

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV AUTOMATIZACE A INFORMATIKY

FACULTY OF MECHANICAL ENGINEERING INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

IDENTIFIKACE REGULOVANÉ SOUSTAVY – EXPERIMENTÁLNÍ JEDNOTKA PRO SPALOVÁNÍ BIOMASY

PROCESS IDENTIFICATION, EXPERIMENTAL UNIT FOR BIOMASS COMBUSTION

DIPLOMOVÁ PRÁCE
DIPLOMA THESIS

AUTOR PRÁCE **Radek Macalík**
AUTHOR

VEDOUCÍ PRÁCE **Ing. Vítězslav Máša**
SUPERVISOR

BRNO 2008

ZADÁNÍ DIPLOMOVÉ PRÁCE

ABSTRAKT

Diplomová práce se zabývá identifikací reálné experimentální jednotky pro spalování biomasy. Rozebírá jednotlivé metody identifikace, volbu modelu a metody odhadu jeho parametrů, zejména s využitím metody nejmenších čtverců a analýzy přechodové charakteristiky. Popisuje obecný identifikační cyklus a konkrétní algoritmy, které byly prakticky využity pro získání parametrů modelu. Uvádí nejrozšířenější softwarové nástroje využitelné v modelování a identifikaci systémů. Součástí je také implementace získaných matematických popisů s využitím programovacího prostředí Borland Delphi.

ABSTRACT

This thesis dissertates the identification of existing experimental unit of biomass combustion. It analyses particular methods of identification, the selection of a model and methods of the estimation of parameters, especially by using last square method and analysis of time-response characteristic. It describes common identification procedure and particular algorithms which were used in practice for obtaining model's parameters. The thesis also mentions popular software tools used for the modeling and the identification of systems. It comprises the implementation of obtained mathematic models using programming environment Borland Delphi as well.

KLÍČOVÁ SLOVA

Modelování, identifikace, identifikační cyklus, metoda nejmenších čtverců**,** System Identification Toolbox, spalování biomasy, Borland Delphi.

KEYWORDS

Modeling, identification, identification procedure, last square method, System Identification Toolbox, biomass combustion, Borland Delphi.

Poděkování

Na tomto místě bych rád poděkoval Ing. Vítězslavu Mášovi a doc. Ing. Zdeňku Němcovi, CSc. za jejich podněty a odborné připomínky, které mi pomohly při psaní této diplomové práce. Také bych rád poděkoval své přítelkyni a rodině za jejich všestrannou podporu.

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a že jsem uvedl všechny použité prameny a literaturu, ze kterých jsem čerpal.

Brno, říjen 2008 Radek Macalík

Obsah:

Zadání diplomové práce 2
Abstrakt 3
Poděkování 5
Prohlášení 6
1 Úvod 9
2 Identifikace systémů 10
2.1 Základní pojmy 10
2.1.1 Model, modelování a simulace 10
2.1.2 Systém 12
2.1.3 Identifikace 13
2.2. Matematický model dynamického systému 13
2.2.1 Definice a rozdělení 13
2.2.2 Odhad parametrů modelu 14
2.2.3 Praktické využití 16
2.3 Identifikační metody 16
2.3.1 Rozdělení metod 16
2.3.2 Analytická a experimentální identifikace 17
2.4 Identifikační cyklus 18
2.4.1 Volba vstupních a výstupních veličin 20
2.4.2 Testovací signál a vzorkovací frekvence 20
2.4.3 Struktury modelu 21
2.4.4 Odhad parametrů modelu metodou nejmenších čtverců 27
2.4.5 Ověřování a volba optimálního modelu 31
2.4.6 Mnohorozměrové regulační obvody 33
2.5. Softwarová podpora 34
2.5.1 Matlab 34
2.5.2 System Identification Toolbox (SIT) 35
2.5.3 Simulink 35
3 Experimentální jednotka pro spalování biomasy 37
3.1 Technologie 38
3.2 Řídicí systém 39
3.3 Rozbor funkčních vazeb a závislostí 40

4. Identifikace předmětné technologie 42
4.1 Výběr procesních veličin pro identifikaci 42
4.2 Návrh experimentálního měření a jeho realizace 43
4.3 Předzpracování provozních dat 45
4.4. Algoritmus 1 (model ARX) 45
4.5 Technická omezení identifikace 46
4.5.1 Nelineární chování 46
4.5.2 Dopravní zpoždění 48
4.5.3. Zásahy regulace 48
4.5.4 Systém dodávky paliva 50
4.5.5 Závěr 50
4.6 Sestavení modelu 51
4.7 Algoritmus 2 (analýza přechodové charakteristiky) 52
4.8 Zhodnocení procesu identifikace 59
5. Dynamický model jednotky 60
5.1 Softwarová podpora (Borland Delphi 7) 60
5.2 Přehled GUI a možností programu 61
5.3 Přehled implementovaných tříd, procedur a funkcí 64
5.4 Výpočetní logika – algoritmus zpracování 65
5.5 Výsledky modelování 69
6. Závěr 71
Seznam použitelné literatury 72
Seznam příloh 73

1 ÚVOD

Tato diplomová práce se zabývá identifikací reálné technologie experimentální jednotky pro spalování biomasy a fytomasy, která stojí v Kojetíně. Cílem práce, jak ho definuje zadání, je sestavení matematického modelu této reálné technologie. Vedle představení samotné jednotky a její technologie se práce věnuje zejména identifikaci systémů, jako efektivní cestě k sestavení matematického popisu tohoto komplexního dynamického systému. Možnost sběru a vyhodnocení dostatečného množství procesních dat z reálného provozu byla pobídkou k využití tzv. experimentální identifikace, která popisuje systém pomocí relací mezi jednotlivými vstupy a výstupy. Prakticky to obnáší sběr a analýzu naměřených dat, jejich zpracování do použitelné podoby, volbu vhodného modelu, kritéria kvality, metody odhadu parametrů a na závěr verifikaci shody modelu s reálným systémem.

Důležitou oblastí problematiky identifikace je popis a rozbor metod stanovení matematických modelů, schopných popsat a vyjádřit chování soustavy a rovněž jsou uvedeny metody odhadu parametrů modelů zvolené struktury. Důraz je kladen zejména na metodu nejmenších čtverců a odhad parametrů pomocí analýzy přechodové charakteristiky.

Diplomová práce popisuje obecný identifikační cyklus a následně detailněji pracuje s jeho jednotlivými kroky jak v obecné rovině, tak ve vztahu k řešené úloze.

Další významnou oblastí, které se práce dotýká je popis softwarových nástrojů používaných pro identifikaci a modelování systémů se zaměřením na jejich praktické využití.

Závěrečná kapitola se zabývá využitím programovacího prostředí Borland Delphi, v němž byly získané a ověřené přenosy implementovány s využitím diferenčních rovnic. Tento program představuje vizuální simulaci provozu identifikované jednotky v okolí definovaného pracovního bodu a umožňuje uživateli ovlivňovat chování této jednotky změnou hodnot vstupních parametrů.

2 IDENTIFIKACE SYSTÉMŮ

S pojmem identifikace se můžeme setkat v mnoha oborech lidské činnosti. Nejen v oborech spjatých s výpočetní technikou nebo matematikou, ale rovněž v bankovnictví, v ekonomii (obecně mluvíme o finanční identifikaci) nebo v přírodovědě. Nahradíme-li slovo identifikace českým ekvivalentem „rozpoznávání" nebo „analýza," zjistíme, že tento termín zasahuje téměř všude. Identifikace nám pomáhá vyznat se v dílčích oblastech světa, který nás obklopuje, lépe rozpoznat jeho vlastnosti a chování. Takto vytyčené podmnožiny našeho okolního světa pak můžeme označit pojmem „systém." Zkoumaný systém můžeme právě pomocí identifikace mapovat a získané znalosti a informace dále využívat. Mluvíme-li o identifikaci systémové, jejím cílem je nalezení modelu nějaké reálné soustavy, který by co nejpřesněji popisoval (vyjadřoval) její chování ve smyslu vnitřních funkčních závislostí mezi vstupními a výstupními veličinami.

Konkrétně to znamená, že musíme vědět, jaký systém chceme identifikovat, mít přehled o vnitřním chování tohoto systému, nebo o jeho chování v závislosti na vnějších podnětech na něj působících a musíme znát určitý postup, kterým chceme samotnou identifikaci provést.

V poslední době se identifikace uplatňuje zejména tam, kde to dříve kvůli nedostatečným výpočetním kapacitám schopným pracovat s náročnými matematickými modely buď nebylo možné, nebo by to nebylo efektivní (náklady na identifikaci by převýšily přínosy).

2.1 Základní pojmy

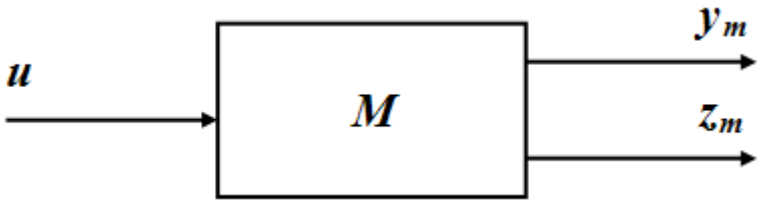
Než se budeme věnovat jednotlivým krokům systémové identifikace, uveďme si některé základní pojmy, související s touto problematikou a jejich vysvětlení.

2.1.1 Model, modelování a simulace

Termínem *model* rozumíme matematický popis reálného systému (viz kap. 2.1.2) vytvářeného pro určité účely, např. hluboké porozumění modelovanému systému (matematicko-fyzikální analýza), odhadnutí signálů, jež nemůžeme změřit, z nepřímého měření, testování hypotéz (diagnózy v medicíně), předpovídání krátkodobého a dlouhodobého dynamického chování soustavy nebo řízení procesů.

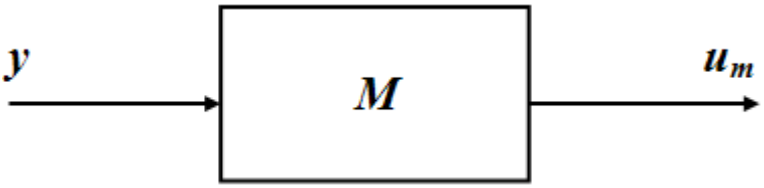
Modelem dynamického systému rozumíme pravidlo, podle něhož můžeme z předem známých nebo změřených veličin vypočítat časový průběh sledovaných veličin systému. Model se nejčastěji vypočítává ze známého vstupu u, výstup y_m (výstup modelu) a je navržen tak, aby se výstup modelu blížil skutečnému výstupu ze systému y jen jak je to možné (obr. 1). Symbol z v obrázku reprezentuje další charakteristiky, které v sobě mohou zahrnovat neměřitelné signály a tím se lišit od y.

Zkráceně lze říci, že úkolem modelu je co nejvěrohodněji predikovat či reprodukovat chování systému. [1]



Obr. 1 Model

Existuje rovněž tzv. inverzní model, který ze známého výstupu y ze systému spočítá vektor u_m, který se co nejvíce blíží skutečnému vstupu u (obr. 2).



Obr. 2 Inverzní model

Získání relevantního modelu reálné technologie (soustavy) – kotle na biomasu, popisujícího chování kolem nominálního bodu, je předmětem této diplomové práce. Než si vysvětlíme další pojmy, podívejme se v této souvislosti na možné rozdělení obecných modelů.

Podle kritéria při jejich vytváření je můžeme rozdělit na fyzikální, nebo abstraktní. Jak uvádí [2], fyzikální model představuje přirozený nebo umělý hmotný systém, vytvořený na základě fyzikální podobnosti. Abstraktní model je pak tvořen nehmotným systémem, představou nebo grafickým vyjádřením popisujícím zkoumaný systém. Abstraktní modely můžeme dále rozdělit na matematické a simulační. Matematický model je vyjádřen matematickými vztahy a rovnicemi, simulační model je vyjádřen programovacími prostředky. Podobným způsobem můžeme rozdělit modely na dva typy: white-box, nebo black-box. V případě white-box modelu známe předem všechny důležité informace o modelovaném systému, zpravidla je získáváme matematicko-fyzikální analýzou. U black-box modelu naproti tomu žádné podobné informace o systému nemáme, jeho chování musíme aproximovat. Black-box model může být ještě rozdělen na uzavřenou smyčku (close-loop, obsahuje zpětnou vazbu regulace) nebo otevřenou smyčku (open-loop, neobsahuje zpětnou vazbu regulace).

Modelování představuje experimentální proces, při němž se zkoumanému originálu – modelovanému systému (reálnému objektu, dílu, stroji) jednoznačně podle určitých kritérií přiřazuje fyzický nebo abstraktní model – jiný systém [2]. Jinými slovy, jedná se o nahrazení reálného modelu modelem simulačním, což s sebou ovšem nese nutný fakt zjednodušení celého problému na užší, specializované měřítko, které obsahuje pouze pro nás důležité parametry.

V případě tvorby modelu experimentální soustavy je nutno vycházet z celkové architektury systému a poté vybrat z možných variant takovou, která nejlépe odpovídá cílům účinného provozu dané soustavy, tj. cílům regulace. To znamená stanovit vhodné vstupní a výstupní veličiny, které jsou reálně a efektivně měřitelné a jejichž výsledky jsou pro nás důležité. V našem případě jsme se z těchto důvodů

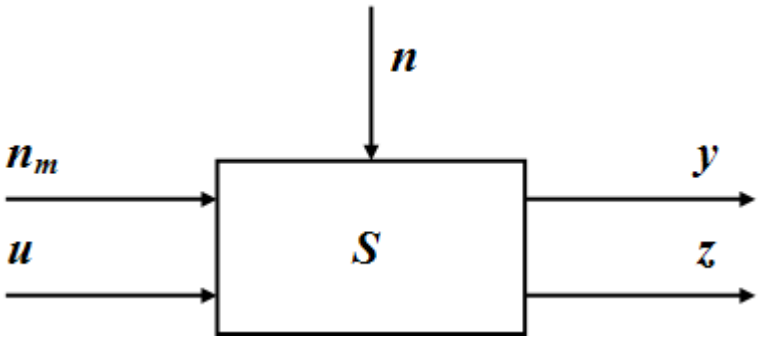
rozhodli například pro sledování koncentrace o_2 ve spalinách, což je hodnota, která vypovídá o dokonalosti spalovacího procesu nebo teploty výstupní vody (koncový produkt celého systému), která se využívá jako topná voda pro ohřev výrobní haly.

S modelováním je také úzce spojen termín simulace. Jedná se o napodobování chování reálného procesu nebo soustavy (obecně systému), pomocí vhodného, například matematicko-fyzikálního popisu. Zejména s využitím výpočetní techniky můžeme testovat nebo předvídat chování systému pomocí změny vstupních, nebo jiných podmínek. Simulace najde využití zejména tam, kde by testování reálného provozu bylo časově, ekonomicky či jinak nemožné, nebo neefektivní.

2.1.2 Systém

V souladu s literaturou [1] můžeme definovat systém (proces, soustavu) jako vybranou součástí našeho okolí, kterou považujeme za celek, a jsme s ní ve vzájemné interakci. Pro systém platí:

- Pozorujeme jeho vnější charakteristiky (chování) a výsledek tohoto pozorování označujeme jako výstup ze systému, vektor y.
- Ze systému mohou vystupovat další charakteristiky (označme je z), které v sobě mohou zahrnovat neměřitelné signály a tím se lišit od y.
- Působíme na systém veličinami, jejichž průběh v čase známe a můžeme ho více či méně ovlivňovat. Tyto veličiny jsou vstupy a označíme je jako u.
- Na systém působí dva druhy poruch: ty, které můžeme měřit přímo na vstupu nazveme jako měřené poruchy a označíme jako n_m; ty, jejichž působení můžeme pozorovat pouze na výstupu ze systému nazveme šumem a označíme jako n.
- Pro účely modelování je rozdíl mezi vstupním signálem u a měřenou poruchou n_m nepodstatný, takže se jí již dále nebudeme zabývat a bude pro nás jen dalším vstupem do systému.



Obr. 3 Systém

Systémem tedy může být prakticky jakékoliv reálné zařízení, splňující výše uvedené podmínky.

2.1.3 Identifikace

Pojem identifikace jsme si již krátce představili v úvodu této kapitoly. Nyní se podívejme podrobněji na rozdělení tzv. systémové identifikace, jejímž cílem je nalezení modelu nějaké reálné soustavy, který by co nejpřesněji popisoval (vyjadřoval) její chování ve smyslu vnitřních funkčních závislostí mezi vstupními a výstupními veličinami.

V této souvislosti můžeme hovořit o identifikaci *analytické* a *experimentální*. Jak uvádí [2], u identifikace analytické vycházejí základní postupy vedoucí k sestavení matematických modelů systému z matematicko-fyzikální analýzy. Jedná se nejčastěji o systémy elektrické, hydraulické, pneumatické, mechanické a tepelné. U identifikace experimentální vyšetřujeme dynamické vlastnosti systému a jeho matematický model sestavujeme experimentálním postupem. To znamená, že pomocí vhodných testovacích signálů působíme na systém a zaznamenáváme jeho odezvu. Z vyhodnocení měřených signálů poté určíme model systému. Porovnání těchto dvou typů identifikace se podrobně věnuje kapitola 2.3.

Ke kvalitní identifikaci je nutno získat dostatečně přesný popis statických i dynamických vlastností systému, který chceme regulovat. V případě experimentální identifikace to znamená provést sběr dat příslušných provozních veličin, jimiž jsou hodnoty vstupu a výstupu systému a následně mezi nimi najít odpovídající vztahy a závislosti. Pro účely této diplomové práce byla v první fázi využita nadstavba programu Matlab – modul *System Identification Toolbox*. Cílem bylo získat parametry modelu na základě jeho vhodně zvolené struktury.

2.2. Matematický model dynamického systému

2.2.1 Definice a rozdělení

Jak již bylo uvedeno v kap. 2.1.1, abstraktní modely se dělí na matematické a simulační. *Matematický model* je vyjádřen matematickými vztahy a rovnicemi, kdežto *simulační model* je vyjádřen programovacími prostředky. Právě popisu chování dynamického systému (tj. systému, jehož parametry se mění v čase) pomocí matematického modelu se věnuje tato kapitola. Literatura [4] definuje matematický model jako *matematické vyjádření podstatných vlastností existujícího nebo konstruovaného systému, který popisuje znalosti o systému v použitelné formě.*

Matematický model můžeme rozdělit na *parametrický* a *neparametrický*.

Parametrické modely

Mají danou strukturu. Strukturou rozumíme řád a zvolený typ diferenciální či diferenční rovnice (lineární, nelineární diferenční rovnice, typ nelinearity, atd.) nebo soustavu těchto rovnic, obrazový nebo diskrétní přenos se zvolenými stupni polynomů v čitateli a jmenovateli nebo přenosovými maticemi zvoleného rozměru, atd. Parametrické modely představují z matematického hlediska rovnice nebo soustavy rovnic a algebraické vztahy, které explicitně obsahují koeficienty těchto rovnic a vztahů. Obecně pak označujeme tyto koeficienty jako parametry matematických modelů. [3]

Neparametrické modely

Představují zpravidla funkční závislost mezi zvoleným vstupním a odpovídajícím výstupním signálem (např. y/u, y/d, atd.). Tato závislost se vyjadřuje buď graficky pomocí záznamu z měřených odezev systému (zapisovač signálů), nebo pomocí tabulky hodnot, popisující číselně danou závislost. Neparametrické modely vyjadřují zpravidla přechodovou, váhovou nebo frekvenční charakteristiku v grafické, nebo v tabulkové formě. Parametry modelu jsou pak obsaženy implicitně v těchto funkčních závislostech. Lze je získat až jejich následným vyhodnocením pro zvolenou strukturu modelu. [3]

Matematickým modelem dynamického systému tedy budeme rozumět abstraktní model určité reálné soustavy (originálního objektu), který nám umožňuje zkoumat jevy probíhající na příslušné reálné soustavě, pomocí matematického popisu jejich průběhu. Sestavení matematického modelu vychází z vymezení zkoumaného jevu a definice sledovaných příznaků – definice systému na reálném objektu. Pomocí matematického modelu můžeme zkoumat průběhy sledovaných fyzikálních veličin. [2]

Samotný matematický model nám však ještě nedává informace v takové podobě, kterou potřebujeme pro vyhodnocení zkoumaného děje. Tento matematický model je třeba nejprve vyhodnotit pomocí vhodných výpočetních metod [14].

2.2.2 Odhad parametrů modelu

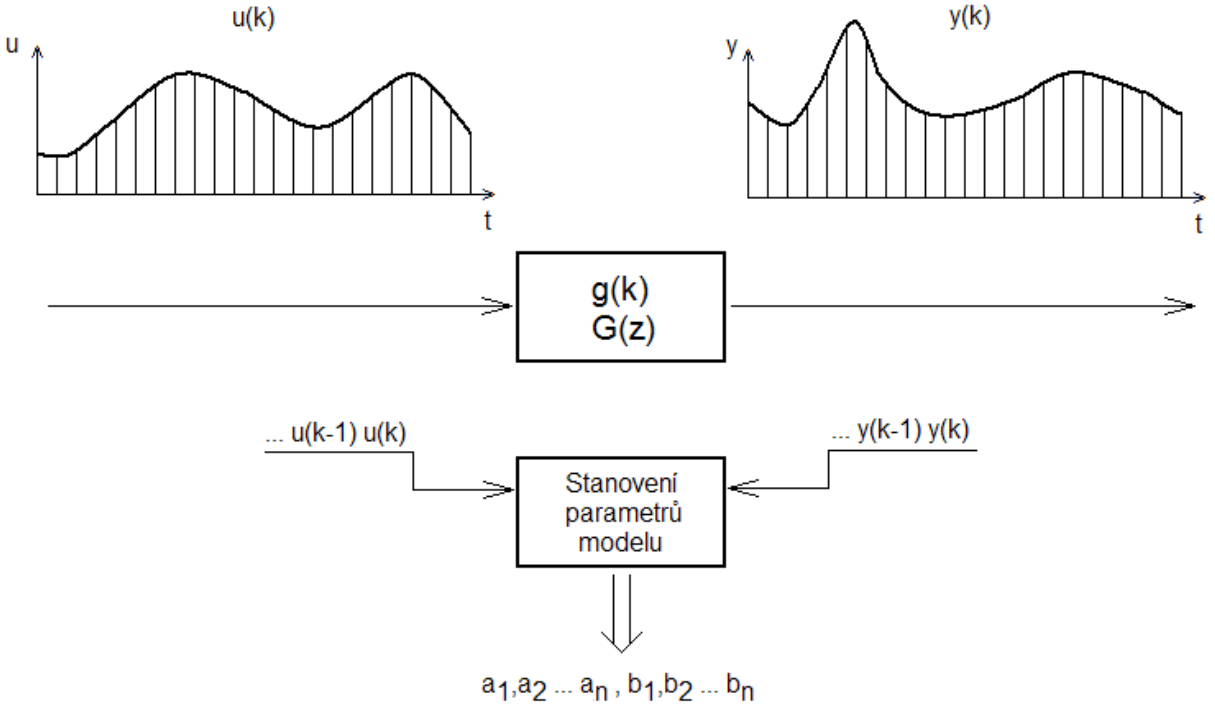
[2] Metody odhadů parametrů umožňují stanovit parametry modelu zvolené struktury (řád modelu, stupeň polynomu v čitateli a jmenovateli přenosu, zavedení dopravního zpoždění, atd.) z průběhu posloupnosti změřených hodnot vstupu *u* a výstupu *y*. Metody předpokládají diskrétní měření signálů, čímž se získají odpovídající si číselné posloupnosti hodnot vstupu *u(k)* a výstupu *y(k)*.

Matematický model se předpokládá v diskrétním tvaru, vyjádřený např. diferenční rovnicí,

$$y(k) + a_1 * y(k-1) + \ldots + a_n * (k-n) = b_o * u(k) + b_1 * u(k-1) + \ldots + b_n * u(k-n) \qquad (1)$$

resp. Z-přenosem:

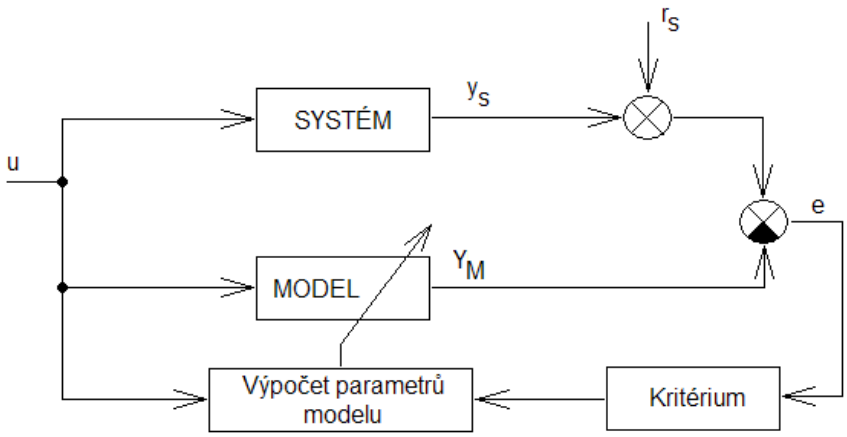
$$G(z) = \frac{b_o + b_1 * (z-1) + \ldots + b_n * (z-n)}{1 + a_1 * (z-1) + \ldots + a_n * (z-n)} \qquad (2)$$



Obr. 4 Odhad parametrů modelu [2]

Ověření správnosti modelu se provádí porovnáním výstupu systému y s výstupem modelu y_M při vybuzení systému i modelu stejným vstupním signálem. Obr.5 ukazuje blokové schéma postupu při odhadu parametrů modelu. Měřitelný výstupní signál systému y se skládá z neměřitelného výstupu y_s a poruchy na výstupu r_s. Přenos modelu je vyhodnocován odchylkou výstupů $e = y - y_M$.

