalita výslednej regresie. Na výsledkoch merania môžeme sledovať, že pri zmene výkonu sa výrazne menila ako nameraná frekvencia, tak aj hodnota amplitúdy.

1.4 Regresný strom

Pomaly sa dostávame k vytvoreniu našej procedúry, kde využívame regresný strom, akési strojové učenie, respektíve umelú inteligenciu. Regresný strom je model, kde do nášho akčného člena, v našom prípade do nášho zariadenia flexy, prichádzajú vstupné dáta vo forme sily ventilátora a odchádzajú vo forme zvukového, respektíve vibračného signálu. Tento signál sme spracovali pomocou vyššie spomínanej rýchlej Fourierovej transformácie, kde sme pri zvuku pracovali so štyrmi významnými píkmi a teda dostali sme 4 rôzne amplitúdy a frekvencie. Pri vibráciách sme pracovali iba s dvoma píkmi, nakoľko na frekvenčnom spektre vytvorenom z dát

akcelerometra boli iba 2 vyčnievajúce píky. Túto závislosť štyroch amplitúd a frekvencií sme spracovali pomocou regresného stromu. Regresný strom funguje tak, že z frekvenčného spektra zoberie nami určený počet píkov, teda hodnôt amplitúd a frekvencií a vytvorí z nich akýsi model. Pomocou tohto modelu dokážeme aktuálny zvukový signál prepočítať na výkon a porovnať ho s reálnym vstupným výkonom, pričom ak sme vytvorili vhodný model, tak reálne nastavený výkon a vypočítaný výkon by sa mali líšiť len minimálne . Ak by sme mali vhodný model a naše výkony by sa nezhodovali, mohli by sme tvrdiť, že s našim systémom niečo nie je v poriadku, nastala nejaká porucha alebo že náš systém bol napadnutý.



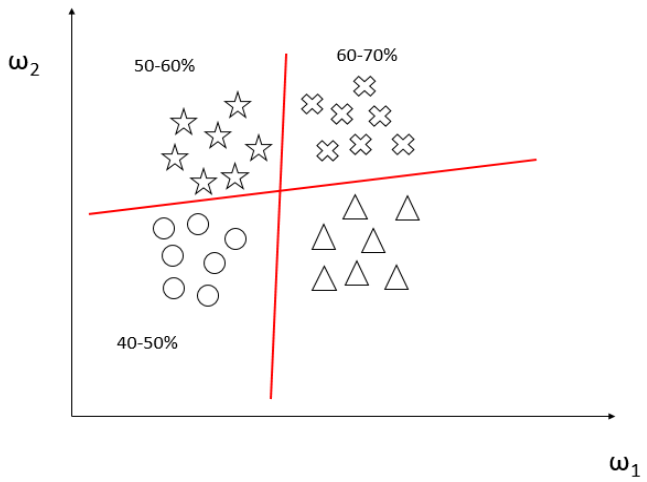
Obrázok 7 - závislosť dvoch frekvencií

Na jednoduchšie vysvetlenie fungovania takéhoto regresného stromu budeme pracovať iba v 2D, konkrétne so závislosťou dvoch zo štyroch frekvencií. Ako som už spomínal vyššie, v našej práci sme pracovali až so štyrmi píkmi a teda máme závislosť štyroch amplitúd a štyroch frekvencii, teda dokopy máme 8 závislých premenných, avšak z grafického hľadiska by to bolo oveľa zložitejšie vysvetliť, preto použije ako príklad 2D obrázok. Regresný strom funguje tak, že z frekvenčných spektier našich vzoriek pri rôznych výkonoch vyberie hodnoty jednotlivých frekvencií a amplitúd a vykreslí ich na grafe. Tieto body potom rozdelí na určité menšie segmenty, čo môžeme vidieť na obrázku číslo 8.

Obrázok 8 - rozdelenie na viacero segmentov

Toto rozdelenie vykoná pomocou väčšieho počtu čiar a vytvorí tak oddelené skupinky bodov. Jednotlivé časti potom preloží lineárnou regresiou a vytvorí tak rovnicu, pomocou ktorej dokážeme spätne vypočítať reálnu hodnotu výkonu nášho zariadenia.