Pokud je tato dostatečně malá, znamená to, že byly odhadnuty dostatečně přesné parametry modelu. Pokud tomu tak není, je nutné vhodným způsobem změnit koeficienty modelu, aby se chyba zmenšila. Stavitelné parametry modelu systému shrnujeme do vektoru parametrů p. Jak uvádí [5], tento vektor je v podstatě optimální sada parametrů pro aproximační model, ve smyslu minimalizace střední kvadratické odchylky mezi modelem a signálem. O jednom z těchto aproximačních modelů, metodě nejmenších čtverců, pojednává kapitola 2.4.4.



Obr.5 Znázornění postupu stanovení parametrů modelu systému [2]

2.2.3 Praktické využití

Takto získaný model již můžeme využít například pro předpovídání chování příslušného systému v závislosti na provedených změnách vstupních veličin, modelování chování soustavy v krajních situacích, hledání optimálního chování soustavy a podobně. Pomocí kvalitního modelu můžeme testovat chování soustavy lépe, rychleji, levněji a efektivněji, než by bylo možné na reálné soustavě.

2.3 Identifikační metody

2.3.1 Rozdělení metod

Z výše uvedené definice vyplývá, že volba konkrétní identifikační metody je závislá na mnoha vnitřních parametrech a vlastnostech analyzovaného systému. Pojďme si proto stručně představit některé z těchto kritérií.

1. Typ matematického modelu

   Může být parametrický (analytická identifikace) nebo neparametrický (experimentální identifikace). Bližší představení těchto dvou typů modelů viz kapitola 2.3.2.

2. Typ použitého vstupního signálu

   Deterministický signál je takový, který můžeme popsat analyticky, například skokovou změnou nebo pulsem. Stochastický signál je signál náhodný, popsaný statistickými charakteristikami střední hodnotou a rozptylem (opak deterministického signálu). Pseudonáhodný signál je kus náhodného signálu, opakující se každou periodu *T*. Uměle vytvořený signál (testovací) je signál záměrně vybuzen naší činností.

3. Podle měření a vyhodnocení dat

   V této souvislosti můžeme hovořit o off-line identifikaci, při níž jsou v průběhu experimentu data nejdříve zaznamenána, uchována na vhodném nosiči a po následném vyhodnocení jsou stanoveny parametry modelu, nebo on-line identifikaci, která předpokládá připojení počítače v reálném čase k identifikovanému procesu a bezprostřední vyhodnocení parametrů modelu.

4. Podle algoritmu zpracováni dat

   V návaznosti na předchozí bod uveďme algoritmus zpracování v dávkách – jednorázově (nerekurzivně) pro off-line identifikaci, nebo algoritmus zpracování dat v reálném čase (rekurzivně) pro on-line identifikaci. V prvním případě nejprve naměříme experimentální data a poté je vyhodnocujeme, v druhém případě je vyhodnocujeme rovnou na běhu systému v reálném čase.

2.3.2 Analytická a experimentální identifikace

V kapitole 2.1.3 byla zmíněna experimentální identifikace. Ta spolu s identifikací analytickou představuje dva základní přístupy. Pojďme si nyní říct více o tom, co je charakterizuje, jak se vzájemně liší a kdy je vhodné použít který. Analytický i experimentální přístup má své výhody a nevýhody, a proto je v praxi musíme často vzájemně kombinovat [2].

Jak uvádí [2], zásadní rozdíl je v tom, že analytické postupy sestavení matematického modelu umožňují analyzovat dynamické vlastnosti vyvíjených zařízení ještě před jejich zhotovením. Simulační závěry tak mohou významně přispět k dimenzování zařízení, mohou upřesnit projekční podklady zařízení a podobně. Analyticky získané modely lze následně upřesnit pomocí měření a aplikace metod experimentální identifikace na prototypovém nebo zkušebním zařízení.

Metody experimentální identifikace vyžadují existenci zkoumaného objektu a možnost experimentovat s ním. Nevyžadují přesnou znalost struktury systému a popis probíhajících procesů, avšak jsou náročné na přístrojové vybavení. Celkově platí, že čím lepší apriorní znalosti o zkoumaném objektu jsou k dispozici, tím lépe je možné připravit identifikační postup, vyhodnotit měřená data a sestavit přesnější matematický model. Analytické metody identifikace vyžadují hlubokou znalost fyzikálních procesů a jejich matematického popisu probíhajícího ve zkoumaném systému. Získání potřebných zkušeností je dlouhodobou záležitosti a je podmíněno i dostatečným zázemím. Proto se analytické metody rozvíjejí zejména v odvětvích, která mají silná průmyslová zázemí.

Experimentální metody jsou nákladné pro potřebné přístrojové vybavení a experimentování s objektem. Získávané zkušenosti z vyhodnocení dat jsou obecně platné a použitelné při identifikaci různých procesů. Potřebná znalost procesu je nižší, než u metod analytické identifikace.

Tabulka 1 porovnává důležité vlastnosti analytické a experimentální identifikace:

Vlastnost modelu	***Analytická identifikace***	***Experimentální identifikace***
Struktura modelu	Struktura vyplývá z přírodních zákonů.	Struktura musí být zvolena.
Popis systému	Systém je popsán pomocí vnitřních stavových proměnných a vstupo-výstupního chování.	Systém je popsán pouze pomocí relace vstup – výstup.
Parametry modelu	Parametry modelu jsou funkcemi systémových veličin, mají fyzikální význam.	Parametry modelu jsou analytické proměnné, které neumožňují většinou nalézt souvislost s fyzikálními systémovými proměnnými.
Platnost modelu	Model platí pro celou třídu typů procesu a pro různé provozní stavy.	Model platí pouze pro zkoumaný proces a konkrétní provozní stav.

Existence originálního systému	Model může být vytvořen i pro neexistující systém.	Model může být identifikován pouze pro existující systém.
Znalost vnitřní struktury	Důležité vnitřní procesy systému musí být známé a matematicky popsatelné.	Vnitřní procesy nemusí být známé.
Opakované použití metody	Každá tvorba modelu představuje opakovanou aplikaci fyzikálních zákonů. Zvyšování know-how řešitele.	Metody nezávislé na jednotlivých systémech, vytvořené programové vybavení může být opakovaně použito pro identifikaci různých systémů.
Časová náročnost	Tvorba modelu vyžaduje větší časové nároky.	Menší časové nároky.

Tab. 1 Vlastnosti analytické a experimentální identifikace [2]

Tématem této diplomové práce je identifikace reálné technologie, proto se budeme pohybovat v oblasti experimentální identifikace, jejímž výsledkem by měl být matematický model (definice v kap. 2.2) prototypu jednotky pro spalování biomasy. Parametry byly zjištěny experimentálně, nemají fyzikální význam. Analytická identifikace by byla vzhledem ke složitosti matematického popisu spalovacího procesu prakticky nerealizovatelná. Použijeme off-line identifikaci, v dávkách, nerekurzivně, neboť nejprve jsme data naměřili a poté vyhodnotili. Typ signálu je deterministický, jedná se o skokové změny řídicích veličin.

2.4 Identifikační cyklus

Proces hledání uspokojivého modelu se skládá s následujících kroků, jak jsou uvedeny v literatuře [16]:

1. Plánování experimentu - experimentovat s reálným systémem je náročné a drahé, proto se používá analýza odezvy systému na vstupní signál.

2. Volba struktury modelu – zvolíme ji na základě znalosti o systému, poruchách, které na něj působí nebo podle pracovních bodů.

3. Volby vhodného kritéria kvality – zvolením přesnosti, s jakou budeme chtít sestavit model.

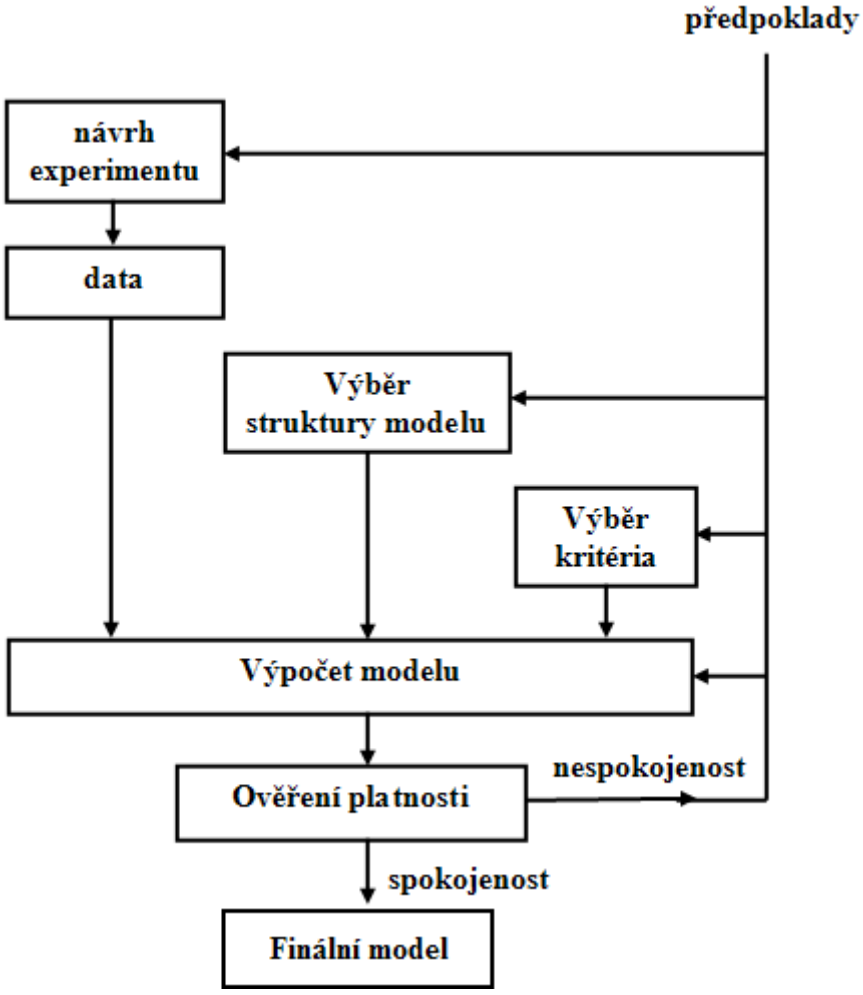
4. Odhad parametrů – k jejich odhadu potřebujeme znát: vstupní/výstupní data, třídu přesnosti, kritérium. Poté můžeme použít klasické metody odhadu parametrů:

   - Analýza přechodové a frekvenční charakteristiky (určení časových konstant, řádu systému).
   - Metoda korelační a spektrální analýzy (analýza odezvy na Dirakův impuls).

- Metoda nejmenších čtverců a její modifikace.
- Metoda maximální věrohodnosti.

5. Test shody chování modelu a systému = verifikace. Spočívá v porovnání odezev modelu a skutečného systému.

Uvedený postup graficky znázorňuje obr. 6:



Obr. 6 Identifikační cyklus [1]

Model nemusí vyhovovat z různých důvodů, uveďme např.:

- Selhal numerický algoritmus hledající nejlepší model dle daného kritéria.
- Bylo špatně zvoleno kritérium.
- Zvolena nevhodná struktura modelu.
- Změřená data nebyla dostatečně informativní (málo dat, vliv poruchových veličin, zásahy regulátoru, nepředpokládané chování soustavy, atd.)

Pokud jsme s výsledným modelem spokojeni, získali jsme finální model vhodný pro další použití a popis soustavy a identifikační cyklus je ukončen. V opačném případě, pokud výsledný model neprošel testem kvality, se musíme vrátit zpět v cyklu a změnit příslušné kroky.

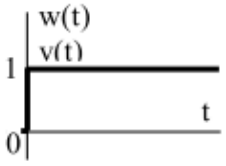
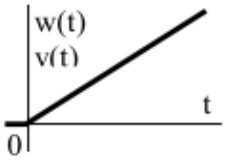
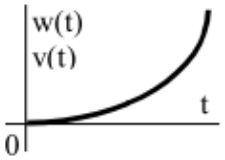
2.4.1 Volba vstupních a výstupních veličin

Prvním krokem v procesu identifikace je výběr vstupních a výstupních (řídicích a řízených) veličin, vhodně zvolený interval vzorkování (sběru dat) a vhodný testovací signál. Vybíráme pokud možno na sobě nezávislé vstupní veličiny, které ovlivňují měřené výstupy. V procesu řízení se vstupy nazývají řídicí veličiny, výstupy naopak řízené veličiny. Důležité jsou také poruchové veličiny. V procesu spalování získáváme tyto veličiny na základě požadavků řízení a zkušenosti. Přehled konkrétních zvolených řídicích a řízených veličin vybraných pro účely této diplomové práce uvádí kap. 4.1.

Poruchové veličiny, které by mohly být v případě jednotky pro spalování biomasy zvažovány (zejména tepelné ztráty), jsou časově stálé a plně závislé na vstupních veličinách. Ve výsledku jsou proto zahrnuty v přenosu regulované soustavy.

2.4.2 Testovací signál a vzorkovací frekvence

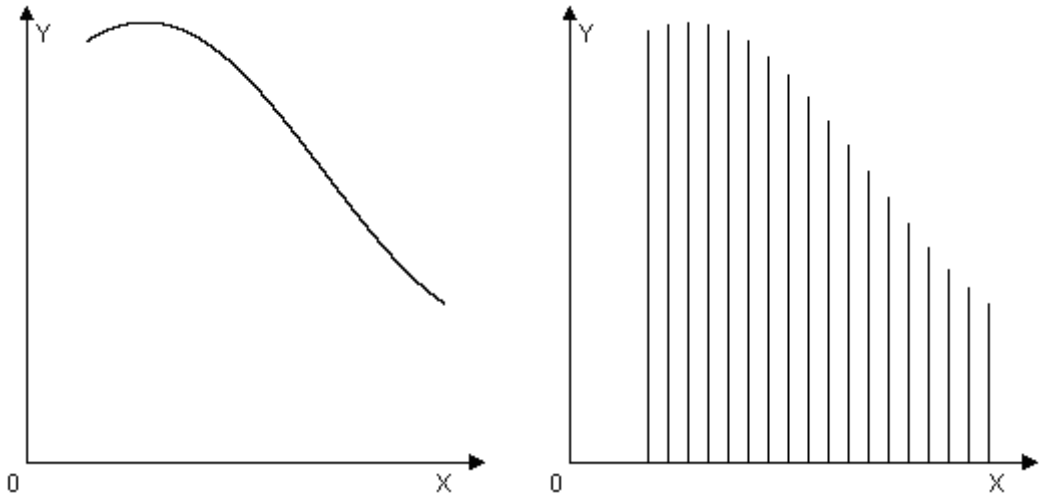
Testovacím signálem obecně rozumíme změnu řídicí veličiny. Podle charakteru změny můžeme hovořit o skoku polohy (dané veličiny), skoku rychlosti (rampová nebo též lineární funkce) nebo skoku zrychlení (kvadratická funkce), viz následující tabulka [7]:

testovací signál	skok polohy (dané veličiny)	skok rychlosti (rampová nebo též lineární funkce)	skok zrychlení (kvadratická funkce)
graf			
rovnice	$w=v=1 \quad (t \geq 0)$	$w=v=t \quad (t \geq 0)$	$w=v=t^2/2 \quad (t \geq 0)$
Laplaceův obraz	$W(s)=V(s)=\frac{1}{s}$	$W(s)=V(s)=\frac{1}{s^2}$	$W(s)=V(s)=\frac{1}{s^3}$

Tab. 2 Testovací vstupní signály [7]

Analýza odezvy systému na zvolený testovací signál umožňuje stanovit pracovní oblast, ve které se proces chová lineárně. Jako testovací signál pro analýzu, stejně jako pro identifikaci, byla použita skoková změna v obou směrech.

Vzorkovací frekvencí rozumíme časovou periodu, v níž jsou spojitá data zaznamenávána. Jedná se v podstatě o diskretizaci spojitého průběhu měřených veličin. Délka této periody závisí na znalosti konkrétního systému a jeho vnitřních dějů. V případě technologie pro spalovani biomasy se ukázala jako dostatečná vzorkovací perioda 15 s. Odezvy byly zaznamenány a uloženy do databáze.



Obr.7 Spojitá vs. diskrétní oblast

Obrázek vlevo ukazuje skutečný spojitý průběh měřené veličiny, zatímco obrázek vpravo zobrazuje interval vzorkování pro danou časovou periodu.

2.4.3 Struktury modelu

Volba struktury (tj. množiny kandidátů struktur) modelu patří k nejdůležitějším a nejsložitějším krokům v procesu identifikace. Při výběru modelu hrají kromě dobré znalosti regulovaného systému důležitou roli také zkušenosti a intuice, neboť předem nikdy nemůžeme s určitostí vědět, který model se ve výsledku osvědčí. Proto se v praxi často testuje několik různých variant, abychom mohli vybrat tu nejlepší.

Výběr vhodného modelu také musíme podřídit následujícím kritériím [1]:

- Účel modelování.
- Pracovní podmínky, při kterých bude model použit (rozsahy a vlastnosti vstupů, šumů, atd.).
- Přípustné náklady na model.
- Různé další užitečné informace (např. informace o řídicím systému, s kterým bude model spolupracovat).

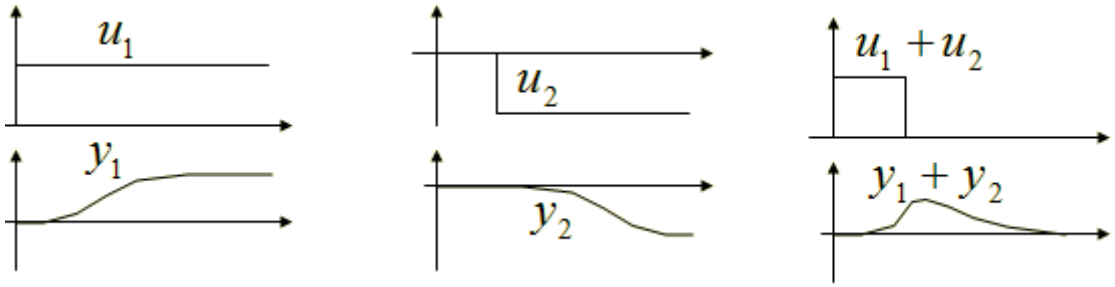
Modely můžeme dále rozdělit na:

- Lineární vs. nelineární.
- Modely ve spojité vs. diskrétní oblasti.

U lineárního modelu rozlišujeme linearitu ve vstupech a linearitu v parametrech.

Literatura [1] dále uvádí, že model je lineární ve vstupech, jestliže jeho výstupy splňují princip superpozice s ohledem na jeho vstupy, což znamená jestliže platí:

$$\mathbf{y}_m(t,\mathbf{p},\lambda\mathbf{u}_1+\mu\mathbf{u}_2)=\lambda\mathbf{y}_m(t,\mathbf{p},\mathbf{u}_1)+\mu\mathbf{y}_m(t,\mathbf{p},\mathbf{u}_2) \tag{3}$$



Obr.8 Princip superpozice [8]

kde $\mathbf{y}_m(t,\mathbf{p},\mathbf{u})$ je výstup modelu v čase t, s parametry p, se vstupy u a s nulovými počátečními podmínkami. Většinou se také předpokládá, že se jedná o model časově nezávislý, tedy že jeho parametry nejsou závislé na posunu časové osy.

Model je lineární v parametrech, jestliže jeho výstupy splňují princip superpozice s ohledem na jeho parametry, to znamená jestliže platí:

$$\mathbf{y}_m(t,\lambda\mathbf{p}_1+\mu\mathbf{p}_2,\mathbf{u})=\lambda\mathbf{y}_m(t,\mathbf{p}_1,\mathbf{u})+\mu\mathbf{y}_m(t,\mathbf{p}_2,\mathbf{u}) \tag{4}$$

Pro lepší názornost se podívejme na několik příkladů:

$y_m(t+1,p)=pu(t)$	Je lineární jak v parametrech tak i ve vstupech.
$y_m(t+1,p)=py_m(t,p)+u(t)$	Je lineární ve vstupech, ale ne v parametrech.
$y_m(t+1,p)=pu^2(t)$	Je lineární v parametrech, ale ne ve vstupech.
$y_m(t+1,p)=py_m^2(t,p)+u(t)$	Není lineární ani vstupech ani v parametrech.

Pokud je to možné, je lepší preferovat modely lineární. Pro modely lineární ve vstupech existuje rozsáhlá matematická teorie (stabilita, optimální řízení, robusní řízení, adaptivní řízení, atd.), a proto je tento popis zvlášť vhodný například pro oblast technické kybernetiky. Zároveň platí, že nalezení parametrů modelů lineárních v parametrech, je většinou jednoduché a mnohdy je možné použít explicitního vztahu a tím se vyhnout iterování při hledání „nejlepšího“ modelu z množiny kandidátů.

Nevýhodou ovšem je, že se v praxi těžko setkáme se systémem, který by byl lineární v celém rozsahu vstupně-výstupních veličin, a proto mají lineární struktury jen omezenou oblast použitelnosti. [1]

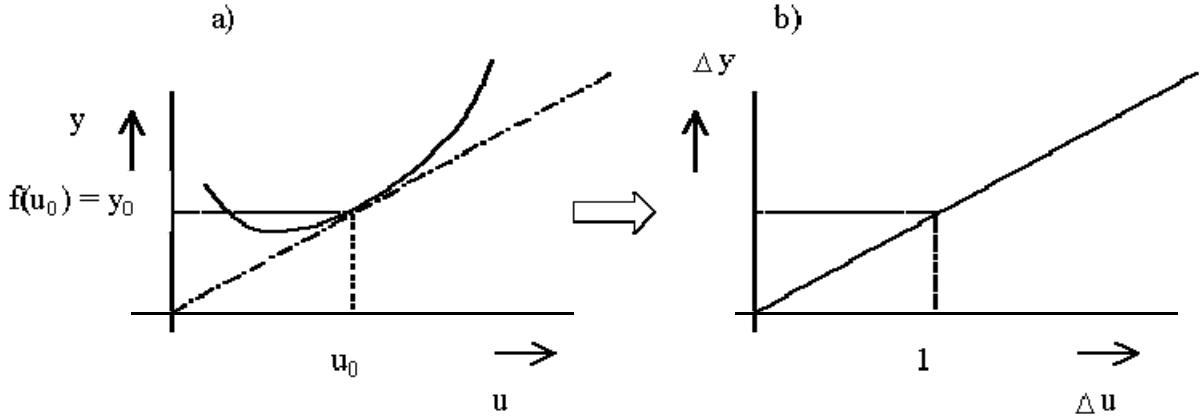
Jak uvádí [9], obecný nelineární model o n_x stavech, n_u vstupech a n_y výstupech tvořený soustavou n_x nelineárních diferenciálních a n_y algebraických rovnic lze zapsat ve vektorovém tvaru jako

$$\frac{dx}{dt} = f(x,u),\ y = g\,(x,u) \tag{5}$$

kde u je vektor n_u vstupních veličin,
y je vektor n_y výstupních veličin,
x je vektor n_x stavových veličin,
f je vektor n_x funkcí soustavy diferenciálních rovnic,
g je vektor n_y funkcí soustavy algebraických hodnot.

Nelineární matematický model dynamického systému se většinou snažíme linearizovat, tj. nahradit jej lineárním matematickým modelem, přičemž z důvodu co nejlepšího zachování vlastností systému linearizujeme charakteristiky pouze v pracovním bodě a jeho nejbližším okolí (pokud se posune, je nutno provést linearizaci znovu). Rozbor metod linearizace je mimo rozsah této diplomové práce, pro přehled si uveďme, že je možné použít metody linearizace tečnou nebo linearizaci z naměřených hodnot. Metody linearizace nelze však aplikovat na tzv. podstatně nelineární systémy, u kterých by se linearizací statické vlastnosti změnily zásadním způsobem.

Obr.9 ilustruje linearizaci nelineárního matematického modelu dynamického systému:



Obr 9. Linearizace nelineárního matematického modelu dynamického systému

Modely ve spojité vs. diskrétní oblasti

Jak uvádí [17], modelované soustavy převážně existují ve spojité časové oblasti. Proto také tradiční přístup k modelování je popis systémů diferenciálními rovnicemi. Na druhou stranu, masové používání číslicových počítačů znamenalo veliký rozmach teorie řízení v diskrétní časové oblasti. Simulace diskrétních modelů na číslicových počítačích je mnohem rychlejší a jednodušší než u spojitých modelů, což je předurčuje pro řízení v reálném čase. Nevýhodou je, že používání diskrétních modelů může způsobit ztrátu některých informací o chování modelovaného spojitého systému.

V této diplomové práci, jak už bylo uvedeno, se zabývám modelováním v diskrétní časové oblasti. Získané přenosy definující závislosti mezi vstupy a výstupy

zkoumaného systému, bylo třeba převést na diskrétní popis. K tomuto účelu byl použit následující postup:

Vyjdeme z přenosu proporcionálního systému se setrvačností 1. řádu:

$$F(s) = \frac{K}{1 + s * \tau} = \frac{Y(s)}{U(s)} \Rightarrow Y(s) + Y(s) * \tau * s = U(s) * K \quad (6)$$

Pomocí derivací převede do časové oblasti:

$$y(t) + y'(t) * \tau = u(t) * K \quad (7)$$

Pomocí kroku k a diskrétního časového intervalu T_v převedeme do diskrétní časové oblasti, kde T_v je vzorkovací interval:

$$t = k * T_v \quad (8)$$

Dostaneme:

$$y(k) + \tau * \frac{u(k) - u(k-1)}{T_v} = u(k) \quad (9)$$

$$y(k) * \left[1 + \frac{\tau}{T_v}\right] = y(k-1) * \frac{\tau}{T_v} + u(k) \quad (10)$$

$$y(k) = y(k-1) * \frac{\frac{\tau}{T_v}}{1 + \frac{\tau}{T_v}} + u(k) * \frac{1}{1 + \frac{\tau}{T_v}} \quad (11)$$

$$\text{kde } \frac{\frac{\tau}{T_v}}{1 + \frac{\tau}{T_v}} = a_1, \quad \frac{1}{1 + \frac{\tau}{T_v}} = b_0 \quad (12)$$

Výsledkem je dopředně diferenční rovnice, představující nejobecnější popis diskrétního systému:

$$y(k) = a_1 * y(k-1) + b_0 * u(k) \quad (13)$$

Musíme přesně dodržet podmínku $a_1 + b_0 = 1$, jinak by mohla vzniknout numerická nestabilita. Pracujeme-li s dopravním zpožděním, zahrneme jej do vzorce.

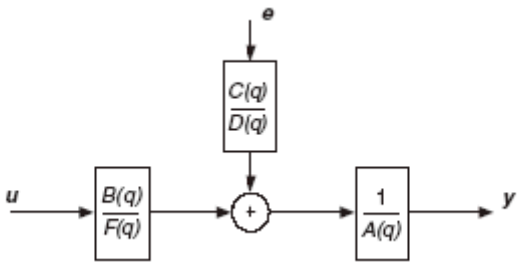
Následující tabulka porovnává modely ve spojité a diskrétní oblasti:

	MODELY	
	Spojitá oblast	Diskrétní oblast
Parametry	nezávisí na vzorkovací frekvenci	závisí na vzorkovací frekvenci
Simulace	vyžaduje diskretizaci	jednoduchá
Fyzikální informace o systému	mohou být zahrnuty viditelně	většinou nebrány v úvahu (zesílení výjimkou)
Časy vzorkování	nezávisle vybírány	dané vzorkovací frekvencí

Tab.3 Porovnání modelů ve spojité a diskrétní oblasti [1]

Obecná struktura vstupně-výstupního lineárního modelu, pro systém s jedním vstupem u a výstupem y, ze které jsou posléze odvozeny její jednotlivé speciální případy, je tato:

$$A(q)y(t) = \sum_{i=1}^{nu} [B_i(q)/F_i(q)]u_i(t-nk_i) + [C(q)/D(q)]e(t) \tag{14}$$



Obr. 10 Obecný vstupně-výstupní lineární model

U_i zde reprezentuje čísla vstupů, A, B_i, C, D a F_i polynomy, které jsou funkcí q (nebo z). Obecná struktura je definována daným časovým zpožděním n_k a stupni uvedených polynomů. Dosazením hodnoty 1 za některý z polynomů $A(q)$, $B(q)$, $C(q)$, $D(q)$ můžeme dostat zjednodušené, odvozené struktury modelu. Například AR, ARX, ARMAX, Box-Jenkins, Output-error. Každá z těchto aplikací má oproti ostatním specifické výhody a nevýhody.

Následuje seznam speciálních, nejčastěji používaných modifikací (modelů), přeloženo z [21].

ARX (AutoRegressive with eXternal input):

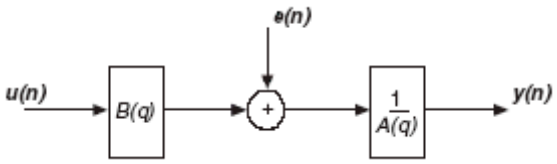
Model ARX, zobrazený na obr. 11, je nejjednodušší model zahrnující budící signál. Výpočet modelu ARX je nejefektivnější z metod výpočtu polynomů, protože je založen na řešení rovnic lineární regrese v analytické formě. Navíc každé řešení je unikátní. Jinými slovy, řešení vždy najde globální minimum ztrátové funkce. Proto je model ARX dobře využitelný zejména pro modely vyšších řádů.

Nevýhodou tohoto modelu je, že poruchy jsou součástí dynamiky systému. Přenos deterministické $G(q–1, è)$ a přenos stochastické $H(q–1, è)$ části systému má stejný počet pólů. Jestliže poruchová veličina systému $e(n)$ není bílý šum, může vazba mezi deterministickou a stochastickou dynamikou ovlivnit výpočet modelu. Zvýšením řádu modelu lze dosáhnout snížení jeho nepřesnosti (chyby

matematického modelu), zejména, pokud je nízký poměr signál/rušení (vyšší řád rovnice víc akcentuje rušení). Nicméně zvýšení řádu modelu může změnit některé jeho dynamické charakteristiky, jako je například stabilita. Velmi důležitou vlastností modelu ARX je stabilita predikce výstupní veličiny a to i v případě, pokud je systém nestabilní.

Následuje rovnice a schéma modelu ARX:

$$A(q)y(t) = B(q)u(t-nk) + e(t) \tag{15}$$

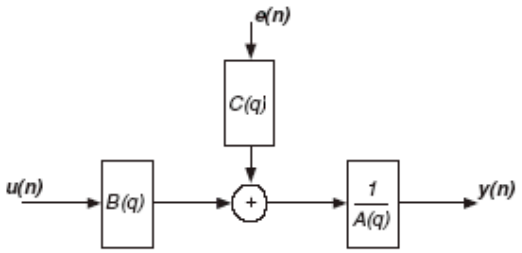


Obr. 11 Model ARX

ARMAX (AutoRegressive Moving Average with eXternal Input):

Na rozdíl od struktury modelu ARX, struktura ARMAX popisuje dynamické chování poruchových veličin. ARMAX je užitečný tam, kde přítomné poruchové veličiny významněji ovlivňují sledovaný proces a jsou měřitelné stejně, jako vstupní veličiny. Model ARMAX si s těmito poruchovými veličinami dokáže poradit lépe, než předchozí ARX.

$$A(q)y(t) = B(q)u(t-nk) + C(q)e(t) \tag{16}$$

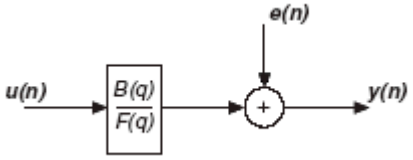


Obr. 12 Model ARMAX

OE (Output-Error model structure):

Tento model se využívá v případě, kdy šum působící na systém je bílý šum měření. Model struktury OE popisuje dynamiku systému odděleně. Pro modelování poruchové charakteristiky nejsou použity žádné parametry.

$$y(t) = [B(q)/F(q)]u(t-nk) + e(t) \tag{17}$$

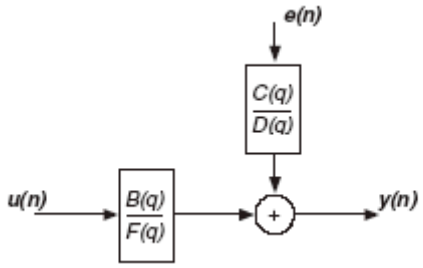


Obr. 13 Model Output-Error

BJ (Box-Jenkins Model):

Tato struktura modelu poskytuje kompletní model s poruchami, modelovanými odděleně od dynamiky systému.

$$y(t) = [B(q)/F(q)]u(t-nk) + [C(q)/D(q)]e(t) \tag{18}$$



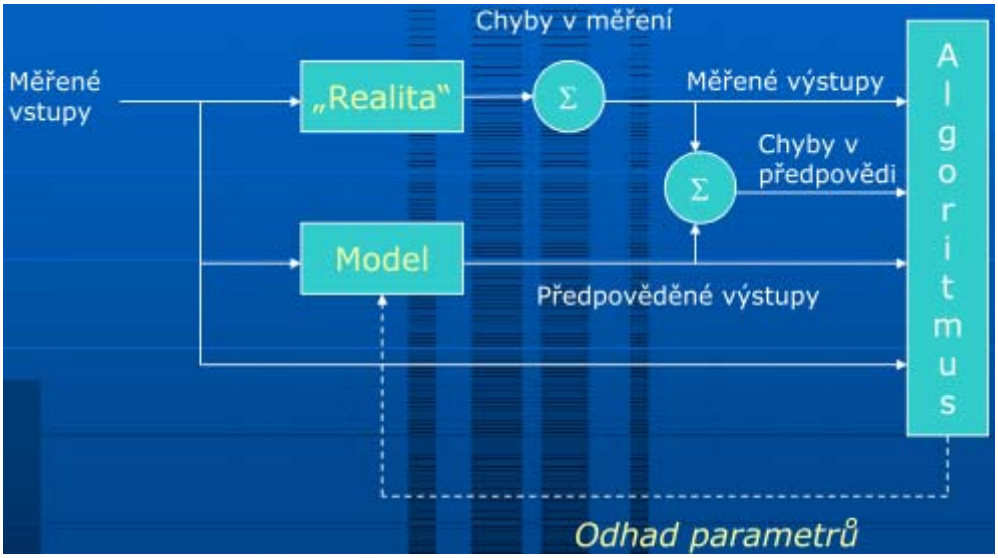
Obr. 14 Model Output-Error

Polynomy modelu jsou vhodné pro převod na diferenční rovnice. Například ARMAX může být zapsán následujícím způsobem:

$$y(t) + a_1 y(t-1) + \ldots + a_{na} y(t-na) = b_1 u(t-nk) + \ldots + b_{nb} u(t-nk-nb+1) + e(t) + c_1 e(t-1) + \ldots + c_{nc} e(t-nc) \tag{19}$$

2.4.4 Odhad parametrů modelu metodou nejmenších čtverců

Parametr, spolu s proměnnou a konstantou, patří mezi tři obecné typy veličin, které mohou v modelu vystupovat. Hodnota parametru musí byt určena (odhadnuta, vypočtena) pro každou jeho konkrétní aplikaci. Mluvíme-li o odhadu parametrů modelu, máme na mysli proces hledání hodnot parametrů, které vedou k co nejpřesnější shodě výsledku modelu s měřenými daty pomocí některé ze statistických metod [15]. Nejlépe tento proces (algoritmus) ilustruje následující schéma:



Obr.15 Odhad parametrů modelu [15]

Statistických metod odhadu parametrů modelu existuje několik, mezi nejznámější patří metoda nejmenších čtverců (MNČ), metoda zobecněných nejmenších čtverců (MZNČ) nebo metoda dvojstupňových nejmenších čtverců. Použití základní MNČ vyžaduje splnění určitých předpokladů o datech a regresním modelu. Pokud jsou některé z těchto předpokladů porušeny, musíme použít jiná kritéria, například kritérium MZNČ. Popišme si nyní základní a nejpoužívanější MNČ.

Metoda nejmenších čtverců

Tuto matematicko-statistickou metodu jako první použil a publikoval K.G.Gauss již v roce 1975. Jedná se o metodu sloužící k co nejpřesnějšímu proložení empiricky zjištěných hodnot pomocí vhodně zvolené aproximační funkce, tj. takové, která prochází v jistém smyslu (nejmenší možný součet obsahů čtverců) co nejblíže empiricky zjištěným hodnotám. Nejmenšími čtverci rozumíme druhé mocniny chyb odchylek (reziduí) mezi nalezeným řešením a původními naměřenými hodnotami. MNČ nám pomáhá určit takové hodnoty koeficientů řešení, aby součet reziduí byl pokud možno minimální.

Pro identifikaci soustavy, kterou se zabývá tato diplomová práce, jsme v první fázi použili výpočtový model ARX (Auto – Regressive with eXternal input), nabízený programem Matlab, který pro odhad parametrů modelu používá právě metodu nejmenších čtverců.

MNČ bývá často používána při regresní analýze k aproximaci zadaných hodnot. Regresní analýzu můžeme rozdělit na aproximaci přímkou, parabolou nebo polynomem. Následuje jejich stručný přehled. Protože byly pro identifikaci (v první fázi) pomocí modelu ARX použity vyšší řády polynomů, nejvíce prostoru dostává popis aproximace polynomem.

A) Aproximace přímkou

Rovnici $y = f(x,a,b) = ax + b$ lze proložit polynomem prvního řádu (tedy přímkou) pomocí koeficientů získaných těmito rovnicemi:

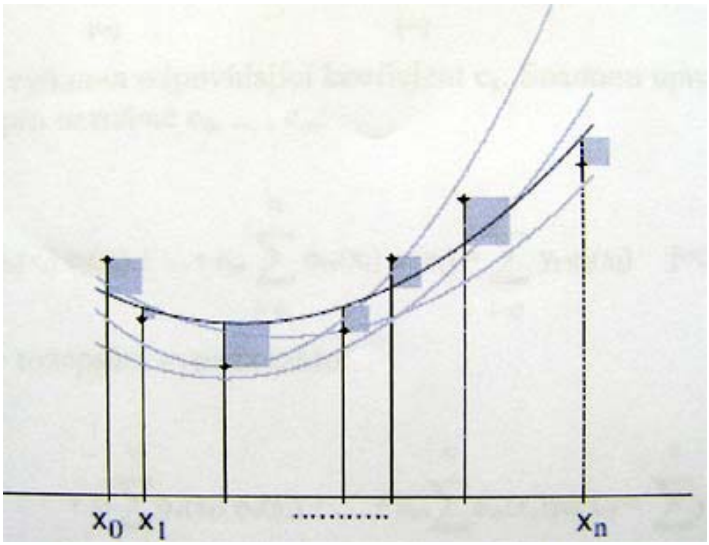
$$a = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - \left(\sum x_i\right)^2} \qquad b = \frac{\sum x_i^2 \sum y_i - \sum x_i \sum x_i y_i}{n\sum x_i^2 - \left(\sum x_i\right)^2} \tag{20,21}$$

B) Aproximace parabolou

Rovnici $y = f(x,a,b,c) = ax^2 + bx + c$ můžeme aproximovat polynomem druhého řádu (kvadratickou funkcí). Koeficienty *a,b,c* získáme řešením této soustavy lineárních rovnic:

$$\begin{aligned} a\sum x_i^4 &+ b\sum x_i^3 &+ c\sum x_i^2 &= \sum y_i x_i^2 \\ a\sum x_i^3 &+ b\sum x_i^2 &+ c\sum x_i &= \sum y_i x_i \\ a\sum x_i^2 &+ b\sum x_i &+ c\cdot n &= \sum y_i \end{aligned} \tag{22,23,24}$$

Aproximaci parabolou znázorňuje obrázek 16:



Obr. 16 Princip MNČ na příkladu aproximace parabolou

C) Aproximace polynomem

Pro aproximaci *n* dvojic hodnot $[x_i,y_i]$ polynomem k-tého řádu. Předpokládáme, že polynom je menšího řádu, než je počet dvojic. V opačném případě by úloha měla nekonečně mnoho řešení. Hledaný polynom $P_k = p_0 + p_1 * x + \ldots + p_k * x^k$ musí mít takové koeficienty $P_0,\ldots P_k$, aby součet *S* byl minimální (viz vzorec 27):

$$p_0 + p_1 x_i + \ldots + p_k x_i^k \longrightarrow y_i \tag{25}$$

$$e_i = p_0 + p_1 x_i + \ldots + p_k x_i^k - y_i \tag{26}$$

$$S = \sum_{i=1}^{n} e_i^2 \longrightarrow min \tag{27}$$

1. způsob výpočtu: parciální derivace

Vychází z aproximace polynomem prvního a druhého řádu (přímkou a parabolou). Pomocí porovnání soustav lineárních rovnic je sestavena soustava pro aproximaci polynomem. Jejím řešením lze získat vektor hledaných koeficientů.

$$\begin{bmatrix} \sum x_i^{2k} & \cdots & \sum x_i^{k+1} & \sum x_i^k \\ \vdots & \ddots & \vdots & \vdots \\ \sum x_i^{k+1} & \cdots & \sum x_i^2 & \sum x_i \\ \sum x_i^k & \cdots & \sum x_i & n \end{bmatrix} \cdot \begin{bmatrix} p_k \\ \vdots \\ p_1 \\ p_0 \end{bmatrix} = \begin{bmatrix} \sum y_i x_i^k \\ \vdots \\ \sum y_i x_i \\ \sum y_i \end{bmatrix} \tag{28}$$

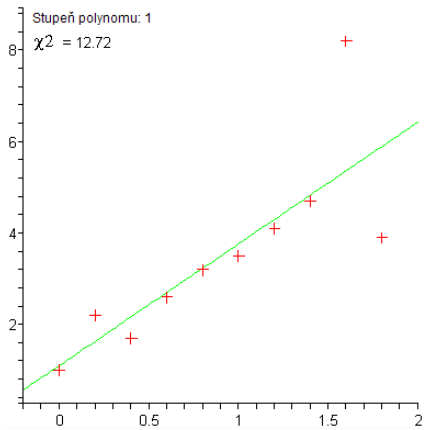
2. způsob výpočtu: přeurčená soustava rovnic

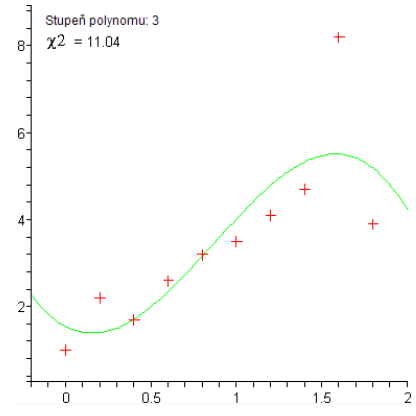
Tento způsob předpokládá přeurčenou soustavu rovnic s obdélníkovou maticí A. Řeší se metodou nejmenších čtverců.

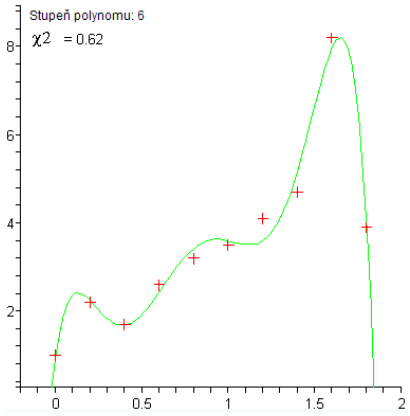
$$\mathbf{A} = \begin{bmatrix} 1 & x_1 & \cdots & x_1^k \\ 1 & x_2 & \dots & x_2^k \\ \vdots & \vdots & & \vdots \\ 1 & x_n & \cdots & x_n^k \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} p_0 \\ p_1 \\ \vdots \\ p_k \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \mathbf{e} = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix}$$

Hledáme takové hodnoty neznámého vektoru **x**, aby velikost vektoru **e** byla minimální. Úlohu můžeme převést na problém **Ax** – **b** = **e**. Poté koeficienty polynomu můžeme vypočítat podle vztahu: $\mathbf{x} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}$

Následuje ukázka aproximace zadaných bodů polynomy několika vybraných řádů (stupňů):







Obr. 17 Srovnání různých stupňů polynomů [převzato z http://commons.wikimedia.org]

Jak je z obrázků patrno, se vzrůstajícím stupněm polynomu roste kvalita aproximace, vyjádřená klesající hodnotou součtu čtverců odchylek S^2. Přestože lineární aproximace (polynom 1. řádu) je v této vyšetřované úloze relativně přesná (protože jsou vhodně zvolené body), polynom 3. řádu je už znatelně lepší a polynom 6. řádu dává už velice kvalitní výsledek. V praxi se však může stát, že lepší výsledek dává i nižší stupeň polynomu. Záleží vždy na charakteru konkrétní úlohy. Vyšší stupně polynomu, ač mohou být přesnější, jsou rovněž výpočtově náročnější.

Pozn.: Pro možnost praktického vyzkoušení a lepšího pochopení problematiky byl sestaven program pro výpočet aproximačního polynomu pomocí metody nejmenších čtverců, napsaný v jazyce Borland Pascal. Program tvoří přílohu [C] této diplomové práce. Vstupem jsou dvojice x_i, y_i a stupně polynomu v modelu, výstupem koeficienty modelu a tabulka hodnot, včetně nalezení minima (nejmenšího čtverce). Spočtené reziduum $(f(x_i) - y_i)^2 = 0$, tj. program spočítal takové hodnoty parametrů řešení, že získaná funkce prochází všemi zadanými body.

2.4.5 Ověřování a volba optimálního modelu

V kap. 2.4 jsme si uvedli, že posledním krokem identifikačního postupu je ověření (verifikace), s jakou přesností získaný model dokáže popsat chování (funkční závislosti) skutečného systému. Tuto míru shody je nutno nějak kvantifikovat, abychom mohli výsledky různých modelů, nebo stejných modelů s různými parametry vzájemně porovnat a vybrat ten nejlepší. V programu Matlab, jehož modul *System Identification Toolbox* jsme v první fázi využili pro získání přenosů pomocí modelu ARX, je pro účel verifikace modelu využíván tzv. FIT index, jehož definiční vztah je následující:

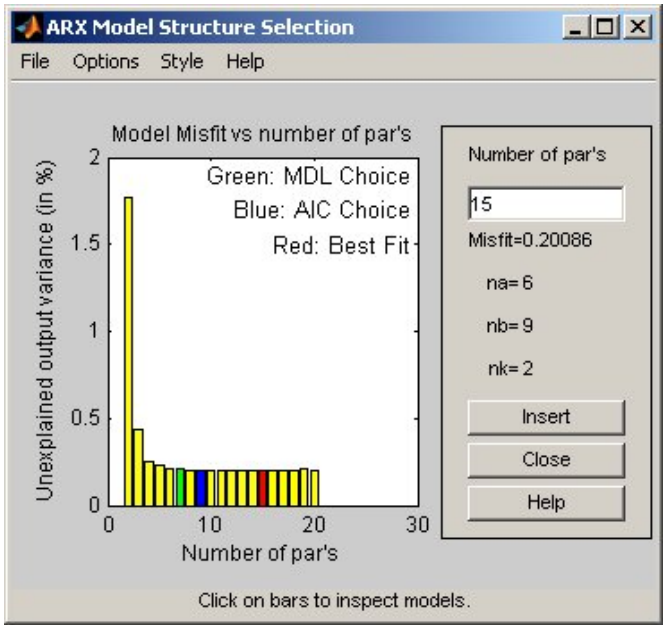
$$\text{FIT index} = \left(1 - \frac{|y - \hat{y}|}{|y - \bar{y}|}\right) * 100\ [\%] \qquad (29)$$

kde y je změřený (odhadnutý) výstup, $\hat{y}$ je simulovaný (předpokládaný) výstup modelu a $\bar{y}$ je střední hodnota y. Můžeme říci, že nejlepší model je ten, jehož hodnota FIT indexu se nejvíc blíží 100%. To znamená takový model, u kterého součet všech odchylek mezi reálným a aproximovaným průběhem je nejmenší, a proto nejlépe popisuje skutečný průběh sledované závislosti. Z definice funkce FIT index vyplývá,

že je možné, aby nabývala záporných hodnot. Záporný (negativní) FIT je horší než nulový (FIT minus 100% znamená dokonale opačný/inverzní průběh funkce) a může být způsobem jedním z následujících důvodů:

- Výpočetní algoritmus selhal ve své konvergenci. Naopak divergoval.
- Nevhodně zvolený odhad parametrů modelu.
- Data pro validaci nebyla vybrána vhodně, nebo byla předzpracována zásadně jiným způsobem, než data k výpočtu (estimation data).

Hodnotu FIT index počítá Matlab automaticky, v procesu identifikace ji nabídne ve třech formách, kdy pro jednotlivé kombinace polynomů (v ARX modelu je n_a počet pólů, n_b je hodnota parametru b a n_k je zpoždění) nabídne následující výběr struktury modelu ARX:



Obr. 18 Výběr struktury modelu ARX

Osa x udává součet řádů jednotlivých modelů ($n_a+n_b+n_k$), na ose y vidíme v procentech velikost odchylky na výstupu, kterou se daným modelem nepodařilo vysvětlit. Jednotlivé doporučené struktury ARX modelu jsou odlišeny barevně. Význam barev je následující:

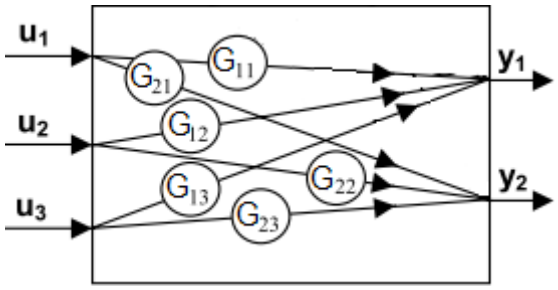
- Červená — Nejlepší hodnota FIT indexu vypočtená jako minimalizace čtverců odchylek mezi modelem a validačními daty. ARX vypočte model z estimation dat metodou nejmenších čtverců a poté testuje tento model na validačních datech s využitím uvedeného vzorce pro FIT index.
- Zelená — Nejlepší hodnota FIT indexu vypočtená pomocí Rissanenova MDL kritéria.
- Modrá — Nejlepší hodnota FIT indexu vypočtená pomocí Akaikeho AIC kritéria.