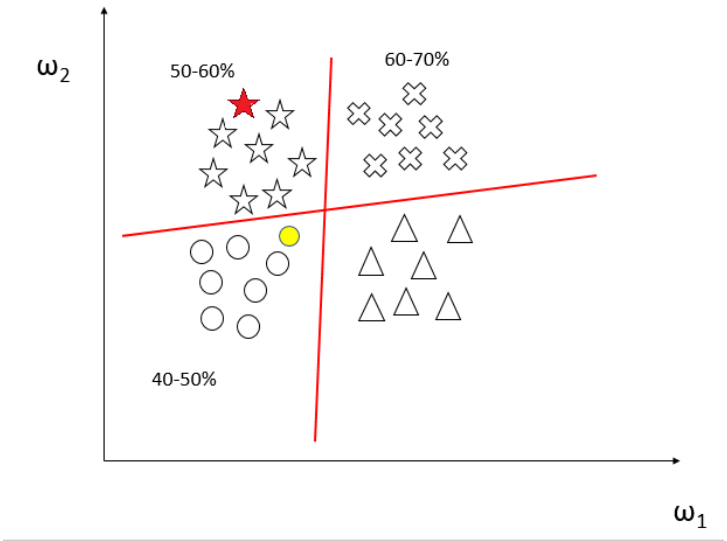
Okrem regresného stromu existuje mnoho ďalších strojových učení, napríklad klasifikačný strom, ktorý sme tak isto odskúšali. Tento typ strojového učenia funguje na trošku odlišnom princípe. Na začiatku si definujeme percentuálne rozhrania, do ktorých klasifikačný strom prerozdelí naše vzorky. Toto rozdelenie je ilustračne zobrazené na obrázku 9.



Obrázok 9 - rozdelenie na viacero segmentov

Ak už chceme otestovať model v reálnom čase, zo zvukového signálu pretransformovaného na frekvenčné spektrum, zistí hodnoty frekvencií a amplitúd významných píkov a podľa týchto hodnôt vyberie percentuálne rozhranie výkonu ventilátora. Nevýhodou tejto metódy je, že nikdy

nedostaneme číselnú hodnotu predikovaného výkonu, ale iba rozhranie v ktorom by sa mala nachádzať. Ďalšou nevýhodou je neúplná presnosť, pretože ak by sme mali ventilátor zapnutý povedzme na 49%, podľa zvukového signálu by mohol náš systém predikovať 51%, nakoľko pri týchto metódach predstavuje odchýlka niekoľko percent, a teda ako výstup by sme dostali rozhranie 50-60%. Pomocou nasledujúceho obrázka a prislúchajúceho textu si porovnáme princípy jednotlivých metód.



Obrázok 10 - zaradenie reálnych hodnôt do nášho stromu

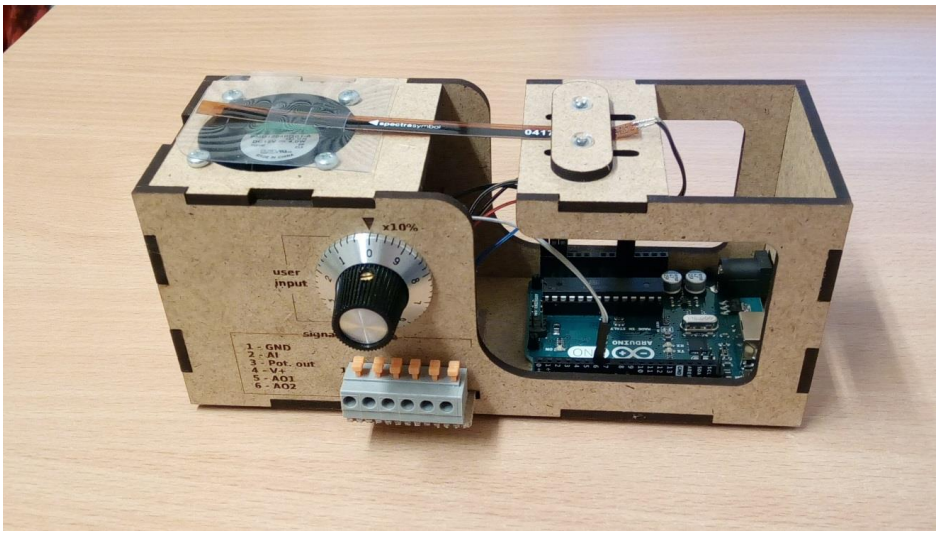
Na tomto obrázku vidíme dva farebne vyznačené body, ktoré predstavujú reálny výkon nášho zariadenia. Povedzme, že predstavujú hodnotu 49% a 55%. Naše strojové učenie si ich zaradilo do jednotlivých segmentov a ako výstup, v prípade použitia klasifikačného stromu, by boli hodnoty 40-50% respektíve 50-60%, čo by zodpovedalo reálnym hodnotám. Avšak pomocou regresného stromu, ktorý sme využili v našej práci, by sa pre jednotlivé segmenty spravila lineárna regresia, vytvorila rovnica a po dosadení do nej by sa zistilo, že prvý bod predstavuje hodnotu výkonu 47% a druhý 57%, čo sa dá považovať za vhodnejšiu metódu, než je použitie klasifikačného stromu. Odchýlky samozrejme vznikli, no v praxi predstavovali vždy menej ako 5%, čo bolo dostačujúce pre našu prácu.

2 Praktická časť

Realizácia praktickej časti prebiehala majoritne v programe Matlab. Pre spracovanie vibračných impulzov sme zvolili program Arduino IDE. Vďaka Arduino IDE sme boli schopný získať vzorky, ale ich ďalšie spracovanie prebiehalo už v spomínanom Matlabe. Prvým krokom bolo spárovanie zariadenie Flexy s počítačom. Toto sme dosiahli pomocou USB rozhrania, obsiahnutého v zariadení. Na to, aby sme dokázali ovládať naše zariadenie, bolo potrebné nainštalovať program Arduino IDE a k nemu prislúchajúce driver-e, konkrétne FTDI driver-e. Po uskutočnení jednotlivých operácií sme mohli začať pracovať na programe.