Rozbor kritérií MDL a AIC je mimo rozsah této diplomové práce. Podrobnější informace lze nalézt v literatuře [19,20]

2.4.6 Mnohorozměrové regulační obvody

V případě mnohorozměrových regulačních obvodů (MIMO – multi-input, multi-output) je výhodné popisovat systém pomocí matice přenosů. Jedná se o soustavy, které obsahují více regulovaných (výstupních) i více řídicích (vstupních) veličin, které je ovlivňují. Na rozdíl od SISO (single-input, single-output) nejsou u mnohorozměrových regulačních obvodů jednotlivé vazby vstup/výstup vzájemně nezávislé (výstup – regulovaná veličina je ovlivněna příspěvkem od všech příslušných přenosů) a jejich řešení je tudíž složitější.

Obr.19 znázorňuje na příkladu vztahy mezi třemi vstupními (u) a dvěmi výstupními (y) veličinami prostřednictvím přenosů (G).



Obr. 19 Schéma mnohorozměrného obvodu [7]

V označení přenosu G levý číselný index představuje číslo regulované veličiny (výstupu), pravý číselný index číslo řídicí veličiny (vstupu). Jednotlivé přenosy G_{ij} poté vyjádříme takto ($Y(s)$ a $U(s)$ jsou Laplaceovy obrazy jednotlivých proměnných):

$$G_{11}(s) = \frac{Y_1(s)}{U_1(s)};\ G_{12}(s) = \frac{Y_1(s)}{U_2(s)};\ G_{13}(s) = \frac{Y_1(s)}{U_3(s)};\ G_{21}(s) = \frac{Y_2(s)}{U_1(s)};\ G_{22}(s) = \frac{Y_2(s)}{U_2(s)};$$

$$G_{23}(s) = \frac{Y_2(s)}{U_3(s)} \qquad (30)$$

Známe-li všechny potřebné přenosy, můžeme sestavit rovnice regulované soustavy, buďto ve složkovém tvaru, nebo pomocí sloupcových matic.

Složkový tvar pro jednotlivé $Y_i(s)$, kde i je 1 až n, vypadá takto:

$$Y_1(s) = G_{11}(s)\,U_1(s) + G_{12}(s)\,U_2(s) + \ldots + G_{1n}\,U_n(s) \qquad (31)$$

$$Y_n(s) = G_{n1}(s)\,U_1(s) + G_{n2}(s)\,U_2(s) + \ldots + G_{mn}\,U_n(s) \qquad (32)$$

Maticový zápis pro Laplaceovy obrazy regulované (Y) a akční (U) veličiny má tuto podobu:

$$\mathbf{Y}(s) = \begin{bmatrix} Y_1(s) \\ \dots \\ Y_n(s) \end{bmatrix}, \mathbf{U}(s) = \begin{bmatrix} U_1(s) \\ \dots \\ U_n(s) \end{bmatrix} \tag{33}$$

Celkovou matici přenosu regulované soustavy sestavíme následujícím způsobem:

$$\mathbf{G}(s) = \begin{bmatrix} G_{11}(s) & G_{12}(s) & \dots & G_{1n}(s) \\ G_{21}(s) & G_{22}(s) & \dots & G_{2n}(s) \\ \dots & \dots & \dots & \dots \\ G_{n1}(s) & G_{n2}(s) & \dots & G_{nm}(s) \end{bmatrix} \tag{34}$$

Maticový tvar rovnice regulované soustavy je poté následující:

$$\mathbf{Y}(s) = \mathbf{G}(s)\,\mathbf{U}(s) \tag{35}$$

2.5. Softwarová podpora

Zatímco identifikaci jednoduchých lineárních systémů bez setrvačností a dopravních zpoždění je možné realizovat ručně, v případě složitějších systémů se neobejdeme bez podpory vhodného uživatelského softwaru. Představme si proto program Matlab, jehož nadstavba SIT představuje jeden z nejefektivnější a nejpoužívanějších nástrojů pro identifikaci systémů, který je v současnosti dostupný na trhu.

2.5.1 Matlab

Literatura [6] definuje *Matlab* jako programové prostředí a skriptovací programovací jazyk pro vědeckotechnické numerické výpočty, modelování, návrhy algoritmů, počítačové simulace, analýzu a prezentaci dat, měření a zpracování signálů, návrhy řídicích a komunikačních systémů. Původně Matlab vznikl jako interaktivní nadstavba pro usnadnění práce s knihovnami *LINPACK* a *EISPACK* pro práci s maticemi. Základním typem byly matice, které na rozdíl od většiny jiných systémů a jazyků nevyžadovaly nastavování dimenzí.

Současný *Matlab* je mnohem více než jen nadstavbou maticové knihovny. Systém obsahuje vlastní interpret jazyku Matlab, ve kterém lze připravit jak dávkové soubory, tak definovat i nové funkce. Tyto funkce mohou být interpretovány buď přímo z textové podoby souborů nazývaných *m-file* nebo z předzpracované podoby *p-file*. Systém dále umožňuje přidávat moduly (soubory *mex-file*) zkompilované do strojového kódu procesoru. Jazykem zdrojových souborů může být jazyk C, C++, Fortran nebo po použití *mcc* (Matlab to C-Compiler) i funkce uložená v *m-file*. V prostředích, kde je možný grafický výstup, je k dispozici velmi silná podpora pro tvorbu uživatelského prostředí a vizualizaci dat. Základní funkce umožňují vizualizaci 2D a 3D dat v grafech s množstvím volitelných parametrů, které lze po doplnění legendou a popisy os snadno vytisknout. Systém umožňuje i tvorbu vlastních dialogů a oken s kombinacemi grafů, tlačítek, listů a dalších objektů, kterým mohou být

přiřazeny volané funkce v *m-file*. Dobře je obslouženo i polohovací zařízení - myš. Pro tvorbu samostatně spustitelných aplikací je k dispozici *Matlab library*.

Asi nejdůležitější částí instalace *Matlab* jsou „knihovny“funkcí (ve skutečnosti adresáře s *m* a *mex* soubory), které jsou nazývány toolboxy. Toolboxy obsahují vždy uceleným způsobem, včetně dokumentace a příkladů, zpracovaný určitý obor numerické matematiky, analytické matematiky, statistiky, systémového přístupu k regulacím a další obory, ve kterých nachází Matlab uplatnění.

2.5.2 System Identification Toolbox (SIT)

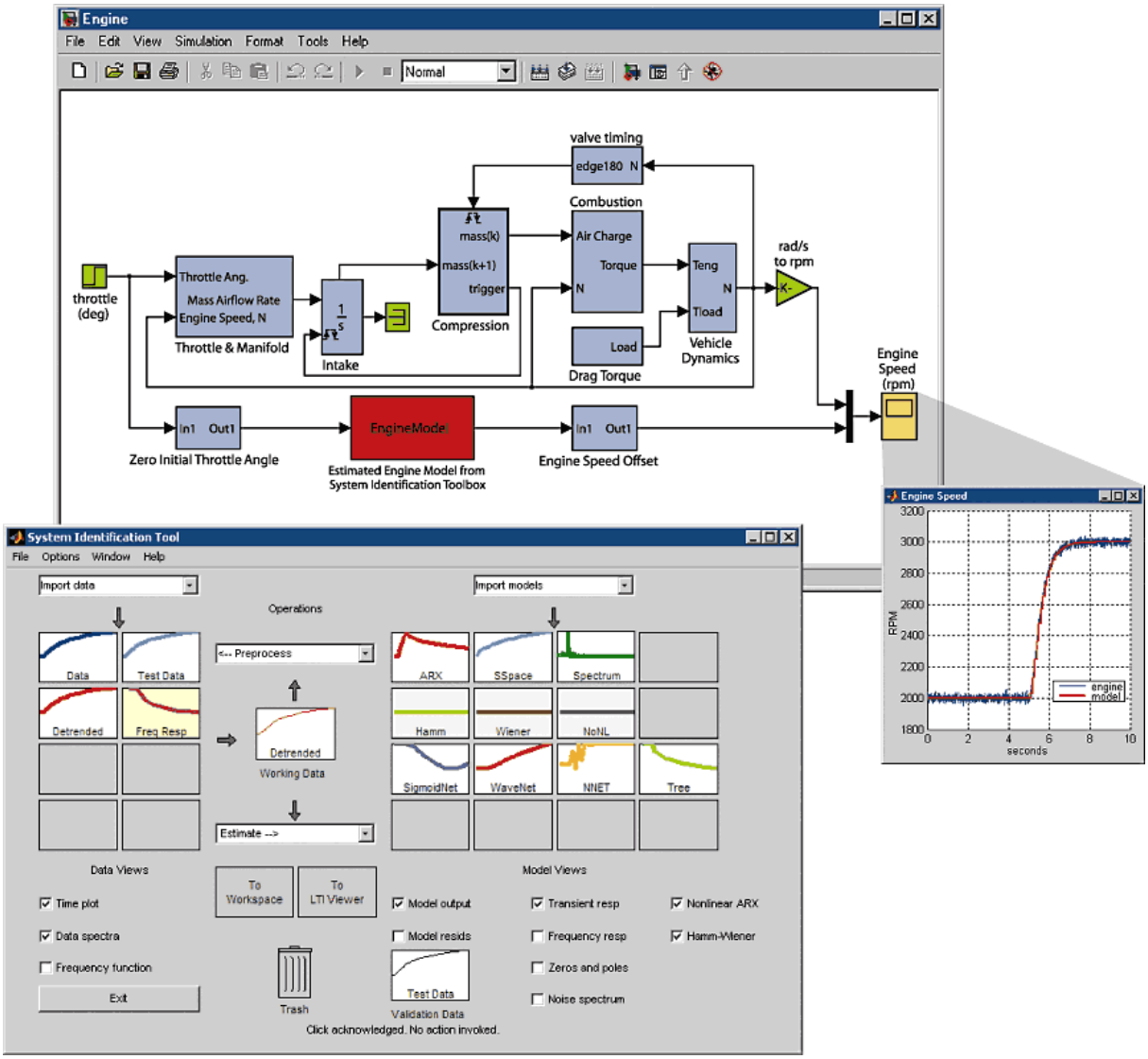
SIT je jeden z předprogramovaných toolboxů v nabídce *Matlab*. Je určen pro vytváření matematických modelů systémů z naměřených dat. Poskytuje nástroje pro vytvoření matematických modelů dynamických systémů, založené na sledování vstupních a výstupních dat. Pro práci využívá grafické uživatelské rozhraní, které usnadňuje práci při organizaci dat a modelů.

Nástroje pro identifikaci poskytované v tomto toolboxu jsou určené pro užití v oblasti od návrhu regulátoru a zpracování signálu až k časové a vibrační analýze. [6]

2.5.3 Simulink

[6] Simulink je nadstavba programu *Matlab*, která umožňuje simulaci (viz kapitola 2.1.1) smíšených systémů obsahujících spojité části, diskrétní části i s různými periodami vzorkování a s posunutými okamžiky vzorkování. Je schopen simulovat i nelineární bloky a aproximovat chování systémů obsahujících algebraické smyčky, na které ovšem před simulací upozorňuje. Dynamické vlastnosti lineárních částí lze popisovat komplexními přenosy, maticemi systémů nebo rovnou použít bloky reprezentující přímo sčítání, integraci, diferenci, násobení konstantou a další elementární operace. V knihovně nelineárních bloků jsou předdefinovány paměťové bloky, přepínače, reléové charakteristiky, násobení a dělení signálů, zdroje hodinových impulsů a mnoho dalších.

Obr.20 ukazuje grafický uživatelský interface (GUI) *System Identification Toolbox* v kombinaci se *Simulinkem*:



Obr. 20 Uživatelské rozhraní System Identification Toolbox a Simulink [převzato z http://www.cybernet.co.jp/matlab/]

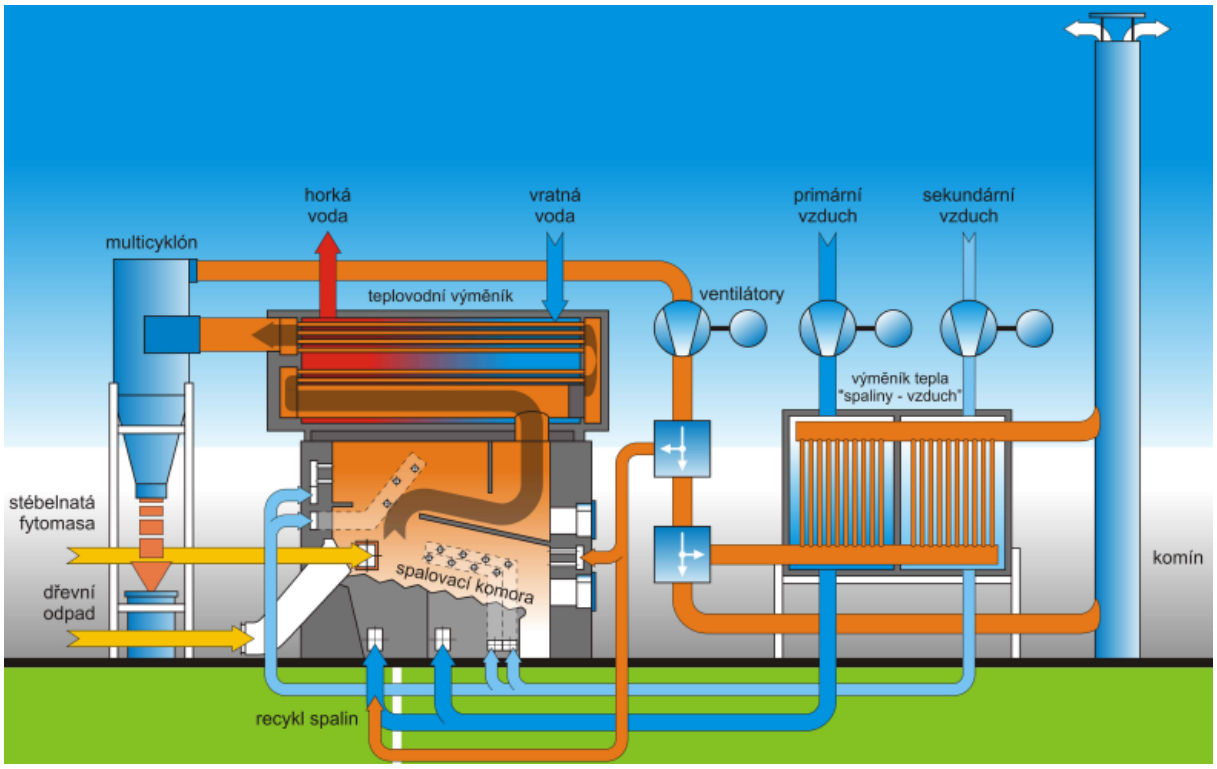
3 EXPERIMENTÁLNÍ JEDNOTKA PRO SPALOVÁNÍ BIOMASY

V souladu se zadáním diplomové práce byl popsaný identifikační postup aplikován na reálné technologii – experimentální jednotce o výkonu 1MW. Pojďme se nyní s touto jednotkou blíže seznámit. Jedná se o evropský projekt, který je zaštítěn Ústavem procesního a ekologického inženýrství Vysokého učení technického v Brně. Hlavním cílem projektu bylo vyvinout a realizovat prototyp ryze české moderní technologie pro energetické využití biomasy a fytomasy (konkrétně ve formě prototypu technologické jednotky, jejíž základ tvoří experimentální teplovodní kotelna středních výkonů) s vysokou termickou účinností, jež umožní testování v provozním měřítku. Předpokládaným palivem jsou různé druhy biomasy od pilin a standardní dřevní štěpky až po cíleně pěstovanou energetickou fytomasu, dodávanou ke kotelně ve formě slisovaných balíků. Účelem je samozřejmě dodávka topné vody k vytápění přidružené tovární haly. [18]

Jednotka stojí v areálu bývalého cukrovaru v Kojetíně. Vyznačuje se například:

- Oddělenými trasami pro biomasu stepkovitého charakteru a pro stébelnatou fytomasu.
- Recirkulací spalin.
- Rekuperací tepla pro předehřev spalovacího vzduchu.
- Umožněním dlouhodobého testování v provozním měřítku. [příloha E]

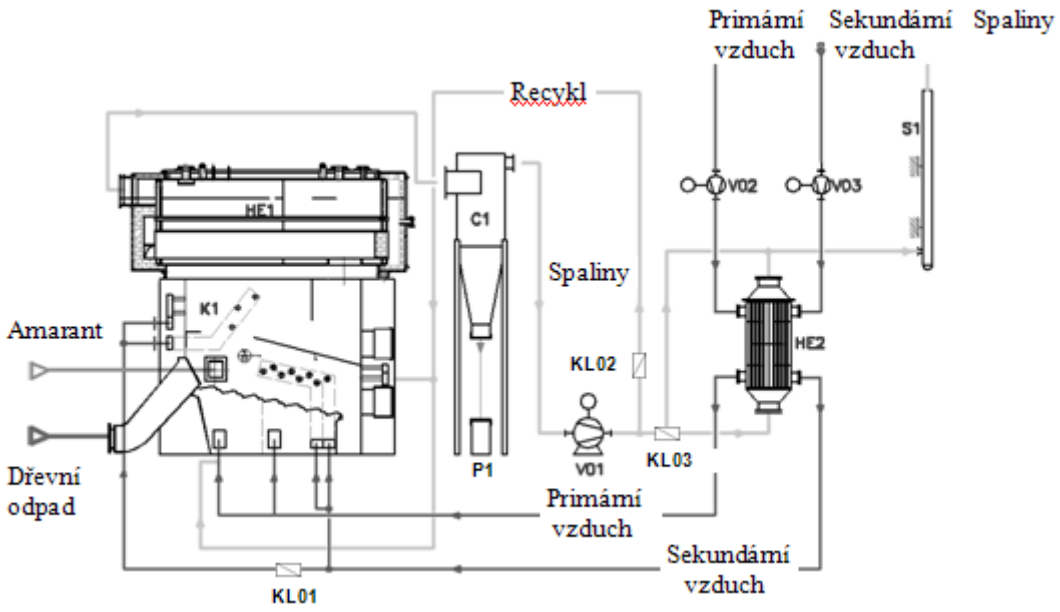
Pro lepší představu si uveďme její grafický model:



Obr. 21 Grafický model identifikované jednotky pro spalování biomasy, [příloha E]

3.1 Technologie

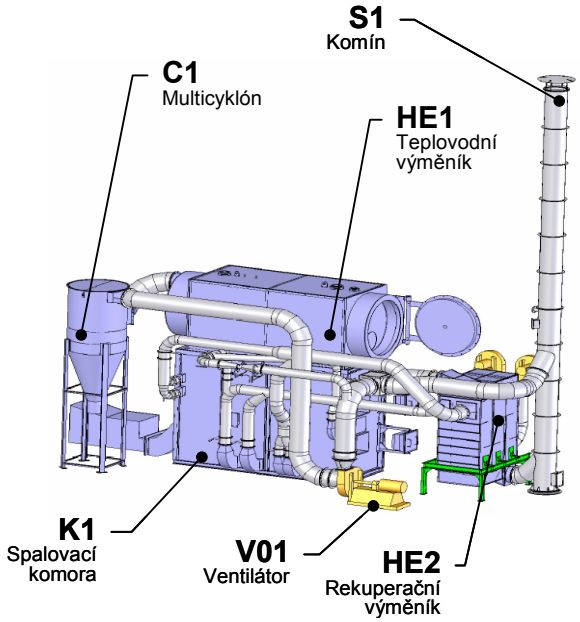
Technologické schéma jednotky a 3D model prototypu jsou zobrazeny na následujících obrázcích (obr. 22, 23).



K1 – Spalovací komora	V01 – Spalinový ventilátor
HE1 – Teplovodní výměník	V02 – Ventilátor primárního vzduchu
HE2 – Rekuperační výměník	V03 – Ventilátor sekundárního vzduchu
C1 – Multicyklon	KL01 – Klapka sekundárního vzduchu
S1 – Komín	KL02 – Klapka recirkulace spalin
P1 – Popelnice	KL03 – Klapka rekuperace spalin

Obr. 22 Zjednodušené technologické schéma jednotky [citace]

[12] Systém dopravy paliva do kotle se skládá ze dvou oddělených cest – pro dřevní biomasu a fytomasu. Vlastní spalování probíhá na šikmém hydraulickém roštu a v sekundární komoře uvnitř spalovací komory (K1). Spalovací vzduch primární i sekundární je předehřát ve speciálním rekuperačním výměníku instalovaném na trase spalin (HE2). Vzniklé spaliny pak proudí teplo-směnným svazkem trubek ve výměníkové části kotle (HE1), ochlazené spaliny jsou dočištěny od jemného popílku v multicyklonu (C1). Za multicyklonem je umístěn spalinový ventilátor (V01), který je jediným hnacím zařízením celé spalinové trasy od spalovací komory až po komín (S1). Za ventilátorem je provedena odbočka tzv. recyklu spalin. Pomocí tohoto recyklu je část spalin z výstupu z kotle přivedena zpět do spalovací komory kotle, čímž se zvyšuje pyrolýzní efekt, palivo se před zapálením rychleji vysuší a zvyšuje se tak celková účinnost kotle. Za odbočkou recyklu spalin následuje předehřev sekundárního spalovacího vzduchu ve výměníku (HE2). V tomto výměníku je využito tepla spalin do maximální možné míry, aby ještě nedocházelo k přílišnému zalepování spalinové strany výměníku. Snahou je ochladit spaliny na co možná nejnižší teplotu a tím eliminovat komínové ztráty. Spaliny ochlazené ve výměníku následně ústí do komína (S1).



Obr. 23 3D model prototypu jednotky pro energetické využití různých druhů biomasy a fytomasy []

Mezi jednotlivými vstupními a výstupními veličinami existuje různá míra závislosti. Výkon kotle je nejvíce (nejsilněji) ovlivněn dodaným množstvím paliva a průtokem primárního vzduchu. Koncentrace kyslíku ve spalinách je silně ovlivněna primárním, sekundárním i recyklovaným vzduchem. Teplotu ve spalovací komoře nejvíc ovlivňuje průtok (množství) sekundárního a recyklovaného vzduchu. V případě recyklovaného vzduchu navíc záleží, zda je předehřátý nebo nikoliv. Předehřátý vzduch teplotu ve spalovací komoře zvyšuje víc.

3.2 Řídicí systém

Jak je uvedeno v [11], v případě prototypu experimentálního zařízení (viz obr. 22) byla jako regulovaná veličina zvolena výstupní teplota média (vody) z kotle. Při projektovaném teplotním spádu je tato teplota 90°C. Žádanou hodnotu lze měnit na operátorském panelu regulátoru. Regulace přísunu paliva závislá pouze na výstupní teplotě média může mít (oproti regulaci podle odběru spotřebitele) nevýhodu v prodloužení doby reakce na změnu zatížení. V našem případě je ovšem jednotka použita pro vytápění výrobní haly, kde lze odebíraný tepelný výkon považovat za konstantní.

Podle výkonu podavače se lineárně řídí otáčky ventilátoru primárního vzduchu V02 (viz obr. 22). Právě volba vhodného poměru otáček ventilátoru k výkonu podavače je určující pro dosažení žádaných emisních poměrů. Do budoucna však bude vhodné doplnit systém kontinuální analýzou spalin. Díky informaci o hodnotách O_2 a CO ve spalinách bude možné lépe řídit otáčky ventilátoru sekundárního vzduchu. Ten je v současné době ovládán podle teploty ve spalovací komoře.

Vedle této základní regulace je systém tvořen také dalšími smyčkami, bez jejichž přítomnosti by kotel jako tepelný zdroj nemohl fungovat. Jde o regulaci:

- teploty vratné vody do kotle směšovacím ventilem (na konstantní hodnotu)
- tlaku v systému doplňováním a odpouštěním vody (na konstantní hodnotu)
- teploty vody ústředního topení směšovacím ventilem (ekvitermní)

Jak dále uvádí [11], pro řízení kotelny byl použit modulární řídicí systémem (ŘS) ADiS 167. ŘS vykonává veškeré úkony zadané obsluhou a automatické algoritmy. Program realizující regulaci jednotky byl vytvořen ve vývojovém prostředí PSP3. Zařízení je vybaveno GSM modemem, který prostřednictvím SMS zpráv (poruchy, stavy) dálkově monitoruje provoz technologie.

Vedle zvýšení termické účinnosti přináší efektivní řízení předehřevu vzduchu a recirkulace spalin také možnost spalovat různé druhy biomasy a fytomasy při zachování konstantních spalovacích podmínek.