2.1 Zariadenie flexy

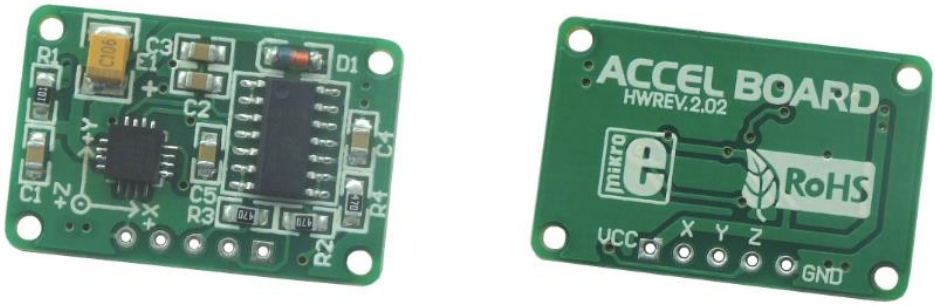
Zariadenie flexy obsahuje okrem spomínaných častí v úvode práce aj zvukový senzor, pomocou ktorého dokážeme merať zvukové signály pri rôznych výkonoch. Ďalej sa tu nachádza Arduino Uno, platforma, na ktorú sa jednoducho nahrávajú kódy a je voľne dostupné. Arduino sa programuje v jazyku C a má dve základné časti a to void setup a void loop. Do void setup-u sa nahráva kód, ktorý je spustený iba raz a do void loop-u kód, ktorý sa vykonáva nepretržite. Akcelerometer, zariadenie, ktorému sa budeme bližšie venovať v kapitole 2.1.1, sme upevnili na náš ventilátor a prepojili s touto platformou. Samozrejme všetky časti sú prepojené medzi sebou, ako môžeme vidieť na obrázku 11.



Obrázok 11 - zariadenie flexy

2.1.1 Akcelerometer

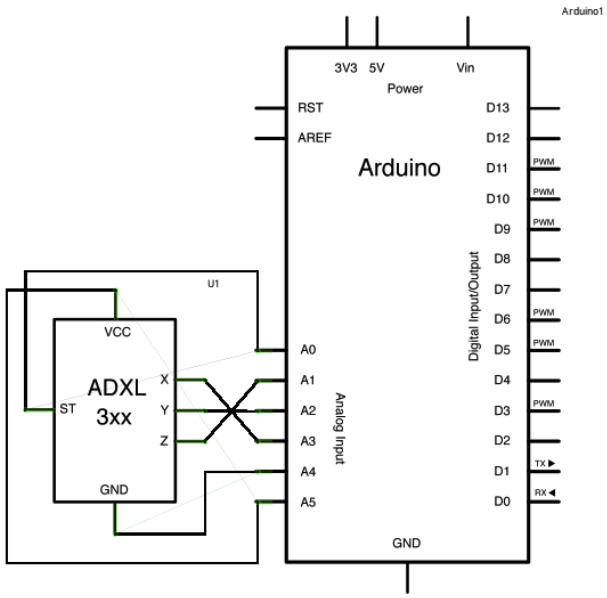
Akcelerometer je zariadenie, ktoré sa používa na meranie rôznych veličín, či už na meranie zrýchlenia, uhla vychýlenia, meranie otrasov, vibrácií alebo aj meranie samotnej gravitácie. Preto sa akcelerometre využívajú v širokej škále technických odvetví. Napríklad vysoko citlivé akcelerometre sú súčasťou navigačných systémov pre lietadlá a tak isto aj pre riadené strely. Nájdeme ich aj v automobilovom priemysle, mobilných telefónoch, herných konzolách ba dokonca aj v dronoch na stabilizáciu letu. Akcelerometre s mikroelektromechanickými systémami (MEMS) sú v súčasnosti najpoužívanejšie a sú čoraz viac prítomné v prenosných elektronických zariadeniach. Akcelerometre typu MEMS využívajú prítomnosť gravitácie a rozkladá ju do jednotlivých osí. S týmto typom akcelerometra pracujeme aj my a použijeme ho na meranie zrýchlenia v smere osi x a smere osi y. Tieto hodnoty nám predstavujú, ako veľmi vibruje náš ventilátor v smere osi x a y [4].



Obrázok 12 – akcelerometer

2.2 Meranie vibrácií

Na spracovanie vibrácií sme akcelerometer typu MEMS, opísaný v časti 2.1.1 pripojili na platformu Arduina pomocou schémy na obrázku číslo 14.