Předehřev vzduchu

Předehřev vzduchu zvyšuje účinnost zařízení zpětným využitím tepla obsaženého ve spalinách k ohřevu vzduchu přiváděného do spalovací komory. Pro předehřev vzduchu bylo použito dvoustavové regulace realizované uzavírací pneumatickou klapkou KL03. Klapka se zavírá, pokud je teplota primárního i sekundárního vzduchu menší než žádaná mez zapnutí. Po jejím uzavření prochází vzduch přes rekuperační výměník. Klapka se otevírá, pokud je některá z teplot větší než mez vypnutí. Je-li klapka otevřena, spaliny proudí zkratem přímo do komína S1. Míra předehřevu je dána pouze teplotou spalin.[11]

Recirkulace spalin

Účinnost je příznivě ovlivňována snižováním koncentrace kyslíku ve spalinách vystupujících z kotle. Recirkulace spalin je zpětné vracení určitého množství spalin do prostoru spalovací komory. Tím je kyslík obsažený ve spalinách zpětně využit a účinnost stoupá. Řízení recirkulace spalin probíhá spojitě. Klapka určující míru recirkulace KL02 je ovládána lineárně dle nastavených teplot pro její minimální a maximální polohu. Optimální hodnoty těchto teplot pro jednotlivá paliva budou předmětem dalšího výzkumu.[11]

3.3 Rozbor funkčních vazeb a závislostí

Tato podkapitola detailněji popisuje důležité vstupní a výstupní veličiny, jejich účel, možnosti jejich regulace i to, jak jsou na sobě vzájemně závislé. Tyto informace nám pomáhají lépe pochopit fungování kotle a možnosti a účel jeho řízení.

Výkon podavače je veličina, která určuje periodu pohybu hydraulického dopravníku paliva a udává tak množství paliva, které je přiváděno do prostoru spalovací komory. Jako taková má přirozeně zásadní vliv na provoz kotle. Svou roli

samozřejmě hraje druh paliva, jeho kvalita a vlhkost. Charakteristiky paliva byly ověřeny jednorázově, palivovým rozborem.

Výkon kotle. Tuto veličinu je důležité sledovat z hlediska vyhodnoceni celkové efektivity soustavy. Naším cílem je samozřejmě udržet výkon kotle na co nejlepší, pokud možno stabilní úrovni, zejména s ohledem na teplotu výstupní (topné) vody. Byla-li tato teplota dosažena, je žádoucí výkon kotle snížit (nemá cenu spalovat další palivo, když je požadovaná teplota vody dosažena). Můžeme jej regulovat pomocí množství paliva, průtoku primárního vzduchu, sekundární vzduch a recyklovaného vzduchu. Tyto veličiny by měly výkon kotle zvyšovat (více paliva, podpora procesu hoření). Výkon kotle je dán vztahem

$$P = \frac{\rho \cdot V \cdot c \cdot \left(T_{vst} - T_{výst}\right)}{3600} \quad [\mathrm{W}] \tag{36}$$

kde: m - hmotnostní průtok [kg·h^{-1}], ρ - měrná hmotnost topného média [kg·m^{-3}], V – objemový průtok topného média [m^{3}·h^{-1}], c – měrná tepelná kapacita topného média [J · kg^{-1} ·K^{-1}], (T_{vst}-$T_{výst}$) – rozdíl mezi vstupní a výstupní teplotou [K], tzv. teplotní spád.

Primární a sekundární vzduch je vháněn do spalovací komory příslušnými ventilátory (V02 pro primární, V03 pro sekundární – viz schéma soustavy na obr.22), recyklovaný vzduch řídíme pomocí klapky Kl01. Ovlivňujeme jimi samotný proces hoření. Za předpokladu dodávky paliva můžeme přísunem vzduchu zvýšit hoření a následně teploty nebo je naopak utlumit. Recyklovaný vzduch můžeme vhánět buď v teplotě okolí, tzn. bez předehřevu, nebo s předehřevem. Je zřejmé, že čím teplejší tento vzduch je, tím více podporuje proces hoření.

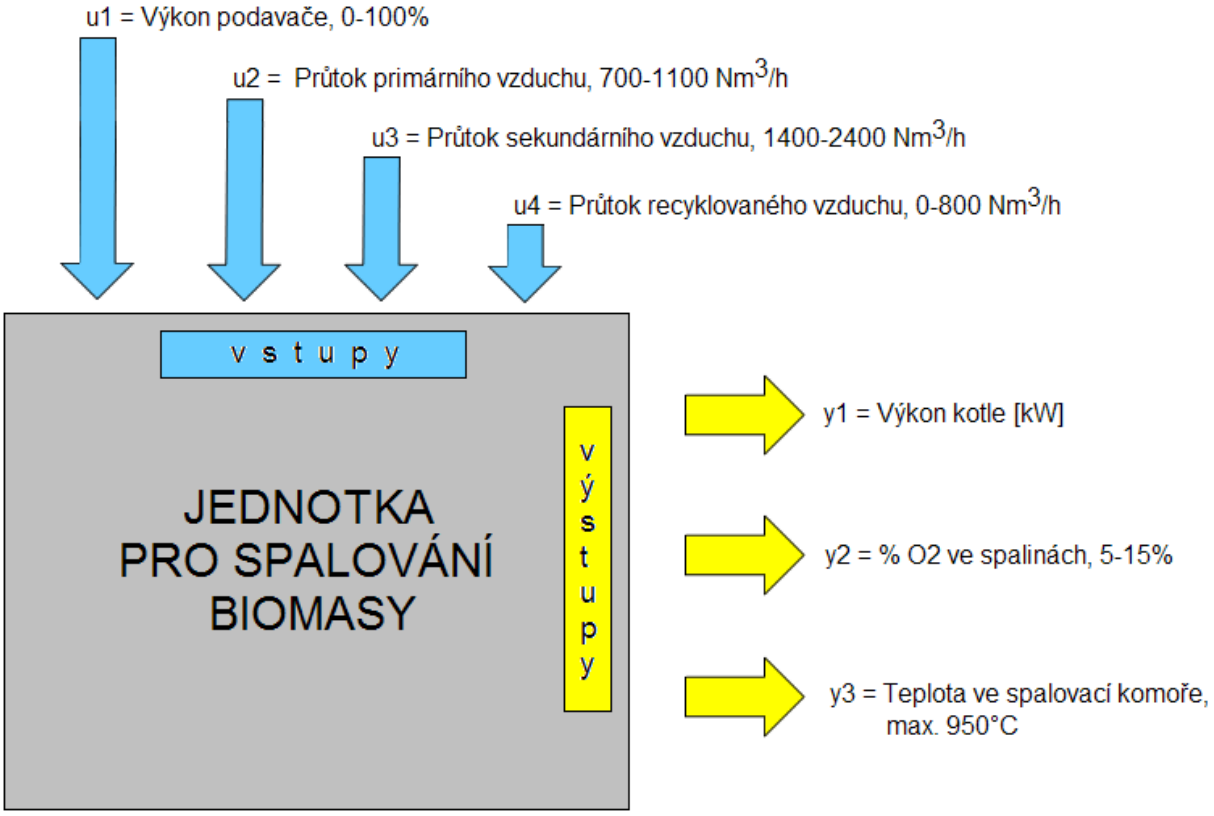
Procento kyslíku ve spalinách je hledisko důležité zejména z ohledu vyhodnocení celkové kvality spalování – je-li zbytkového kyslíku ve spalinách příliš, znamená to, že proces hoření není efektivní. Toto procento nám tedy říká, jak kvalitní je daný spalovací proces. Je-li splněna podmínka dokonalého spalování biomasy, pak můžeme snižování této koncentrace považovat za kladný jev, zvyšující celkovou účinnost systému. [12] Koncentraci kyslíku měříme pomocí měřícího čidla, tzv. lambda – sondy. Tuto koncentraci ovlivňuje primární, sekundární i recyklovaný vzduch. Můžeme ji, při současném udržení teploty ve spalovací komoře na požadované úrovni, snížit také použitím recirkulace spalin, tedy vracením určitého množství spalin do prostoru spalovací komory. Recirkulaci spalin technicky ovlivňujeme nastavením klapky KL02 – viz obr. 22.

Teplota ve spalovací komoře (jedná se o průměr teplot ze tří spalovacích komor, viz schéma kotle) tj. výstupní teplota spalin z kotle, je podstatně závislá na přísunu paliva. Závislost teploty ve spalovací komoře např. na primárním vzduchu je těžké odhadnout, protože buď přívod vzduchu ochladí prostor komory (málo paliva na roštu) nebo podpoří hoření a teplota se zvýší (dostatek paliva na roštu). Efekt primárního vzduchu je tedy závislý na dalším vstupu – výkonu podavače paliva.

4. IDENTIFIKACE PŘEDMĚTNÉ TECHNOLOGIE

4.1 Výběr procesních veličin pro identifikaci

Jako regulované (výstupní) veličiny vyšetřovaného systému byly zvoleny výkon kotle, koncentrace kyslíku ve spalinách a průměrná teplota ve spalovací komoře. Tyto veličiny poskytují ucelenou informaci o aktuálním provozním stavu jednotky. Jako řízené (vstupní) veličiny byly zvoleny výkon podavače a průtok primárního, sekundárního a recyklovaného vzduchu. Tyto veličiny jsme schopni technicky ovlivňovat a tím řídit provoz kotle. Celkově tedy bylo pro účely identifikace vybráno 7 veličin (4x vstup, 3x výstup), které dohromady dávají 12 přenosů. Následuje grafický přehled I/O (obr. 24) a detailnější popis v tabulce 4:



Obr. 24 Grafický přehled vstupních a výstupních veličin

Vstupy	Požadované rozsahy
Výkon podavače	0-100%
Průtok primárního vzduchu	700-1100 Nm³/h
Průtok sekundárního vzduchu	1400-2400 Nm³/h
Průtok recyklovaného vzduchu	0-800 Nm³/h
Výstupy	
Výkon kotle	40-100%
Procento O_2 ve spalinách	5-15%
Teplota ve spalovací komoře	Max. : 950°C

Tab4. Přehled zvolených vstupních a výstupních veličin

Celkový chod kotle samozřejmě ovlivňují další podmínky, které nejsou v našich přenosech explicitně zahrnuty, ale mají svůj důležitý vliv. Jedná se například o teplotu okolí. V době našeho měření byla teplota v hale, kde kotel stojí, přibližně 15 °C, venkovní teplota asi 2,2 °C. Dalším faktorem byl typ použitého paliva – použili jsme dřevní štěpku s výhřevností 13,8 MJ/kg a vlhkostí 28%. Spalovací vzduch vháněný do spalovací komory nebyl předehříván. Kotel byl uveden do provozu po delší odstávce. Pracovní bod výkonu jednotky byl stanoven na přibližně 420kW, což je hodnota na dolní hranici regulovatelnosti této technologie. Hodnoty dalších významných charakteristik prostředí a systému, jak jsou uvedeny v literatuře [10], shrnuje následující tabulka:

Parametr	Hodnota	Jednotka
Atmosférický tlak	101,3	[kPa]
Relativní vlhkost vzduchu	0,50	[-]
Teplota ve spalovací komoře	730	[°C]
Teplota spalin na vstupu do komína - vyhodit	63,5	[°C]
Obsah O_2 ve spalinách	11,4	%

Tab.5 Vybrané podmínky měření

4.2 Návrh experimentálního měření a jeho realizace

Data k identifikaci jsme získali měřením za normálního provozu kotle, bez odpojení regulátorů (což nebylo technicky možné), vhodnou změnou vybraných vstupních veličin. Důležitým faktorem byla rovněž perioda vzorkování, tedy sběru dat. Po odborných konzultacích jsme stanovili vzorkovací frekvenci 15 sekund, což by v našem případě měl být dostatečně jemný interval, reflektující změny sledovaných veličin.

Sledované veličiny jsme samozřejmě mohli řídit jen technicky dostupným způsobem. Přísun paliva byl regulován mechanickým podavačem paliva, tj. hydraulickým zařízením, které nejprve nahrne palivo do přívodního koryta a poté jej hrne do kotle. U podavače se dá nastavit jeho rychlost, případně je možné jej úplně vypnout. Pokud je zapnutý, je množství dodávaného paliva víceméně konstantní a je dodáváno do kotle v nastavené periodě. Průtok primárního a sekundárního vzduchu je technicky možné ovlivnit nepřímo, přes změnu otáček příslušných ventilátorů, tj. otáček primárního a sekundárního ventilátoru. Řízenou změnou těchto otáček jsme provedli skokové změny průtoku primárního a sekundárního vzduchu ve dvou opačných směrech a zaznamenali změny průběhu sledovaných veličin.

Uveďme si přehled skokových změn na datech z 3.12.2007 pro jednotlivé řídicí veličiny (viz příloha *[A]*).

Změna výkonu podavače (Tab.6)

Čas	Směr skokové změny	Velikost
13:13	nahoru	60% -> 100%
12:46	dolů	100% -> 33%

Průtok primárního vzduchu (Tab.7)

Čas	Směr skokové změny	Velikost
12:30	dolů	65% -> 50%
12:46	nahoru	50% -> 80%

Průtok sekundárního vzduchu (Tab.8)

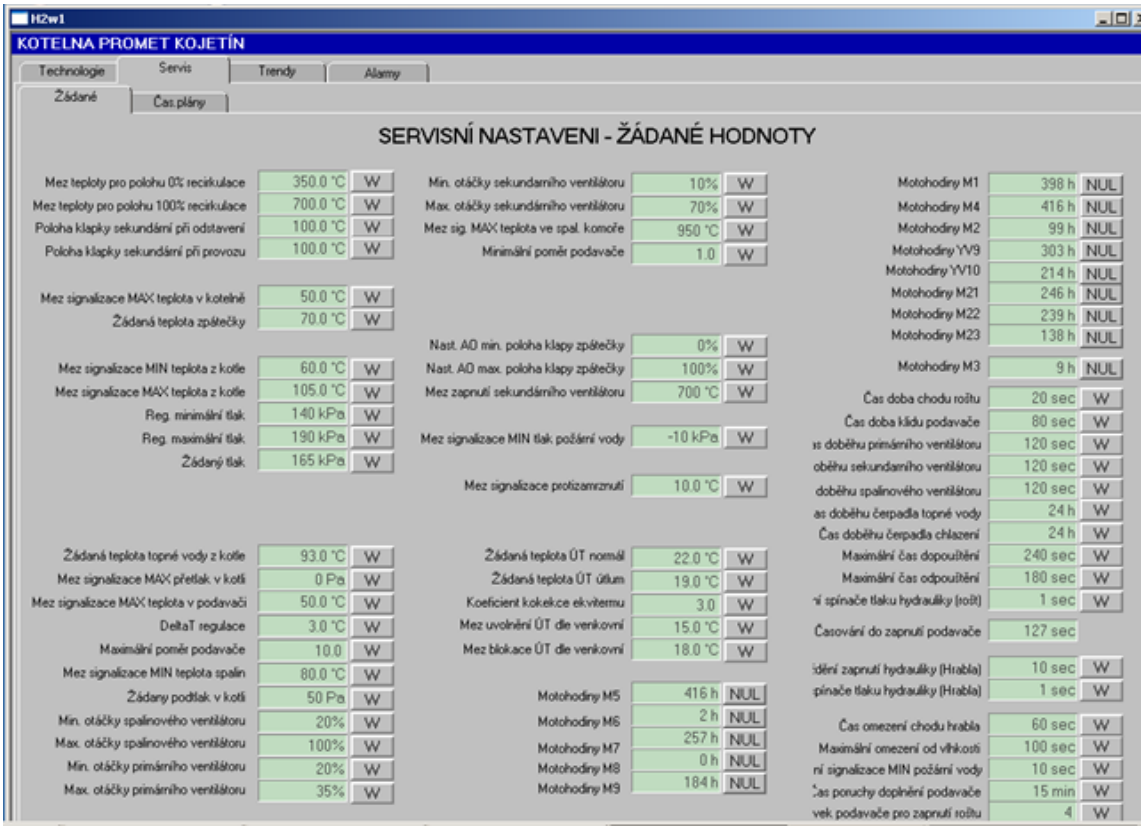
Čas	Směr skokové změny	Velikost
14:02	dolů	30% -> 25%
14:27	nahoru	25% -> 40%

Druhá změna bez efektu – kotel byl v útlumu, žádaná teplota topné vody byla dosažena.

Průtok recyklovaného vzduchu (Tab.9)

Čas	Směr skokové změny	Velikost
11:45	Odstavení	Mez teploty pro polohu 0% recirkulace: 350°C -> 1000°C
12:05	Start	Mez teploty pro polohu 0% recirkulace: 1000°C -> 350°C

Veškeré změny řízených a řídicích veličin se u tohoto kotle provádějí přes GUI (graphic user interface) na operátorské řídicí stanici:



Obr. 25 Operátorská řídicí stanice, Kotelna Promet Kojetín

4.3 Předzpracování provozních dat

Před přistoupením k samotné identifikaci bylo nutno shromáždit, setřídit a sjednotit všechna data, získaná během měření v různých dnech. Pro sledování chemického složení spalin, zejména koncentrace kyslíku, bylo využito zařízení analyzátor plynu – infralyt. Výstupem byl textový soubor s daty snímanými v definované časové periodě (15s). Výhřevnost paliva byla stanovena na základě jednorázového odběru vzorku a následného palivového rozboru v akreditované zkušební laboratoři. Veškeré hodnoty vstupních a výstupních veličin byly exportovány do databázového systému Microsoft Access. Poznámky měření, shrnující provozní podmínky kotle a časy jednotlivých akčních zásahů do chodu soustavy, byly uloženy v souborovém formátu Microsoft Word. Bylo tedy nutné příslušné hodnoty a časy sjednotit do excelových tabulek, některé veličiny následně dopočítat (výkon, průměrnou teplotu ve spalovací komoře,…) a poté podle poznámek měření určit intervaly dat, vybrané pro estimaci (odhad/výpočet) modelu a pro jeho validaci (ověření). Pro tento účel byly vybrány intervaly, zahrnující obě skokové změny (oběma směry) ohraničené zleva i zprava pěti, až deseti minutami, z důvodu komplexnějšího náhledu na daný interval. Součástí předzpracování provozních dat bylo rovněž vytvoření všech příslušných grafických závislostí, které byly v dalších krocích, jak o nich pojednávají následující kapitoly, analyzovány a vyhodnoceny. Předzpracovaná data tvoří přílohu [A].

4.4. Algoritmus 1 (model ARX)

Tento postup byl použit v první fázi identifikace jednotky, kdy jsme předpokládali sestavení přenosů pomocí modelu ARX. Skládá se z následujících kroků:

1. Vytvoření tabulek v Excelu pro jednotlivé přenosy jako zdroj dat pro měření. Tzn. shromáždění příslušných dat do jednoho souboru, který byl dále analyzován a využit pro grafickou analýzu vstupně-výstupních závislostí. Viz příloha [A].

2. Stanovení intervalu pro estimation (výpočetní) data, tzn. takového intervalu, který zahrnoval obě námi vybuzené skokové změny vstupní veličiny.

3. Výběr intervalu pro validation (validační) data. Vybrán byl stejně dlouhý interval příslušné dvojice vstup-výstup se stejným počtem vzorků, zároveň byla ověřena linearita i zde.

4. Grafická analýza pro estimation (výpočetní) interval. Odečtení charakteru a času reakce výstupu na změnu vstupu i samotný charakter příslušných grafických závislostí (vstup = f(čas), výstup = f(čas)).

5. Ověření linearity pomocí grafické závislosti mezi vstupní a výstupní veličinou.

6. Tvorba datových souborů se vstupně-výstupními hodnotami pro jednotlivé přenosy, zahrnující estimation i validation data. Jejich import do *Matlab Workspace*, dále zpracování v *System Identification Toolbox*.

7. Zhodnocení výsledných přenosů, zejména hodnoty funkce *Fit index*.

4.5 Technická omezení identifikace

Výše uvedený postup však nebyl zcela úspěšný, z důvodů, které jsou shrnuty v této kapitole. Jsou jimi zejména:

4.5.1 Nelineární chování

Identifikace pomocí lineárního modelu ARX předpokládala lineární závislost mezi sledovaným vstupem a jeho odezvou na výstupu. Z matematického hlediska je to klíčová vlastnost, z důvodu stanovení parametrů modelu metodou nejmenších čtverců.

Lineární závislost mezi dvěma veličinami je taková, kdy hodnota první veličiny je dána pouze součinem s hodnotou druhé veličiny (*f(x) = 10x)*, nebo součinem s druhou veličinou a přičtením určité konstanty (*f(x) = 10x +5*). V takovém případě se jedná o přímkovou závislost, vyjádřenou lineární funkcí, jejíž hodnota na jejím definičním oboru klesá nebo roste rovnoměrně.

Lineární funkci f(x) vyjádříme ve tvaru:

$$f(x) = k \cdot x + q \tag{37}$$

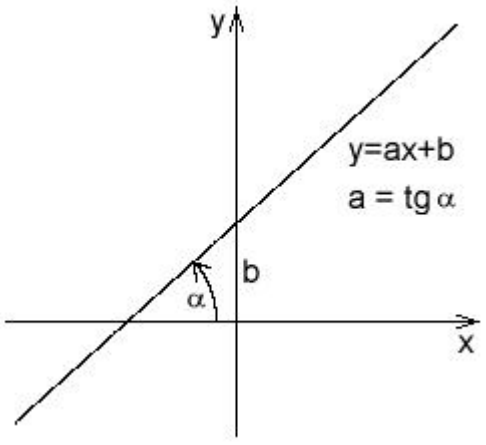
kde *k* i *q* jsou konstanty a k ≠ 0.

Parametr *k* nazýváme směrnice přímky, konstanta *q* definuje její posun v ose x.

Lineární funkce f(x) s *n* proměnnými má tvar:

$$f(x_1, x_2, ..., x_n) = a_1 \cdot x_1 + a_2 \cdot x_2 + ... + a_n \cdot x_n + b \tag{38}$$

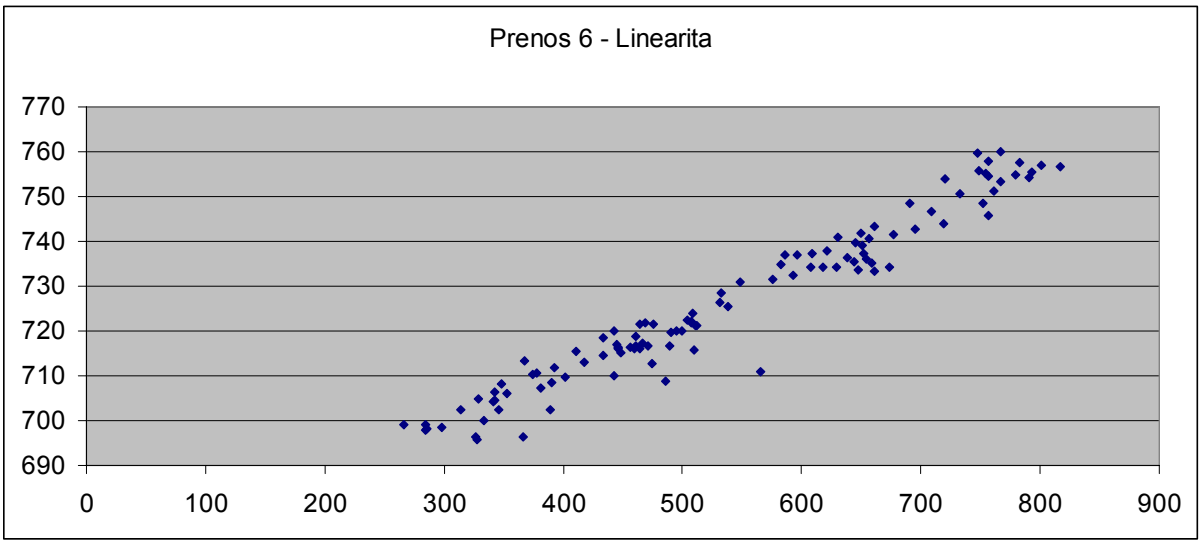
Grafem takové funkce je přímka:



Obr. 26 Graf lineární funkce

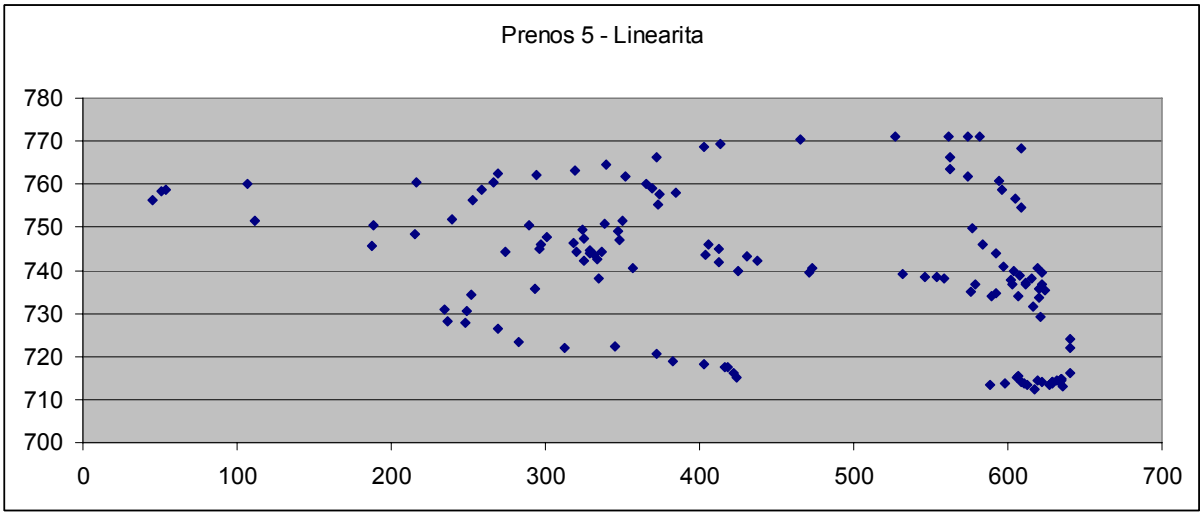
Takový případ by byl ovšem ideální, v praxi málokdy dosažitelný, proto jsme předpokládali alespoň přibližně lineární chování. Bohužel u většiny závislostí se ukázalo, že nejsou ani přibližně lineární, nebo byly lineární jen velice úzké oblasti kolem pracovního bodu systému, a proto byly nepoužitelné pro získání relevantního přenosu.

Jedna z mála lineárních závislostí byla zaznamenána mezi průtokem sekundárního vzduchu a teplotou ve spalovací komoře. Na tomto obrázku je vidět, že získané hodnoty je možné proložit lineární závislostí (přímkou). Nutno však podotknout, že daná závislost platí pouze pro případ plně rozvinutého procesu hoření ve spalovací komoře. Z toho vyplývá omezená použitelnost modelu pouze pro definované podmínky uvedené v kapitole 4.1.



Obr. 27 Přenos 6, ukázka linearita

Naproti tomu závislost teploty ve spalovací komoře na průtoku primárního vzduchu měla následující charakter:

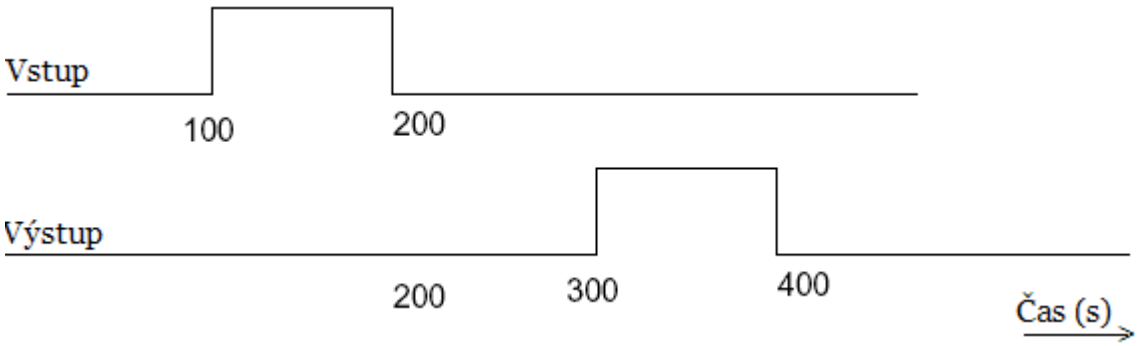


Obr. 28 Přenos 5, ukázka nelinearity

Zde je na první pohled patrné, že tato data není možné proložit ani přibližnou lineární závislostí. V důsledku toho nebyl model ARX schopen vygenerovat žádný relevantní přenos, tj. takový, který by měl dostačující hodnotu FIT funkce. Mimoto bylo předpokládáno i lineární chování ve vstupech - zákon superpozice, což nebylo také dodrženo, viz předchozí graf.

4.5.2 Dopravní zpoždění

Je to časová prodleva mezi změnou na vstupu a odpovídající reakcí na výstupu soustavy, jak ukazuje obr. 29:



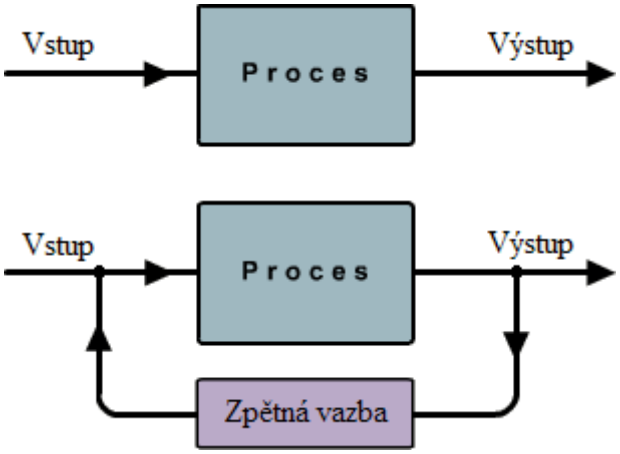
Obr. 29 Dopravní zpoždění

Vysoká dopravní zpoždění (tj. taková, která jsou srovnatelná nebo vyšší než časové konstanty soustavy, tzn. doby ustálení přechodové charakteristiky) zkreslují průběh závislostí a v našem případě, kdy jsme data získali experimentálně, se nedají u některých vstupně-výstupních závislostí ani přesně určit. Například je téměř nemožné (bylo by to pouhé hádání) určit hodnotu dopravního zpoždění mezi okamžikem přísunu paliva a časem, v němž zareagovala teplota ve spalovací komoře. Vlivem zpětné regulace kotle, kterou není technicky možné odpojit, je navíc v grafu výstupní veličiny velký počet skoků, které byly způsobeny jinými faktory a regulačními zásahy a nedá se tedy přesně zjistit, která odezva je „ta pravá.“ Problémem rovněž může být, pokud jsou zásadně různé hodnoty dopravního zpoždění při skokových změnách v obou (různých) směrech.

Pokud chování technologie spalující biomasu srovnáme například s plynovým kotlem, u něhož je možné díky nepoměrně konstantním vlastnostem plynu jako paliva určit dopravní zpoždění mezi přísunem paliva (plynu) a odezvou ve formě zvýšení teploty ve spalovací komoře relativně snadno, u kotle na tuhá paliva, která mají samy o sobě rozmanité vlastnosti, to možné není. Vyskytuje se zde také „zdánlivě“ nelogické chování, kdy po přísunu paliva teplota nejprve klesne (nove palivo rošt na chvíli „udusí“, než se samo rozhoří) a pak teprve vlivem procesu hoření dochází k nárůstu teploty. U plynových kotlů tento problém odpadá.

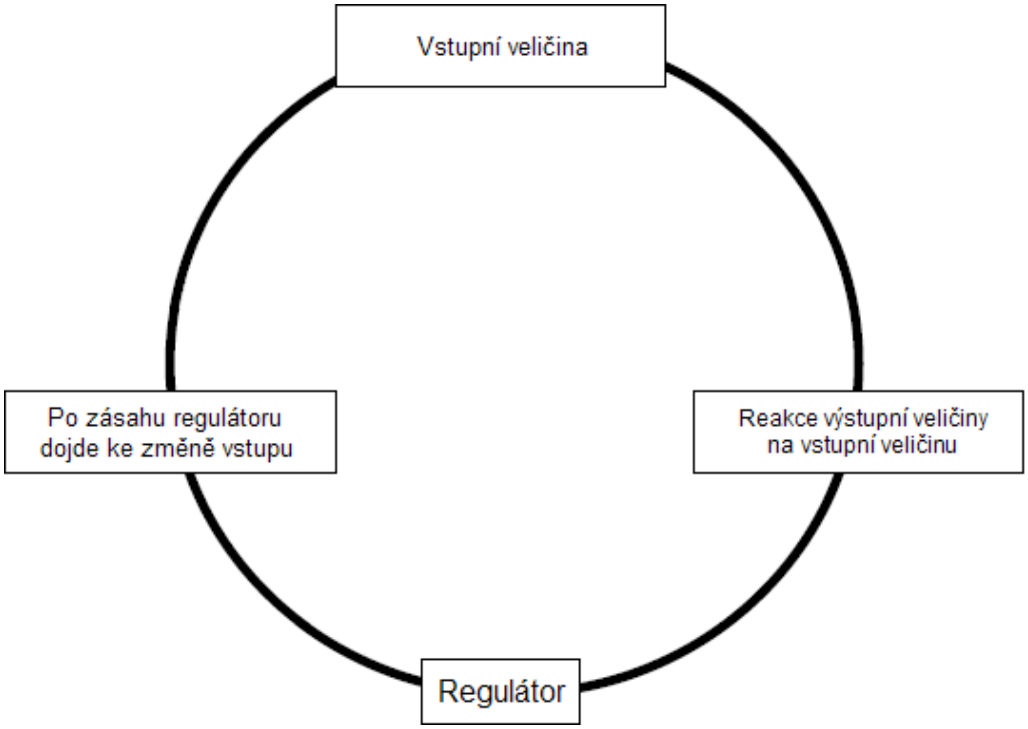
4.5.3. Zásahy regulace

V systémech regulace v uzavřené smyčce, na rozdíl od řízení v otevřené smyčce, je skutečná hodnota regulované veličiny zavedena zpět na vstup regulátoru, jak ukazuje následující schéma:



Obr. 30 Otevřená a uzavřená smyčka

V našem případě může vliv uzavřené regulační smyčky vypadat zjednodušeně například takto:



Obr. 31 Zpětná vazba regulátoru

V praxi má takové chování následující vliv. Teoreticky bychom například očekávali přímo úměrnou závislost mezi průtokem primárního nebo sekundárního vzduchu a koncentrací kyslíku ve spalinách. Tj. že zvýšíme-li průtok primárního vzduchu, tato koncentrace se zvýší. V prvním okamžiku k tomu sice dojde, ale téměř okamžitě zasáhne zpětná vazba regulace, která v reakci na zvýšení primárního vzduchu sníží vzduch sekundární. Ve výsledku se tedy koncentrace kyslíku ve spalinách naopak sníží, tedy vznikne nepřímo úměrná funkční závislost. Takového chování pak samozřejmě generuje nelinearity ve vstupně-výstupní závislosti.

4.5.4 Systém dodávky paliva

Dodávka paliva je klíčovým vstupem celé soustavy, proto se na experimentálně získaných datech negativně projevily i tyto aspekty:

- různorodé složení spalovaného materiálu
- proměnlivá vlhkost
- nestejná výhřevnost použitého paliva
- problém s technologickou jednotkou - podavač paliva (příliš "hrubá" regulace)
- atd.

4.5.5 Závěr

Z těchto příčin se tedy nepodařilo získat dostatečně kvalitní (reprezentativní) přenosy využitím ARX modelu pomocí identifikace v *System Identification Toolbox*. Až na výjimky dávaly jednotlivé přenosy velice slabé hodnoty *Fit index,* v některých případech byly dokonce záporné.

Výše uvedené překážky identifikace by bylo možno odstranit, případně zredukovat, například využitím následujících možností:

1. Odpojení příslušných regulátorů – systém by se však stal nestabilním.

2. Navržení vhodnější metodiky řízení experimentálního měření s těmito cíly:
   - Stanovit přesnější hodnoty dopravního zpoždění
   - Definovat vliv regulátoru na jednotlivé závislosti (moment zásahu), blíže se seznámit s funkcí regulátoru.
   - Specifikovat vliv aktuálního stavu systému (zejména množstvím a polohou paliva na roštu) na jednotlivé vstupně-výstupní závislosti.

3. Určit míru nejistoty v dodávce paliva a zahrnout ji do modelu.

4. Využít pro identifikaci některý z nelineárních modelů.

Shrnutí: Vzhledem k neúspěchu s identifikací pomocí lineárního modelu ARX, byla pro další fáze této diplomové práce využita alternativní varianta, uvedená v kapitole 2.4 ve čtvrtém bodu identifikačního cyklu. Jedná se o odhad parametrů modelu pomocí analýzy přechodové charakteristiky. Je to jedna z nejpoužívanějších metod pro identifikaci neparametrických modelů systému. Umožňuje nám stanovit závislost vstupů/výstupů využitelných pro stanovení předpokládané vnitřní struktury modelu. Tuto volbu a sestavení modelu popisuje podkapitola 4.6 , algoritmus získání příslušných parametrů pro jednotlivé přenosy je popsán v podkapitole 4.7. Takto získané přenosy byly v závěru diplomové práce využity pro dynamické modelování chování jednotky pomocí programu Borland Delphi 7, viz kapitola 5.

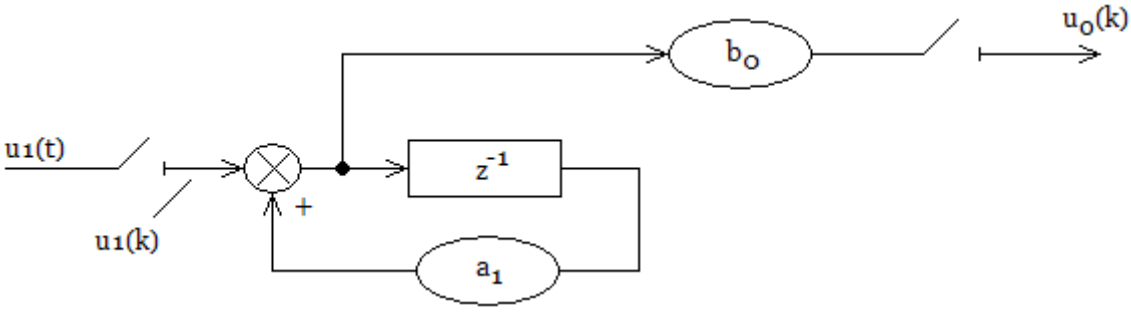
4.6 Sestavení modelu

Pro výběr vhodného typu přenosu a odhad jeho parametrů na základě analýzy přechodové charakteristiky bylo nutno zvolit takovou jeho strukturu, která s dostatečnou přesností popisuje závislost vstupně-výstupních veličin. Předpokladem byl proporcionální charakter závislostí se setrvačností 1. řádu. Byl vybrán následující přenos se setrvačností 1. řádu, splňující výše uvedené požadavky:

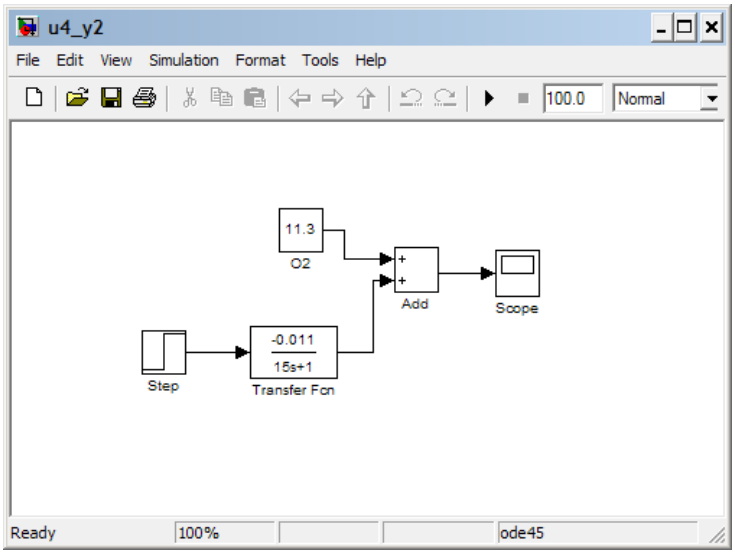
$$G(s) = \frac{K}{ts+1} \cdot e^{-Tds} \tag{39}$$

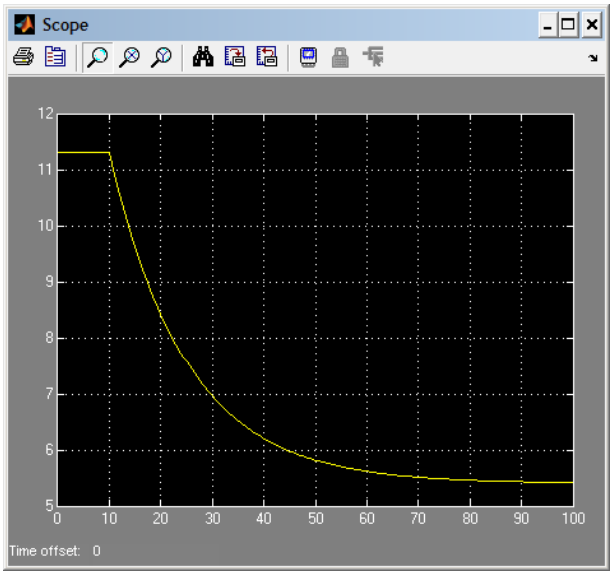
Takto definovaný přenos nám dává jasnou informaci o směru a charakteru funkční závislosti řízené veličiny na změně řídicí veličiny, včetně velikosti dopravního zpoždění. Zvolená struktura přenosu byla využita k sestavení blokového modelu v programu Simulink. Využil jsem k tomu předdefinovaného bloku pro přenosovou funkci (*Transfer function*), kterou jsem doplnil získanými koeficienty *k* (směrnice přímky, kladná nebo záporná) a dopravním zpožděním T_d. Konstantu τ ve jmenovateli přenosové funkce jsem získal experimentálně – měnil jsem ji, dokud se přenos neustálil na konstantní hladině. Do bloku přenosové funkce byla s dopravním zpožděním poslána pomocí bloku *Step* skoková změna vstupní veličiny. Dopravní zpoždění je nastaveno jako parametr tohoto bloku. Dalším použitým blokem je konstanta (*Constant*), která definuje ustálenou hodnotu výstupní veličiny, z níž byl poté realizován skok nahoru, nebo dolů. Tato konstanta spolu s výstupem přenosové funkce je použita jako vstup do součtového bloku (*Add/Sum*) a poté je poslána jako hotový výstup do bloku *Scope*, který se postará o grafické vykreslení. Z technického hlediska v tomto modelu je zpožděn vstup do přenosové funkce, namísto zpoždění výstupu z ní, nicméně prakticky tato skutečnost nehraje žádnou roli. Všechny modely sestavené v Simulinku tvoří přílohu [B].

Následující obrázky ukazují programové a blokové schéma zvoleného modelu, včetně grafického výstupu pomocí bloku *Scope*:



Obr. 32 Programové schéma modelu





Obr. 33 Blokový model a jeho grafický výstup

Následuje postup, kterým byly získány příslušné parametry modelu pro jednotlivé přenosy, včetně celkového přehledu všech získaných přenosů a jejich charakteristik.

4.7 Algoritmus 2 (analýza přechodové charakteristiky)

Pro všechny kroky následujícího postupu byly, stejně jako v případě algoritmu 1, využity předpřipravené soubory v příloze [A].

1. V estimation intervalu vybereme 3 (pokud možno ustálené) hladiny vstupní veličiny a zaznamenáme intervaly hodnot v těchto hladinách pro vstup i výstup.

2. Spočítáme průměry z těchto hodnot jako dvojice $[x_i,y_i]$ a jimi proložíme graf.

3. Provedeme lineární regresní analýzu, tzn. tento graf proložíme přímkou pomocí Excelu (označit graf -> Přidat spojnici trendu -> Lineární).

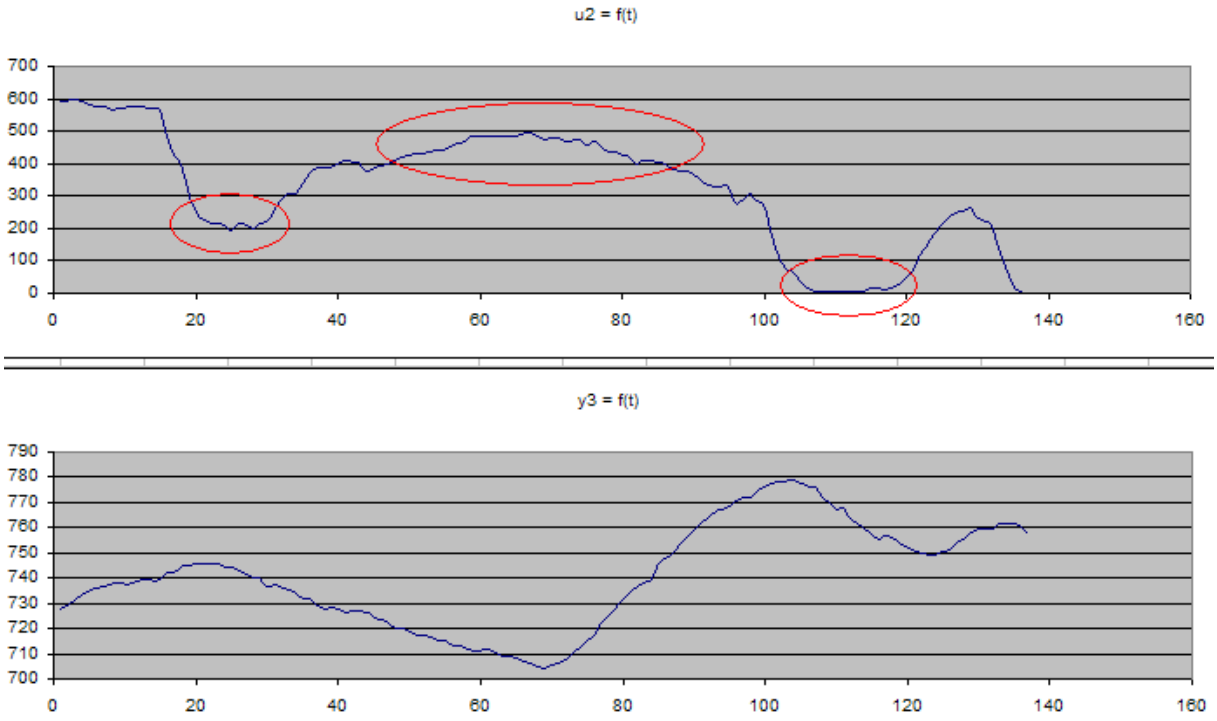
4. Zobrazíme si rovnici regrese (rovnici přímky) a hodnotu spolehlivosti R^2.

5. Pomocí rovnice regrese a hodnot x_i dopočítáme odpovídající y_i.

6. Dopočítáme koeficienty - *diff* pro vstup a výstup jako rozdíl min. a max. hodnoty x_i, y_i; K jako podíl diferencí; T_d (dopravní zpoždění) odečteme z grafu; τ určíme pomocí programu Simulink, nutno měnit parametry a sledovat, kdy se veličina ustálí.

Uveďme si výše uvedený postup na příkladu přenosu u_2-y_3 [Teplota ve spalovací komoře = f (Průtok primárního vzduchu)]:

1. Výběr intervalů



2. Průměry hodnot + graf

u2 (lvl 1)	u2 (lvl 2)	u2 (lvl 3)
598-607	*636-650*	*683-695*
227,11	483,5041	14,80064
216,4076	484,3015	7,943286
213,2429	486,5247	3,906495
209,4269	486,4465	3,777308
195,5628	484,4133	4,032901
214,7383	487,1105	7,366733
209,6974	485,8098	7,274851
201,1767	488,0287	7,1711
214,2795	492,756	6,571196

y3 (lvl 1)	y3 (lvl 2)	y3 (lvl 3)
598-607	*636-650*	*683-695*
745,7419	711,4187	776,315
745,6454	711,355	775,9808
745,6521	711,6179	771,8389
744,1571	710,7078	769,68
744,6441	709,2772	767,3839
743,2583	708,7434	767,5748
741,2795	708,0035	763,8592
740,4175	706,9911	761,6423
739,994	706,0013	759,2419

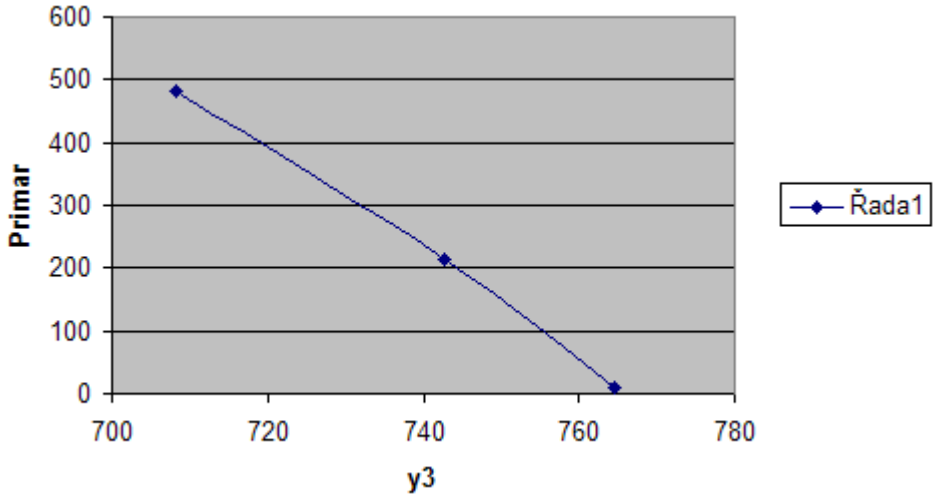
222,0399	485,9074	13,96214
	475,4889	15,02147
	480,7833	11,23793
	479,7387	17,60221
	466,183	
	466,5833	

736,9108	705,1314	757,4551
	704,4431	755,6647
	705,5874	756,6013
	706,5134	755,0977
	707,3522	
	710,4335	