Obrázok 13 - zapojenie akcelerometra na Arduino

Ako si môžeme všimnúť, piny A1 až A3 sú prepojené s parametrami x, y a z, kde každý z nich predstavuje zrýchlenie v smere osi x, respektíve y a z. V pine A4 sa nachádza uzemnenie, v pine A5 VCC, čo predstavuje volty potrebné na prácu akcelerometra. Ako posledné tu máme ST čo znamená „self-test“ ,voľne preložené ako „samotestovanie“ , ktorý je pripojený k pinu A0 [5]. Avšak nakoľko je naša platforma prepojená s ventilátorom, niektoré piny sú už obsadené pre ventilátor a preto sme merali vibrácie iba v smere osi x a y. vibrácie v smere osi z sme teda zanedbali, aj keď ich spracovanie by prebiehalo identicky ako v ostatných smeroch. Na získanie vzoriek sme použili krátky program vytvorený v Arduine. Na začiatku sme si deklarovali všetky potrebné premenné a pridelili im hodnoty pinov, v ktorých boli zapojené.

```
const int powerpin = A0;
const int xpin = A2;
const int ypin = A1;
int powerFan = 100;
int pwmPower;
int pocitadlo = 1;
```

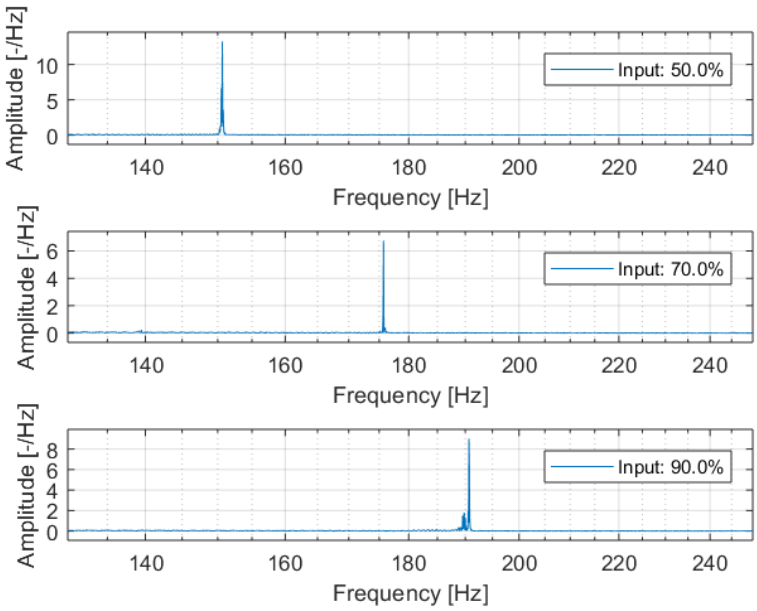
Okrem týchto premenných sme si vytvorili ďalšie premenné „powerFan“, ktorá predstavuje silu ventilátora v percentách(v našom prípade je ventilátor spustený na 100%) a „počitadlo“, ktorá nám slúžila na nameria požadovaného množstva vzoriek.

```
void setup() {
  Serial.begin(115200);
  pinMode(powerpin, OUTPUT);
  digitalWrite(powerpin, HIGH);
```

V časti void setup sa nachádza časť kódu, ktorá sa spustí iba raz. Táto časť slúži na to, aby sme dostali výstupné informácie v nami požadovanej forme.

```
void loop() {
 if(pocitadlo<10000){
   pwmPower = round(255*powerFan/100);
   analogWrite(6, pwmPower);
   Serial.print(analogRead(xpin));
   Serial.print("\t");
   Serial.print(analogRead(ypin));
   Serial.println("\t");
   delay(1);
   pocitadlo = pocitadlo+1;
 }}
```

Časť kódu nachádzajúca sa v časti void loop je najdôležitejšia a vykonáva sa nepretržite. Vďaka tejto časti nám program zaznamenával hodnoty zrýchlení v smere osi x a aj y až do okamihu, keď premenná počítadlo nedosiahla hodnotu 10 000. Z nameraných hodnôt zrýchlení sme pomocou Fourierovej transformácie získali frekvenčné spektrum. Na obrázku 15 možno vidieť porovnanie týchto spektier pri rôznych výkonoch ventilátora. Môžeme si všimnúť, že hodnoty frekvencií a amplitúd jednotlivých významných píkov je odlišná, takže existuje závislosť medzi výkonmi ventilátora a danými frekvenciami a amplitúdami.



Obrázok 14 - frekvenčné spektrá pri rôznych výkonoch ventilátora

2.3 Meranie zvuku

Na to, aby sme dokázali merať zvukovú stopu sme si museli inicializovať niektoré premenné. Prvým krokom bolo určenie takzvaného samplingRatu, čo predstavuje vzorkovaciu frekvenciu, teda koľkokrát za 1 sekundu prevádzame naše meranie. Ďalej sme si definovali premennú nBits, čo udáva počet alebo veľkosť amplitúdy, pre 8 bitové rozlíšenie poskytuje 256 možných intervalov, 16 bitové niečo viac ako 65 000 možných intervalov. Po inicializácií sme pripojili naše flexy k Matlabu.

```
%% initialization
samplingRate = 44100
nBits = 16;
nChannels = 1;
recObj = audiorecorder(samplingRate, nBits, nChannels);
f = Flexy('COM3');
```

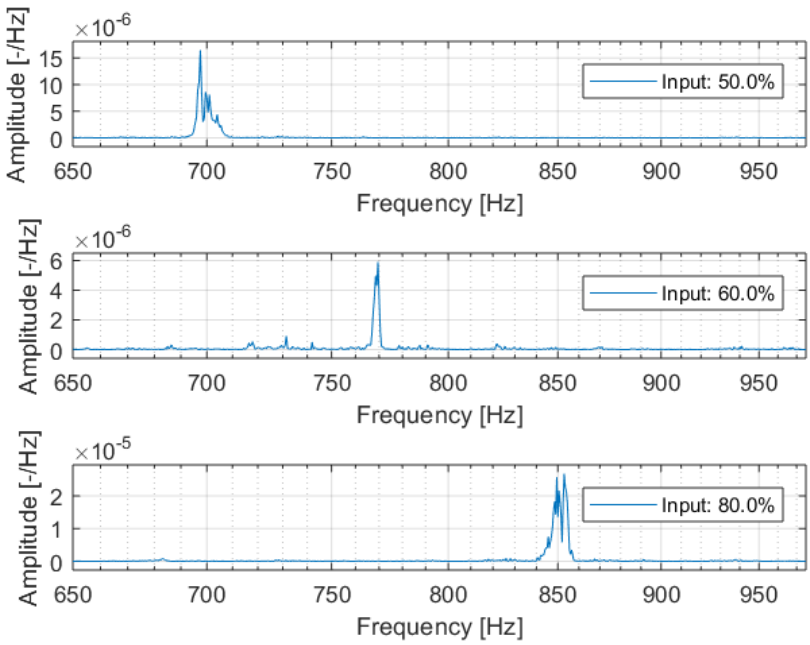
2.3.1 Program na zber dát

Nasledujúca časť nášho programu spočívala v zbieraní dát a ich úprave na požadovaný formát. V prvej časti sme si určili koľko vzoriek chceme nazbierať, čo sme inicializovali pod premennou n, a takisto sme si určili, v akom rozsahu chceme tieto vzorky zbierať. Vzorky s

výkonom pod 30% sme nebrali do úvahy, nakoľko zvuková stopa ventilátora pri 30%-nom výkone bola veľmi slabá a mohla by byť výrazne ovplyvnená šumom z okolia. Ďalej už prebiehala nahrávka vzoriek. Každá jedna vzorka sa nahrávala 2 sekundy, na monitor nám vypísalo ktorá vzorka sa práve nahráva a pri akom výkone sa nahráva. Po tomto úkone sa spraví frekvenčné spektrum, kde ako vstup prichádza daný zvukový signál, na začiatku tejto časti spomínaný samplingRate a výkon, pri ktorom bola daná vzorka nahraná. Po nahraní všetkých vzoriek sme si jednotlivé spektrá uložili do premennej soundBatch a mohli sme pokračovať v našej procedúre. Samozrejme počet nahrávok sa dal ľubovoľne určiť, čím sme mali väčší počet vzoriek, tým sme vytvorili presnejšie strojové učenie.

```
%% data acquisition
n=200;power_range=round(rand(1,1000)*100);
power_range(power_range<30)=[]; power_range = power_range(1:n);
pause(10);
soundBatch = [];
duration = 2;
   if isequal(f.SerialStatus, 'Closed')
      f.connect();
   end
   for i = 1:numel(power_range)
      power = power_range(i);
      fprintf('Scenario%d/%d,power:%d%%\n',i,numel(power_range),
power);
      f.setFanSpeedPerc(power);
      pause(0.5);
      recObj.recordblocking(duration);
      soundData = getaudiodata(recObj);
      fs = FrequencySpectrum(soundData, samplingRate, power);
      soundBatch = [soundBatch, fs];
   end
f.setFanSpeedPerc(0);
f.close();
soundBatch.compute('units', '-/Hz');
save soundBatch soundBatch
```

Tak isto ako pri vibráciách, aj pri zvukových signáloch sa hodnoty frekvencií a amplitúd jednotlivých významných píkov líšia, čo nám dokazuje porovnanie spektier na obrázku 16.