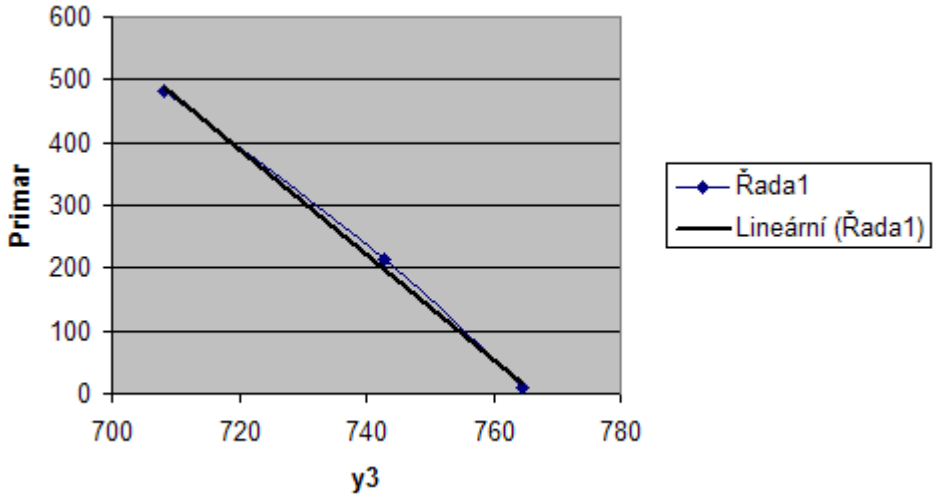
Průměry

212,37	**482,2386**	**9,282174**

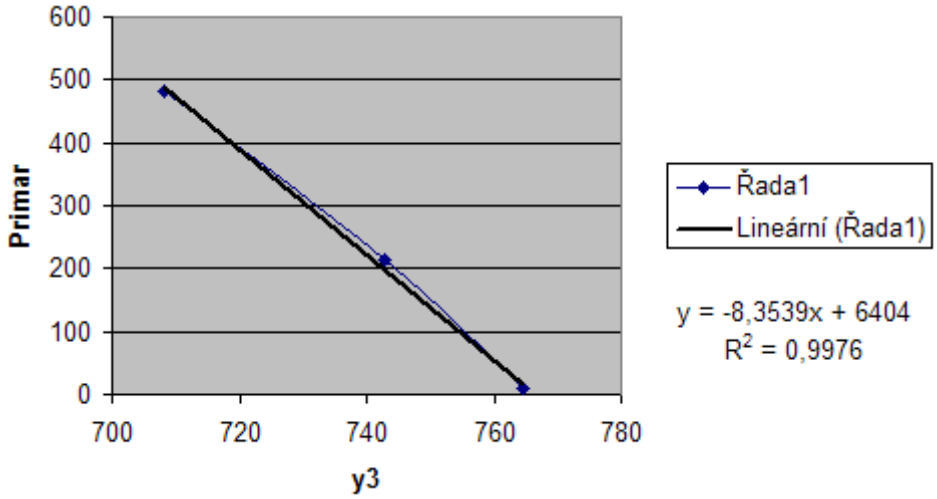
742,77	**708,24**	**764,49**



3. Lineární regrese grafu



4. Přidání rovnice regrese a hodnoty spolehlivosti R^2



5. Výpočet nových hodnot y_i

x1	764,4873	y1	17,54954
x2	708,2384	y2	487,4472

y1 = =(-8,3539*x1)+6404; y2=(-8,3539*x2)+6404

6. Výpočet koeficientů

Diff:	
56,25	469,8977
K =	-0,120
Td =	120
τ =	15

Postup: K = 56,25 / 469,90; T_d odečteno z grafu, τ stanoveno v Simulinku

Přehled všech 12 přenosů a jejich parametrů:

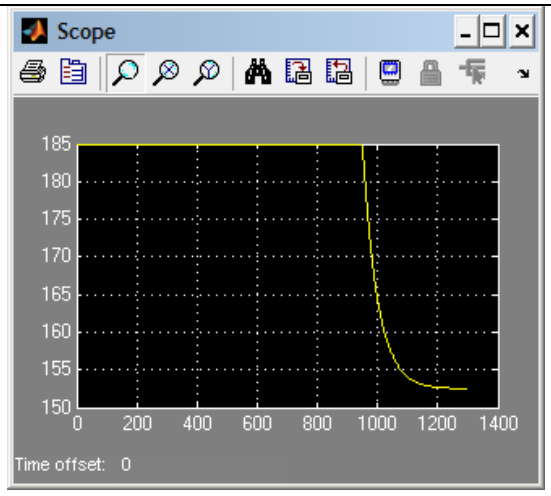
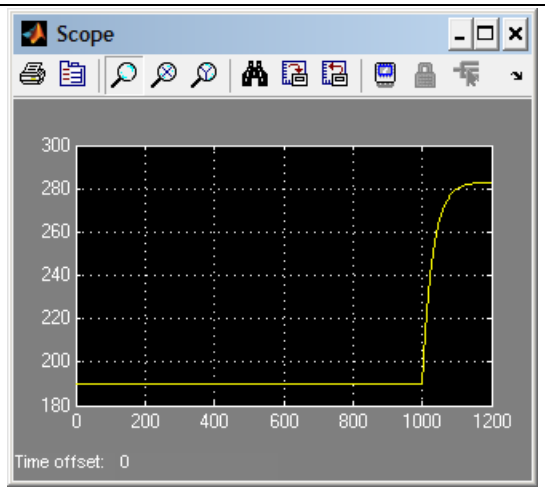
Přenos	**Přenos**	**Faktor spolehlivosti R^2**
u_1-y_1	$G(s)=\frac{1.084}{50s+1}\cdot e^{-950s}$	0,8721
u_2-y_1	$G(s)=\frac{-0.305}{30s+1}\cdot e^{-1000s}$	0,7635
u_3-y_1	$G(s)=\frac{0.188}{30s+1}\cdot e^{-285s}$	0,9935
u_4-y_1	$G(s)=\frac{0.059}{50s+1}\cdot e^{-1100s}$	0,8982
u_1-y_2	$G(s)=\frac{0.004}{20s+1}\cdot e^{-700s}$	1

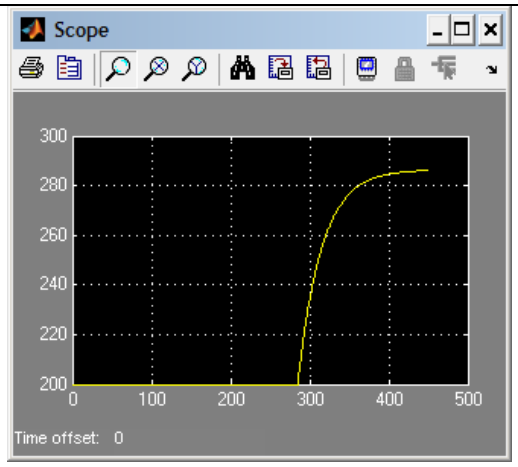
u_2-y_2	$G(s)=\frac{-0.004}{15s+1}\cdot e^{-20s}$	0,7925
u_3-y_2	$G(s)=\frac{-0.009}{9s+1}\cdot e^{-112s}$	0,8685
u_4-y_2	$G(s)=\frac{-0.011}{15s+1}\cdot e^{-50s}$	0,99
u_1-y_3	$G(s)=\frac{-0.541}{20s+1}\cdot e^{-600s}$	0,9365
u_2-y_3	$G(s)=\frac{-0.120}{13s+1}\cdot e^{-120s}$	0,9976
u_3-y_3	$G(s)=\frac{0.067}{7s+1}\cdot e^{-105s}$	0,937
u_4-y_3	$G(s)=\frac{0.145}{12s+1}\cdot e^{-37,5s}$	0,97

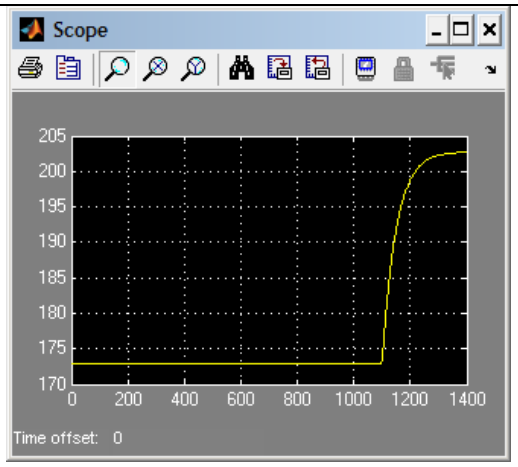
Tab.10 Seznam přenosů

Přehled všech funkčních závislostí mezi jednotlivými vstupy/výstupy s využitím získaných přenosů a dle nich sestavených modelů. D znamená skokovou změnu dolů, N nahoru. Provedli jsme tedy skokové změny oběma směry, abychom získali komplexnější popis chování soustavy. Pro ilustraci je v následujícím přehledu uvedena vždy jen jedna skoková změna vybraným směrem a reakce výstupu na ní.

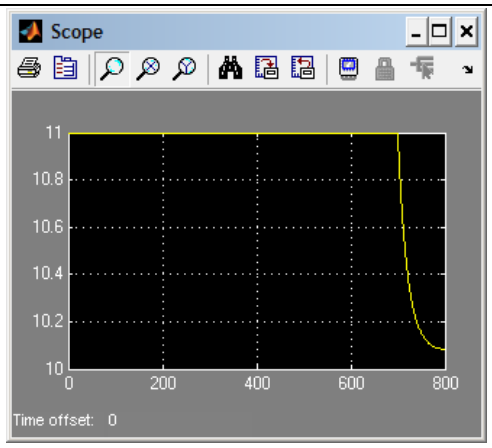
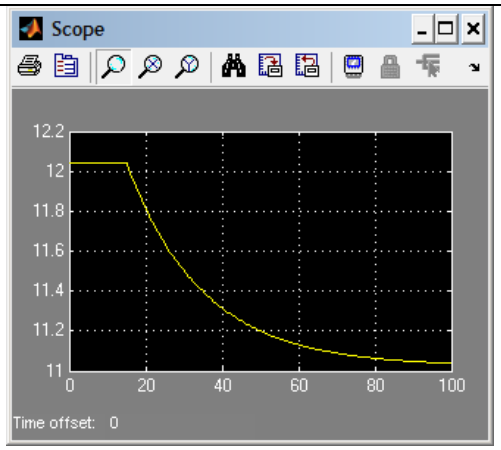
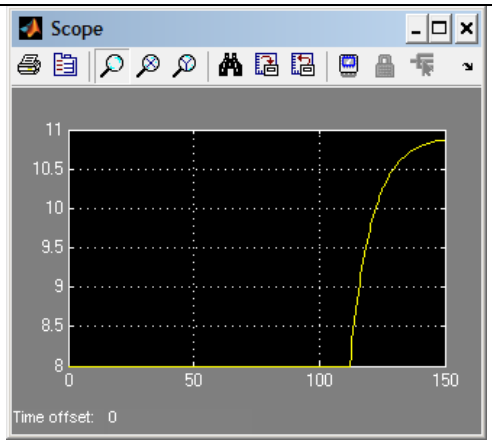
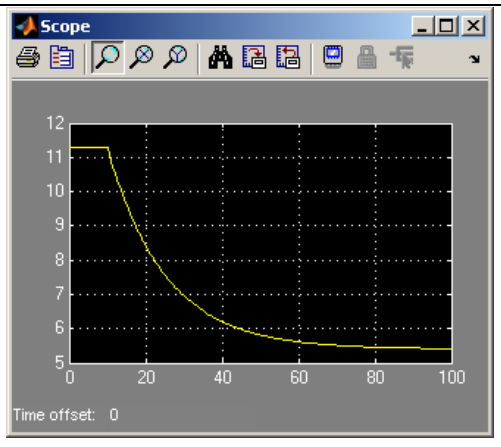
Y_1 – Výkon kotle

u_1 – Výkon podavače (D)	u_2 – Průtok primárního vzduchu (D)
	
u_3 – Průtok sekundárního vzduchu (N)	u_4 – Průtok recyklu (N)

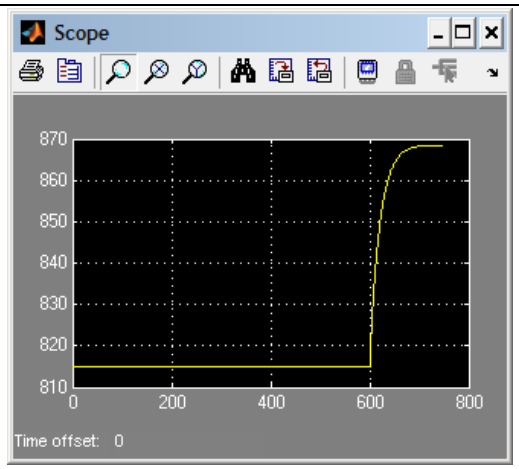
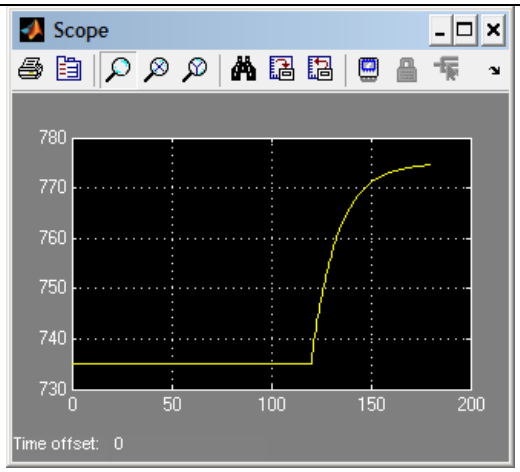
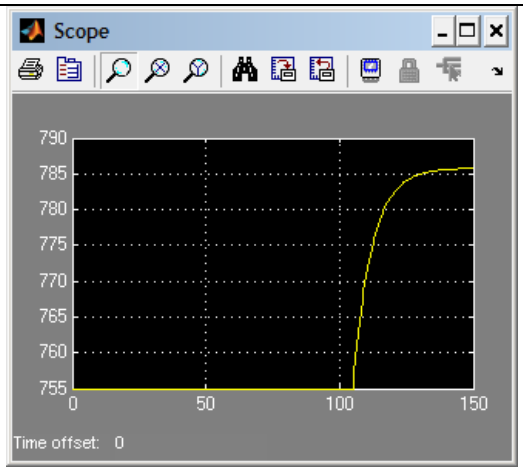
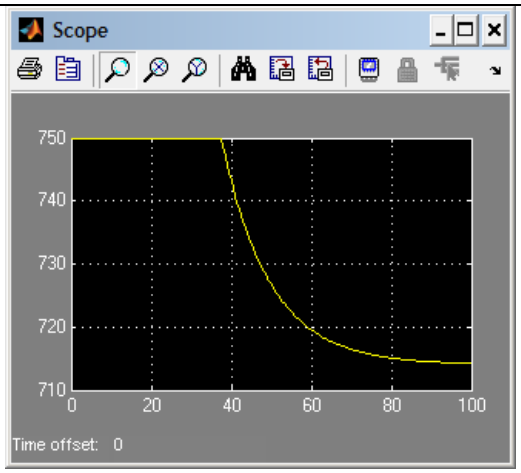




Y_2 – Procento kyslíku ve spalinách

u_1 – Výkon podavače (D)	u_2 – Průtok primárního vzduchu (N)
	
u_3 – Průtok sekundárního vzduchu (D)	u_4 – Průtok recyklu (N)
	

Y_3 – Teplota ve spalovací komoře

u_1 – Výkon podavače (D)	u_2 – Průtok primárního vzduchu (N)
	
u_3 – Průtok sekundárního vzduchu (N)	u_4 – Průtok recyklu (D)
	

Tab.11 Grafický přehled vybraných funkčních závislostí

Poznámka: Faktor spolehlivosti R^2, uvedený v tab.10, je ve statistice nazýván koeficient determinace a slouží ke kontrole, jak dobře proložená závislost vyhovuje všem bodům v grafu. Hodnota koeficientu determinace leží v rozsahu <0,1> a udává, jaký podíl rozptylu v pozorování závislé proměnné se podařilo regresí vysvětlit (větší hodnoty znamenají větší úspěšnost regrese). V našem případě se průměrná hodnota R^2 blíží číslu 0,92 a lze ji považovat za uspokojivou. Pokud by tato hodnota byla nižší než 0,75, nelze ji povazovat za vhodně zvolenou.

Jedním z úkolů této diplomové práce bylo sestavit popis reálné regulované soustavy ve formě přenosové matice. Jak tato matice vypadá v obecném tvaru, jsme si ukázali již v kapitole 2.4.6. Nyní už známe všechny hledané přenosy $G_{ij}(s)$ a můžeme tedy sestavit konkrétní přenosovou matici pro naši soustavu:

$$\mathbf{G}(s) = \begin{bmatrix} G_{11}(s) & G_{12}(s) & G_{13}(s) & G_{14}(s) \\ G_{21}(s) & G_{22}(s) & G_{23}(s) & G_{24}(s) \\ G_{31}(s) & G_{32}(s) & G_{33}(s) & G_{34}(s) \end{bmatrix} =$$

$$= \begin{bmatrix} \frac{1,084}{50s+1}e^{-950s} & -\frac{0,305}{30s+1}e^{-1000s} & \frac{0,188}{30s+1}e^{-285s} & \frac{0,059}{50s+1}e^{-1100s} \\ \frac{0,004}{20s+1}e^{-700s} & -\frac{0,004}{15s+1}e^{-20s} & -\frac{0,009}{9s+1}e^{-112s} & -\frac{0,011}{15s+1}e^{-50s} \\ -\frac{0,541}{20s+1}e^{-600s} & -\frac{0,120}{13s+1}e^{-120s} & \frac{0,067}{7s+1}e^{-105s} & \frac{0,145}{12s+1}e^{-37,5s} \end{bmatrix}$$

4.8 Zhodnocení procesu identifikace

Důležitým kritériem pro hodnoceni získaných přenosů je pro nás faktor spolehlivosti, který přesáhl, nebo se blížil hodnotě 0,9 (90%). Takovou hodnotu je možné považovat za úspěch. Modely byly naimplementovány v programu Simulink a jejich graficky výstup dává dobrý obraz o funkčních závislostech řízených veličin a odpovídá jejich skutečnému chování (jak již bylo uvedeno, za předpokladu splnění určitých počátečních podmínek). Pomocí takto získaných modelů tedy dokážeme vyjádřit, jak se chová řízená veličina v závislosti na změnách řídicí veličiny, můžeme sledovat rychlosti trendu změny, její dopravní zpoždění a také to, kdy se po provedení skokové změny ustálí na konstantní hladině. Z těchto aspektů lze považovat získané přenosy za dostačující ve smyslu zadání této diplomové práce, tj. modelovaní dynamického chováni identifikované soustavy.

5. DYNAMICKÝ MODEL JEDNOTKY

V předchozích kapitolách jsme získali funkční a dostatečně přesné přenosy pro všech 12 závislostí. Použili jsme jednoduchou strukturu modelu a pomocí analýzy frekvenční charakteristiky stanovili příslušné parametry pro jednotlivé přenosy. Posledním krokem této diplomové práce byla tvorba programu, který za prvé ověří správnost těchto přenosů i na jiné platformě než je Simulink a za druhé, pomocí využití jednotlivých přenosů a příslušné diferenční rovnice, umožní modelování chování soustavy v reálném čase, tj. modelovaní změny chování výstupních veličin v závislosti na změnách vstupních veličin. Tento program by tedy měl sloužit jako komplexní vizuální model identifikované jednotky, pomocí něhož je možné testovat chování jednotky i bez zásahů do reálného provozu. K implementaci programu bylo využito prostředí Borland Delphi 7. Program lze spustit na běžném PC s OS MS Windows, min. 500MHz CPU a 512MB RAM. Nevyžaduje žádné dodatečné knihovny ani HW, nebo SW. Program tvoří přílohu [D].

5.1 Softwarová podpora (Borland Delphi 7)

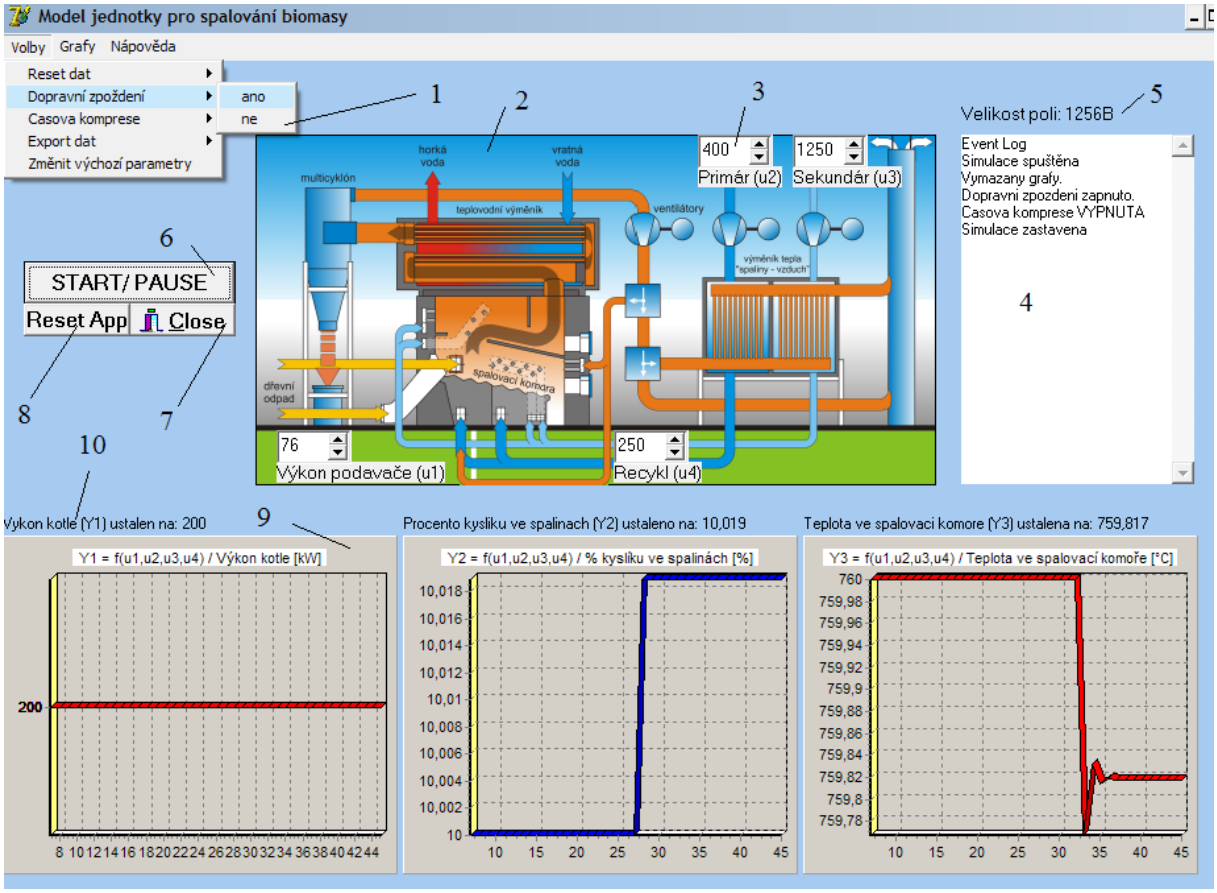
Borland Delphi je integrované grafické vývojové prostředí firmy Borland určené pro tvorbu aplikací na platformě MS Windows v jazyce Object Pascal (objektové nástavbě Pascal). Programování v něm je založeno na použití komponent (to je malý balíček), které vykonávají určitou funkci v programu. Velkou předností Delphi proti některým konkurenčním produktům jsou knihovny komponent, které jsou jeho nedílnou součástí. Další potřebné komponenty lze snadno stáhnou z internetu a využít tak program naplno. [17]

Charakteristické znaky Delphi [13]:

- Rychlý a výkonný, plně 32bitový optimalizační kompilátor a linker.
- Použití jazyka *Object Pascal*
- Výborná databázová podpora
- Možnost vytváření aplikací Klient/Server.
- Vizuální dědičnost formulářů, což obnáší maximální spojení objektově orientovaného programování a vizuálního návrhu aplikací.
- Volně přístupný zdrojový kód knihovny vizuálních komponent.
- Velmi dobrá podpora ovládacích prvků OLE controls.
- Velké množství volně přístupných komponent.
- Možnost vytváření aplikací pro internet.
- S využitím knihovny CLX je možné vytvářet i aplikace pro více platforem.
- Podpora technologií COM a ActiveX.