Obrázok 15s - frekvenčné spektrá pri rôznych výkonoch ventilátora

2.4 Učenie regresného stromu

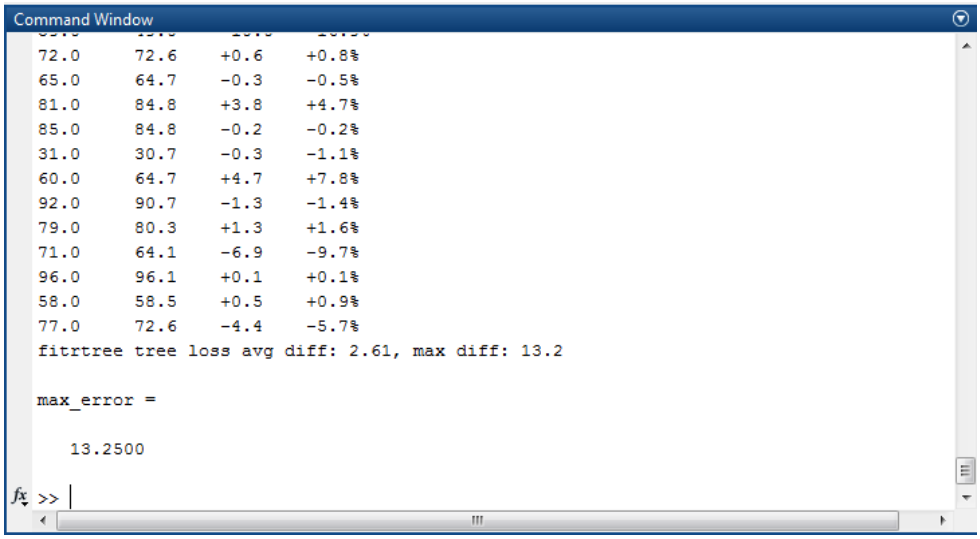
Učenie regresného stromu by sa dalo považovať za najdôležitejšiu časť tejto práce. V tejto časti sme sa pokúšali z grafov frekvenčných spektier nameraných vzoriek vhodne zvoliť dôležité parametre ako frekvenčný rozsah, počet významných píkov a aj vzdialenosť medzi nimi. Na overenie, či sme zvolili vhodné parametre nám slúžila ďalšia časť kódu, kde sa 70% vzoriek použilo na vytvorenie skúšobného regresného stromu.

```
%% classification/regression
tfun = @fitrtree;
fmin = 500;
fmax = 970;
nPeaks = 4; MinPeakDistance = 50;
tmin = 0;
tmax = soundBatch(1).Duration;
p_training = 0.7;
soundBatch.compute('units', '-/Hz');
[pf, pa] = soundBatch.peaks(nPeaks, 'fmin', fmin, 'fmax', fmax,
'MinPeakDistance', MinPeakDistance);
features = [pf, pa];
responses = cat(1, soundBatch.Input);
```

Zvyšných 30% vzoriek sme aplikovali na náš vytvorený skúšobný regresný strom a zisťovali sme, aké veľké odchýlky má. Na toto aplikovanie sme použili funkciu flexy_training_data.

```
tdata = flexy_training_data(features, responses, p_training, tfun);
[tree, max_error] = flexy_learn(tdata); max_error
```

Výstupom z tejto funkcie sú 4 stĺpce, kde prvý znázorňuje reálne hodnotu výkonu zariadenia, druhý znázorňuje hodnotu, ktorú vypočítal náš regresný strom, tretí znázorňuje odchýlku medzi nimi a posledný znázorňuje odchýlku v percentách. Na záver nám vyhodnotí a vypočíta aj priemernú hodnotu odchýlky a takisto aj najvyššiu hodnotu odchýlky, ktorá sa vyskytla pri testovaní nášho skúšobného stromu.



Obrázok 16 - test regresného stromu

Ako môžeme vidieť na obrázku 16, priemerná hodnota odchýlka bola iba 2,61 a maximálna bola 13,2, pričom väčšia odchýlka bola spôsobená kvôli šumom z okolia, ktoré mohli počas nahrávania vzorku poškodiť.

2.5 Testovanie

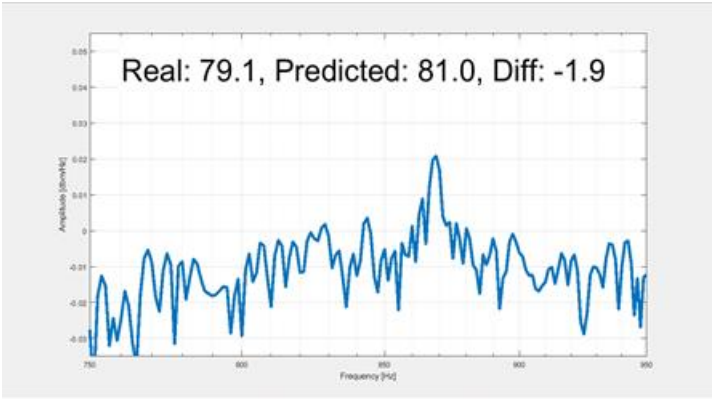
Táto časť praktickej časti spočíva v testovaní nášho vytvoreného a už aj odskúšaného stromu v praxi. Na začiatku si otestujeme strom vytvorený zo zvukových signálov a neskôr sa dostaneme aj k testovaniu stromu vytvoreného z vibrácií.

2.5.1 Testovanie zvuku

Predposledná časť nášho kódu spočíva teda v testovaní, ktoré realizujeme volaním funkcie flexy_test.

```
%% testing
test_duration = 0.2;
flexy_test(tree, test_duration, nPeaks, MinPeakDistance, fmin, fmax);
```

Tá sa na začiatku spáruje s našim zariadením a zaznamenáva jeho zvukové odtlačky. Následne tento signál spracuje rovnakým spôsobom ako keď sme vytvárali náš regresný strom a vytvorí mu frekvenčné spektrum, ktorého píky porovnáva s nami vytvorením stromom. Konečným výstupom tejto funkcie je zobrazenie grafu, kde okrem reálne nastaveného výkonu možno vidieť aj výkon, ktorý predikuje naše strojové učenie a takisto aj odchýlku, o koľko sa naše strojové učenie mýlilo. Zároveň zobrazujeme aj frekvenčné spektrum v danom okamihu.



Obrázok 17 - testovanie strojového učenia v praxi

Posledná časť nášho kódu slúži na zhodnotenie výsledkov, respektíve na porovnávanie jednotlivých spektier a tým pádom aj na vybratie dobrých hodnôt našich dôležitých parametrov.

```
%% process results
if false
    close all
     soundBatch.compute('units', '-/Hz');
     figure;    soundBatch.filter(70,    72).plot('fmin',    500,
    'fmax', 950)
     figure;    soundBatch.filter(50,    52).plot('fmin',    500,
    'fmax', 950)
end
```

V našom prípade to bolo vykreslenie pre výkon 70-72% respektíve 50-52%, pričom sme si zvolili aj vhodnú jednotku na vykreslenie amplitúdy.

2.5.2 Testovanie vibrácií

Testovanie vibrácií nebolo také jednoznačné ako testovanie zvuku. Dáta, ktoré sme namerali pomocou programu v Arduine spomínanom v časti 2.2, sme si uložili a vložili do Matlabu. Učenie regresného stromu a jeho testovanie sme realizovali rovnako ako pri zvuku, samozrejme pri rozdielnych nastaveniach významných vlastností, kde sme brali do úvahy iba 2 významné píky, pričom sa zmenil aj rozsah frekvencie a aj minimálna vzdialenosť píkov. Problém nastal pri testovaní nášho stromu v reálnom čase. Komunikácia Matlab-Arduino-akcelerometer nebola dostatočne rýchla, nakoľko maximálny počet vibračných signálov, ktoré sme dokázali poslať z akcelerometra do Matlabu bol 255 dát za sekundu. Pri takejto frekvencií vzorkovania sme nedokázali zostrojiť kvalitné frekvenčné spektrum, ktoré by sme porovnávali s našim stromom. Pri komunikácií Arduino-akcelerometer bola naša frekvencia vzorkovania vyše 650 vzoriek za sekundu. Preto bolo meranie v reálnom čase a porovnávanie s našim stromom nemožné a testovanie sme teda prevádzali „offline“. Toto testovanie sme uskutočnili pomocou nasledujúceho kódu v Matlabe, pričom najskôr sme museli namerať dáta v Arduine, uložiť a načítať v Matlabe.

```
load data.mat
pow = 50;
x50 = FrequencySpectrum(data(:,1), size(data, 1)/T, pow);
bx = [x50];
bx.compute('units', '-/Hz');
[pq, pw] = bxa.peaks(nPeaks, 'fmin', fmin, 'fmax', fmax,
'MinPeakDistance', MinPeakDistance);
features = [pq pw];
power_pred = predict(tree, features);
fprintf('Actual: %.1f, Predicted: %.1f, Diff: %.1f\n', ...
       pow, power_pred, abs(pow-power_pred));
```

Data získané z Arduina si načítame v Matlabe a pokračujeme tak ako doteraz vždy. Zo získaných dát sa vytvorí frekvenčné spektrum, nastavia sa jednotky v ktorých ho chceme zobraziť. Následne z frekvenčného spektra získame hodnoty významných píkov a porovnáme ich s našim stromom. Ako výstup je opäť hodnota aktuálneho výkonu, výkon, ktorý predikuje náš strom a rozdiel medzi nimi.

3 Výsledky

Výsledky našej experimentálnej časti pri práci so zvukovým signál môžeme vidieť v tabuľke 1. V prvom stĺpci možno vidieť číslo prebiehajúceho merania, v druhom stĺpci môžeme vidieť reálnu hodnotu výkonu nášho zariadenia, v treťom predikovanú hodnotu výkonu, ktorú vypočítal náš regresný strom a v štvrtom stĺpci rozdiel medzi jednotlivými výkonmi. Tieto výsledky sme dosiahli postupnými krokmi opísanými v praktickej časti. Na začiatku sme namerali vzorky zvukových signálov pri rôznych výkonoch a vytvorili sme z nich frekvenčné spektrum. Pomocou jednotlivých spektier sme vytvorili regresný strom, ktorý sme zároveň aj otestovali. Po skompletizovaní regresného stromu sme ho otestovali v reálnom čase, kde sme zvukový signál zaznamenaný pri ľubovoľne sa meniacich výkonoch ventilátora porovnávali s regresným stromom a vypisovali výsledky vo forme tabuľky.

Tabuľka 1 - výsledky testovania nášho systému pri práci so zvukovým signálom

Číslo merania	Reálny výkon[%]	Odhadovaný výkon[%]	Rozdiel výkonov[-]
1	69,2	73,6	4,4
2	69,2	73,6	4,4
3	69,2	73,6	4,4
4	75,4	80,1	4,7
5	75,5	80,1	4,6
6	75,6	80,1	4,5
7	77,3	90,1	12,8
8	80,7	87,0	6,3
9	82,2	87,0	4,8
10	82,2	87,0	4,8
11	82,2	87,0	4,8
12	86,7	93,3	6,6
13	86,7	93,3	6,5
14	86,7	93,3	6,5
15	89,3	69,3	-20,1
16	92,2	96,2	4,0
17	94,0	96,2	2,2
18	94,2	96,2	2,0

Nakoľko vyhodnocovanie odchýlok prebieha nepretržite, pri prechode z jedného požadovaného výkonu na ďalší zaznamená a vyhodnotí aj meranie nachádzajúce sa medzi týmito výkonmi. Toto meranie je neustálené a zvuková stopa môže byť skreslená. Tento jav nám nastal pri meraní číslo 7 a 15, kde odchýlka výrazne vyskočila, no pri ďalšom meraní sa opäť ustálila.

Výsledky experimentálnej časti pri práci s vibračným signálom sme zobrazili v tabuľke 2. Táto tabuľka sa skladá z rovnakých častí ako tabuľka 1, teda číslo merania, reálna hodnota výkonu, predikovaná hodnota výkonu a odchýlka medzi nimi. Toto meranie neprebieha v reálnom čase, takže nenastávajú veľké odchýlky pri prechode medzi jednotlivými výkonmi ako v prípade vyhodnocovania zvukových signálov.

Tabuľka 2 - výsledky testovania nášho systému pri práci s vibračným signálom

Číslo merania	Reálny výkon[%]	Odhadovaný výkon[%]	Rozdiel výkonov[-]
1	50,0	54,4	4,4
2	58,0	57,0	-1,0
3	67,0	69,0	2,0
4	72,0	75,5	3,5
5	77,0	75,5	1,5
6	80,0	80,3	0,3
7	87,0	91,8	4,8
8	94,0	92,0	-2,0
9	100,0	98,3	1,8

Pri oboch testovaniach našich strojových učení sme zistili, že odhady výkonov nášho ventilátora sa približujú k reálnym hodnotám výkonu a líšia sa len mierne. Strojové učenie založené na porovnávaní zvukového signálu dokáže pracovať nepretržite a reagovať na zmeny výkonu ventilátora v reálnom čase. Strojové učenie založené na porovnávaní vibračných signálov síce nedokáže pracovať kontinuálne, no aj napriek tomu podáva dostatočne presné výsledky a dal by sa uplatniť v praxi. Oba naše systémy dokázali odhadnúť danú procesnú veličinu a dali by sa aplikovať aj na ďalšie veličiny, ako napríklad prietok kvapaliny v potrubí alebo otáčky motora.

4 Záver

Cieľom tejto práce bolo navrhnúť systém, ktorý by dokázal odhadnúť rôzne procesné veličiny na základe zvukových a vibračných otlačkov. Pomocou zariadenia flexy sme boli schopný nameraťzložené zvukové a vibračné signály, ktoré sme spracovali pomocou rýchlej Fourierovej transformácie. Táto metóda predstavuje algoritmus, ktorý je schopný premeniť takýto signál na frekvenčné spektrum a z časovej závislosti signálu vytvorí frekvenčnú závislosť. Frekvenčné spektrum je graf závislosti amplitúdy od frekvencie, v ktorom hľadáme významné vlastnosti, takzvané píky. Na Spracovanie píkov sme využili regresný strom, model, pomocou ktorého sme dokázali na základe zvuku a vibrácií predikovať výkon ventilátora. Vysvetlili sme si aj na akom princípe funguje takýto model. Na konci práce sme si experimentálne overili funkčnosť našich modelov a vysvetlili, ktorá metóda sa dá kedy použiť a prečo. Do budúcna by bolo vhodné overiť, ako by sa správalo naše strojové učenie, ak by pracovalo viacero zariadení naraz a navzájom by si ovplyvňovali zvukové a vibračné stopy. Výsledky práce by sa dali zlepšiť, ak by sme miesto regresného stromu využili neurónové siete, poprípade ak by sme pracovali v prostredí s minimálnym šumom.

Zoznam použitej literatúry

[1] ČEPKO P. 2008. Spracovanie akustického signálu pomocou Fourierovej a waveletovej transformácie. Bakalárska práca. Žilina: EF UNIZA v Žiline.

[2] https://www.mathworks.com/discovery/feature-extraction.html

[3] HLADKÁ M. – HOLUB M. 2017. Introduction to Machine Learning. Článok z internetu <https://ufal.mff.cuni.cz/~holub/2017/docs/lec.Feature-Selection.2017-12-06_0.pdf>

[4] http://www.posterus.sk/?p=8350

[5] https://www.arduino.cc/en/Tutorial/ADXL3xx