5.2 Přehled GUI a možností programu

Uživatelské grafické rozhraní ukazuje obrázek 34:

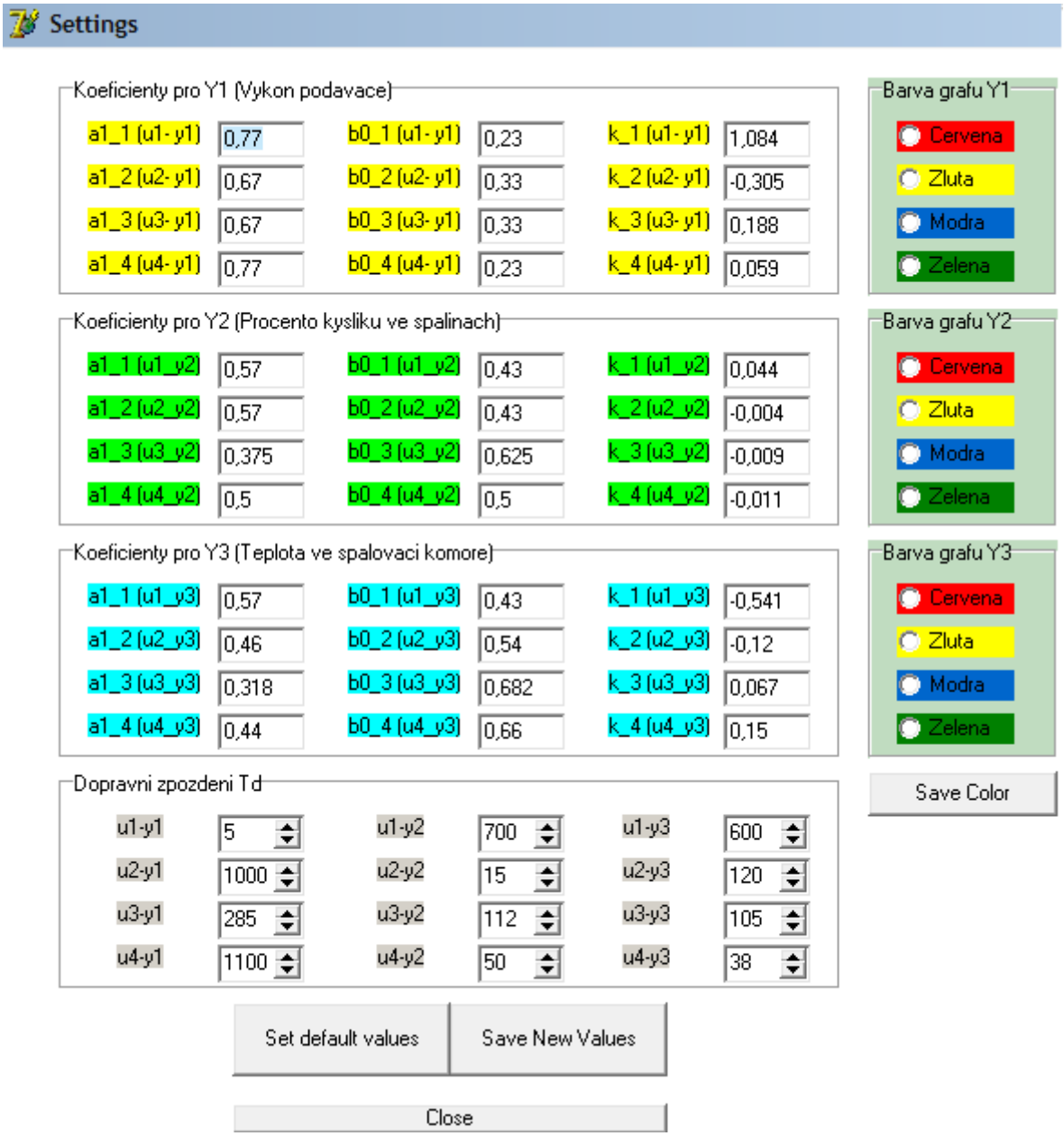


Obr. 34 GUI programu Model jednotky pro spalování biomasy

Popišme si jednotlivé části GUI, které jsou očíslovány na výše uvedeném obrázku.

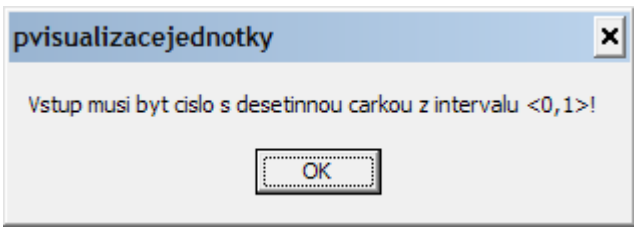
1 – Hlavní menu programu. Obsahuje: *Volby*, *Grafy*, *Nápověda*. V menu *Volby* je možné zvolit *Reset dat -> Grafy* nebo *Reset -> Memo*. První možnost vymaže veškerý obsah všech tří malých grafů, druhá možnost vyčistí Event log. *Dopravní zpoždění -> Ano/Ne* zapne, nebo vypne vliv dopravního zpoždění definovaného v programu. Pokud je vyplé, všechny změny probíhají v reálném čase. *Časová komprese* umožňuje zvolit 4 možnosti: *10x*, *30x*, *50x*, *100x* nebo *1:1*, tedy reálný čas. Možnost časové komprese najde využití v případě zaplého dopravního zpoždění, které je však příliš vysoké a uživatel nechce čekat celou dobu. Aktivní časová komprese změní rychlost vykreslování grafů v příslušném poměru. *Export dat* zajistí exportování celé historie průběhu (hodnot) všech vstupních a výstupních veličin do příslušných *txt* souborů na pevný disk uživatele. Tato data je poté možno např. naimportovat do programu MS Excel a následně s nimi dále pracovat.

Volba *Změnit výchozí parametry* otevře následující dialog *Settings*:



Obr.35 Settings

Zde je možné měnit veškeré parametry. Koeficienty přenosu a_1, b_0 a k pro všechny přenosy. Rovněž hodnoty dopravních zpoždění T_d a barvy všech tří grafů. Validita vstupních hodnot je kontrolována blokem try – except. Pokud uživatel zadá omylem do vstupního políčka např. písmeno nebo jiný znak, než číslo, objeví se varovná hláška:



Obr.36 Validace zadaných hodnot

Jako poslední je zde možnost vrátit zpět výchozí hodnoty (Set Default Values). Tyto hodnoty jsou napevno zapsány v kódu a vycházejí z našich meření.

Volba *Grafy* umožňuje zobrazení (otevření) kteréhokoliv grafu v novém, větším okně. V takovém případě se graf začne vykreslovat až po otevření, není tedy zatížen historií průběhu.

Volba *Nápověda* zobrazí informační okno o autorovi programu.

2 – Model soustavy. Tento obrázek slouží pro dokreslení lepší představy o identifikované jednotce. Jednotlivé vstupy (3) jsou umístěny na místech, kde se v reálu opravdu nacházejí.

3 – Vstupy řídicích veličin. Implementovány pomocí komponenty *TSpinEdit*. Umožnují zadávání hodnot vstupních veličin, buď přepsáním čísla, nebo změnou o definovanou konstantu po kliknutí na šipku nahoru/dolů.

4 – Event log. Implementován pomocí třídy *TMemo*. Slouží k zobrazování informací typu: start/pozastavení simulace, vymazání grafů nebo mema, provedení exportu dat do souborů, zapnutí/vypnutí dopravního zpoždění, změna a aktuální hodnota časové komprese, změna provedená v dialogu *Settings*, dosažení datového limitu (5), restartování aplikace (8).

5 – Velikost polí v B (bytech). Hodnoty všech vstupů i výstupů jsou v pravidelných intervalech ukládány do dynamických polí. To znamená, že s časem lineárně roste jejich datová velikost, která je v každém kroku spočtena příslušnou funkcí. V případě, že paměťová velikost těchto dynamických polí dosáhne stanovené hraniční hodnoty, program se pozastaví a v Event logu informuje uživatele o faktu dosažení max. paměťového limitu.

6 – `Start/Pause` button. Umožňuje spuštění nebo pozastavení aplikace.

7 – `Close` button. Zavře aplikaci.

8 – `Reset app` button. Restartuje aplikaci, tzn. vymaže aktuální obsah všech dynamických polí, vypne dopravní zpoždění a časovou kompresi, vymaže event log a nastaví preddefinované hodnoty vstupů a výstupů na výchozí úroveň.

9 – Grafy průběhu jednotlivých výstupních veličin. Implementovány pomocí třídy *TChart*. Pomocí příslušných funkcí vykreslují hodnoty výstupních veličin v daném časovém intervalu.

10 – Informace o ustálení. Jakmile se výstupní veličina po skokové změně ustálí na konstantní hodnotě, zobrazí se informace a nová hodnota výstupní veličiny.

5.3 Přehled implementovaných tříd, procedur a funkcí

Základní třída je *ToutputYi*:

```
TOutputYi = class(TObject)
     private
       List_a1 : array [1..4] of double;
       List_b0 : array[1..4] of double;
       List_k : array[1..4] of double;
       List_diff: array [1..5] of double;
        Input_u1: smallint;
        Input_u2: smallint;
        Input_u3: smallint;
        Input_u4: smallint;
        Y_value:double;
     public
         constructor Create;
         destructor Destroy; override;
          procedure FillValues_a1(a1_1,a1_2,a1_3,a1_4:double);
          procedure FillValues_b0(b0_1,b0_2,b0_3,b0_4:double);
          procedure FillValues_k(k_1,k_2,k_3,k_4:double);
          procedure Count_Differences(In1,In2,In3,In4:smallint);
end;
```

Z této třídy jsou poté vytvořeny 3 instance, pro každý výstup jedna. Každá instance si drží svoje hodnoty koeficientů a_1,b_0,k ve statických polích. Např. instance pro Y_1 si drží čtveřici koeficientů – od každého vstupu, který ji ovlivňuje, jeden. Statické pole *List_diff* je využíváno procedurou *Count_Differences* k ukládání diferencí v krocích *k – (k-1)* pro jednotlivé hodnoty vstupních veličin u_1-u_4. Do proměnných *Input_*u_i se ukládá nová hodnota vstupní veličiny. Tyto proměnné jsou dále předány jako parametry do procedury *Count_Differences* pro výpočet rozdílů hodnot mezi kroky *k – (k-1)*. Procedury *FillValues* zajišťují naplnění příslušných polí hodnotami parametrů.

Druhou použitou třídou je *THistory*:

```
 THistory = class(TObject)
  private
   y1_history : array of double;    // historie vystupu
   y2_history : array of double;
   y3_history : array of double;
    u1_history : array of smallint;  // historie vstupu
    u2_history : array of smallint;
    u3_history : array of smallint;
    u4_history : array of smallint;
  public
   constructor Create;
   destructor Destroy;override;
    procedure SaveHistory;   // uklada prubehy hodnot
    function CheckHistorySize:integer; // kontroluje pametove naroky
  end;
```

Tato třída si drží dynamická pole pro všechny vstupní a výstupní hodnoty. Procedura *SaveHistory* do nich průběžně ukládá příslušné hodnoty. Z těchto polí se případně vybírají hodnoty I/O pro export. Funkce *CheckHistorySize* průběžně počítá jejich paměťovou velikost. Kromě toho jsou definovány ještě dvě statická pole

DelayHelper a *DelayChange*, která jsou využita v případě simulace s dopravním zpožděním.

V programu jsou použity mimo jiné následující procedury a funkce:

1. `procedure ExportData(SetArrayNameD:array of double; SetArrayNameS: array of smallint; SetFileName:string; double:boolean);`
   - zajišťuje export dat z dynamických polí uvnitř instance THistory

2. `procedure TOutputYi.FillValues_a1(a1_1,a1_2,a1_3,a1_4:double);`
   - naplní koeficienty a_1

3. `procedure TOutputYi.FillValues_b0(b0_1,b0_2,b0_3,b0_4:double);`
   - naplní koeficienty b_0

4. `procedure TOutputYi.FillValues_k(k_1,k_2,k_3,k_4:double);`
   - naplní koeficienty *k*

5. `procedure THistory.SaveHistory;`
   - uloží historii hodnot všech vstupních veličin

6. `function THistory.CheckHistorySize:integer;`
   - spočítá paměťovou velikost všech dynamických polí *THistory* v bytech

7. `procedure TOutputYi.Count_Differences(In1,In2,In3,In4:smallint);`
   - spočítá diference $u_i(k) - u_i(k-1)$

8. Procedura pro aktualizaci pole parametrů podle nastavení v *Settings*.

9. Procedura nastavení výchozích hodnot *Settings*.

A další, zajišťující vlastní chod a logiku aplikace.

5.4 Výpočetní logika – algoritmus zpracování

Vlastní výpočet je primárně zajišťován uvnitř timeru, který spouští a obsahuje vlastní výpočetní cyklus.

Při spuštění programu se provede nejprve následující kód:

```
procedure TForm1.FormCreate(Sender: TObject);
 var x:integer;
  begin
   step:=0; i:=0; j:=0; k:=0; // inicializace na nulu
   DelayEnabled:=false;      // vypnuti dopravniho zpozdeni
   RunY1RealTime:=true;
    RunY2RealTime:=true;    // vykreslovani grafu v realnem case
     RunY3RealTime:=true;

   for x:=1 to 12 do           // inicializace statickych poli
    begin
```

```
    iDelayHelper[x]:=0;
    iDelayChange[x]:=false;
  end;

// Vytvoreni instance Y1 tridy TOutputYi a naplneni def. hodnotami
 Y1:=TOutputYi.Create;      // instance pro Y1 - Vykon podavace
  Y1.FillValues_a1(0.77,0.67,0.67,0.77);   //////
   Y1.FillValues_b0(0.23,0.33,0.33,0.23);  // Naplneni poli konstantami
    Y1.FillValues_k(1.084,-0.305,0.188,0.059);  //////
     Y1.Y_value:=400;     // Pocatecni hodnota     400kW

// Vytvoreni instance Y2 tridy TOutputYi a naplneni def. hodnotami
 Y2:=TOutputYi.Create;    // instance pro Y2 – O2 ve spalinach
  Y2.FillValues_a1(0.57,0.57,0.375,0.5);
   Y2.FillValues_b0(0.43,0.43,0.625,0.5);
    Y2.FillValues_k(0.044,-0.004,-0.009,-0.011);
     Y2.Y_value:=10;  // v procentech

// Vytvoreni instance Y1 tridy TOutputYi a naplneni def. hodnotami
 Y3:=TOutputYi.Create;    // instance pro Y3 - T ve spalovaci komore
  Y3.FillValues_a1(0.57,0.46,0.318,0.44);
   Y3.FillValues_b0(0.43,0.54,0.682,0.66);
    Y3.FillValues_k(-0.541,-0.12,0.067,0.15);
     Y3.Y_value:=755; // Teplota ve spalovaci komore

(* Vytvoreni instance THistory, constructor tridy zaroven vytvori
prislusna dynamicka pole *)

History:=THistory.Create;

(* Naplneni poli vystupu na pocatecni hodnoty. Plni se 2x, aby byla v
prvnim kroku diference k-(k-1) nulova a program pracoval na konstantni
hladine, dokud uzivatel nezmeni vstupni velicinu *)

 History.y1_history[High(History.y1_history)]:=Y1.Y_value;
  SetLength(History.y1_history,length(History.y1_history)+1);
   History.y1_history[High(History.y1_history)]:=Y1.Y_value;

 History.y2_history[High(History.y2_history)]:=Y2.Y_value;
  SetLength(History.y2_history,length(History.y2_history)+1);
   History.y2_history[High(History.y2_history)]:=Y2.Y_value;

 History.y3_history[High(History.y3_history)]:=Y3.Y_value;
  SetLength(History.y3_history,length(History.y3_history)+1);
   History.y3_history[High(History.y3_history)]:=Y3.Y_value;
end;
```

V této fázi program běží na konstantní hladině. Protože se nemění hodnoty vstupních veličin, hodnoty *k – (k-1)* jsou nulové a je nulová i hodnota diferenční funkce. V cyklu *Timer1* je následující algoritmus.

V každém kroku se nejprve provede test paměťové velikosti dynamických polí. Pokud vyhovuje, testují se podmínky na dopravní zpoždění. Tato podmínka vypadá například takto:

```
     if (DelayEnabled=true) then     // je zaple dopravni zpozdeni
      if (iDelayChange[1]=true) then   // u1 meni y1
        begin
         (* pozastavi se vypocet diferencni rovnice. Zmena zatím neprobehla,
takze program az do dosazeni dopravniho zpozdeni pracuje na konstatni
hladine *)
          RunY1RealTime:=false;
           i:=i+1;
       // pri dosazeni Td se spusti vypocet
            if (i=Form2.SpinEdit37.value) then RunY1RealTime:=true;
       // pri dosazeni Td+1 se vypne cyklus, anuluji se prislusne promenne
            if (i=Form2.SpinEdit37.value+1) then
             begin
              iDelayHelper[1]:=0; i:=0; iDelayChange[1]:=false;
RunY1RealTime:=true;
             end
        end;
```

Následuje kód pro výpočet vlastní diferenční rovnice, kterému předchází výpočet hodnot diferencí v rovnici použitých. Po výpočtu diferenční rovnice se její výsledek připočítá k předchozí hodnotě a vykreslí se do grafu. Zároveň se uloží do příslušného pole historie výstupu.

```
if (RunY1RealTime = true) then begin // pokud ma pocitat

(* vypocita hodnoty diferenci vstupu. Jsou ulozeny ve statickych polich
uvnitr jednotlivych instanci*)

Y1.Count_Differences(Y1.Input_u1,Y1.Input_u2,Y1.Input_u3,Y1.Input_u4);

// diference hodnoty vystupu k – (k-1)
Y1.List_diff[5]:=History.y1_history[high(History.y1_history)]-
History.y1_history[high(History.y1_history)-1];

// vypocet vlastni diferencni rovnice
dif_Count_Y1:=(((Y1.List_a1[1]*Y1.List_diff[5]) +
(Y1.List_b0[1]*(Y1.List_diff[1]+ iDelayHelper[1])))*Y1.List_k[1])+
(((Y1.List_a1[2]*Y1.List_diff[5]) + (Y1.List_b0[2]*(Y1.List_diff[2]+
iDelayHelper[4])))*Y1.List_k[2]) +(((Y1.List_a1[3]*Y1.List_diff[5]) +
(Y1.List_b0[3]*(Y1.List_diff[3]+iDelayHelper[7])))*Y1.List_k[3]) +
(((Y1.List_a1[4]*Y1.List_diff[5]) +
(Y1.List_b0[4]*(Y1.List_diff[4]+iDelayHelper[10])))*Y1.List_k[4]);
end else dif_Count_Y1:=0;

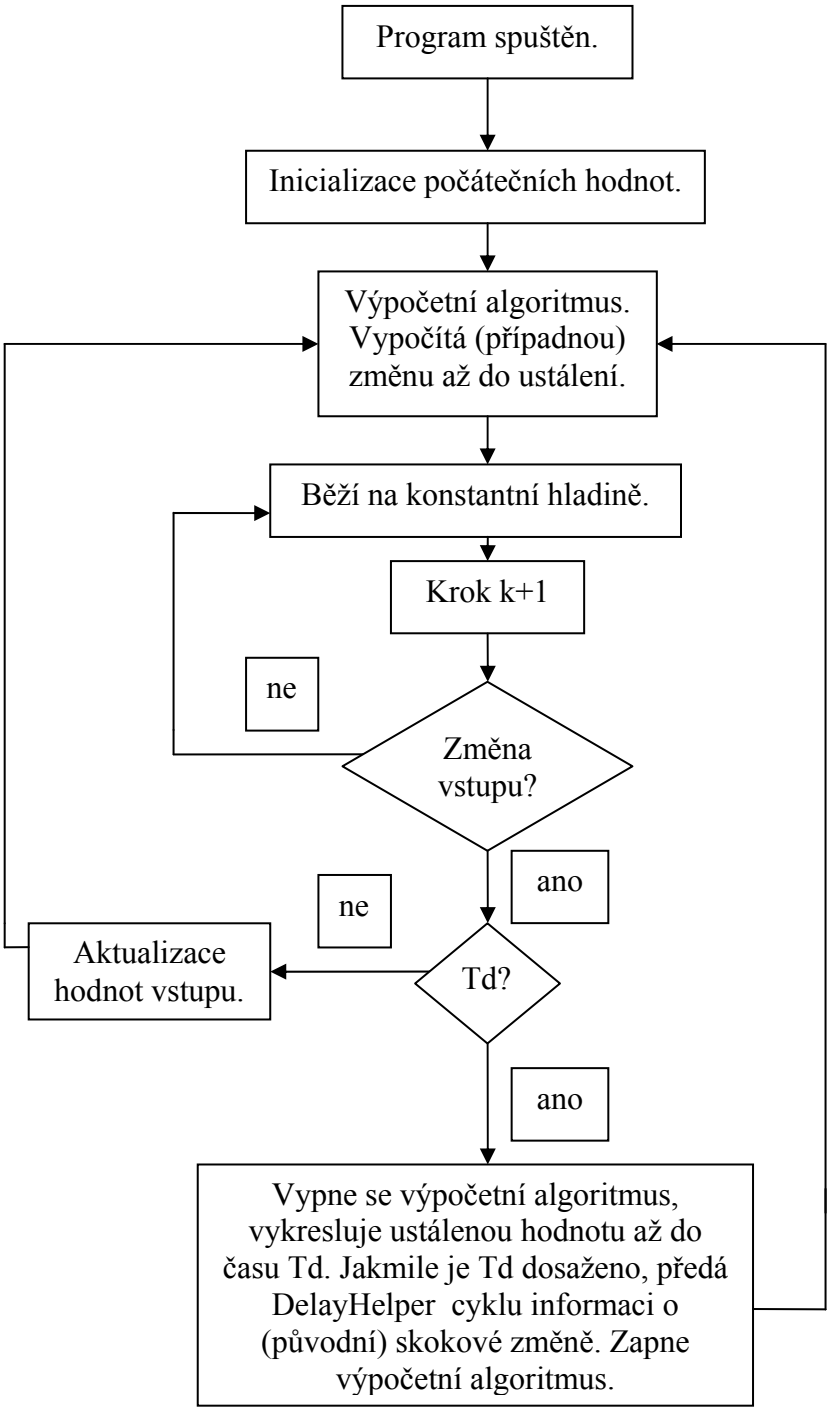
// vypocet nove hodnoty Y1 + test ustaleni
 New_Y1:=History.y1_history[high(history.y1_history)] + dif_Count_Y1;
   if (abs(New_Y1 - History.y1_history[high(history.y1_history)]) < 0.01)
     then
      begin
        New_Y1:= History.y1_history[high(history.y1_history)];
        Y1_round:=RoundTo(New_Y1,-3);
        Label5.Caption:=('Vykon kotle (Y1) ustalen na: ' +
                         FloatToStr(Y1_round));
      end;
// Vykresleni do grafu
    Series1.AddXY(step,New_Y1);
 // Ulozeni do prislusneho dynamickeho pole
     SetLength(History.y1_history,length(History.y1_history)+1);
     History.y1_history[high(History.y1_history)]:=New_Y1;
```

Pokud uživatel změní některou ze vstupních veličin, vykoná se následující kód (příklad změny u_1 – Výkon podavače), který změní aktuální hodnotu vstupu uvnitř atributu všech 3 instancí *TOutpuYi*. Pokud je zaplé dopravní zpoždění, nastaví se *DelayHelper*, který uchová změnu vstupní veličiny až do okamžiku, kdy se naplní dopravní zpoždění. Pak ji pošle do cyklu, diferenční rovnice spočítá novou hodnotu výstupu v reakci na zpožděný vstup a *DelayHelper* se anuluje. *DelayChange* slouží jako logická podmínka.

```
       procedure TForm1.SpinEdit4Change(Sender: TObject);
        var i:integer;
         begin
          if (DelayEnabled=true) then
           begin
            for i:=1 to 3 do
             begin
              iDelayHelper[i]:=SpinEdit4.value -
History.u1_history[high(History.u1_history)];
              iDelayChange[i]:=true;
             end;
           end;

       Form1.SpinEdit4.Color:=clYellow;
        Timer1.Enabled:=false;
         Y1.Input_u1:= Form1.SpinEdit4.value;
         Y2.Input_u1:= Form1.SpinEdit4.value;
         Y3.Input_u1:= Form1.SpinEdit4.value;
          Timer1.Enabled:=true;
         end;
```

Shrňme si celkovou logiku programu následujícím jednoduchým schématem:

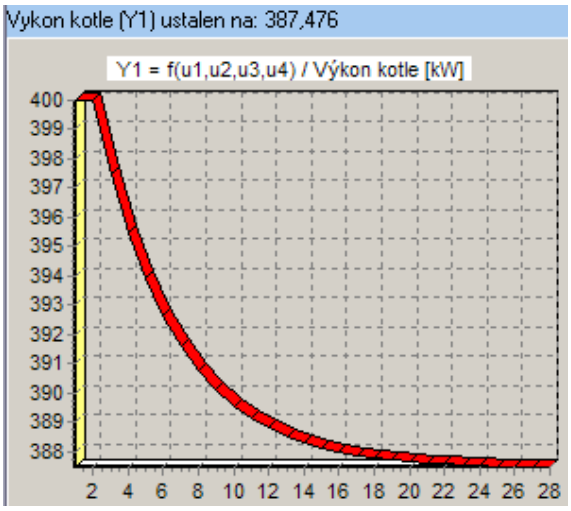


Obr. 37 Schéma logiky programu

5.5 Výsledky modelování

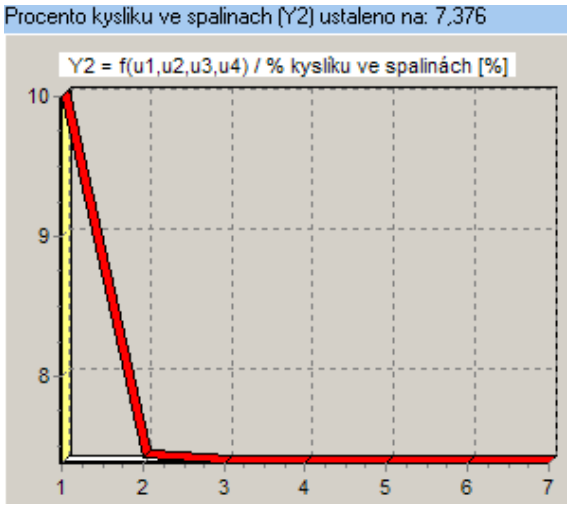
Program dává pro změny vstupních veličin stejný grafický výstup, jako model sestavený v Simulinku. Charaktery změn a jejich trendy si odpovídají. Z těchto aspektů lze považovat výsledky modelování za správné. Uveďme si některé ukázky chování programu i v porovnání s experimentálně naměřenými daty.

Obr.38 ilustruje změnu výkonu kotle v závislosti na skokové změně výkonu podavače. Dopravní zpoždění bylo pro lepší názornost vypnuto. Byla provedena skoková změna výkonu podavače směrem dolů, z 75% -> 65%. Program dal tento výstup, na kterém vidíme snížení výkonu kotle o 12,524 kW.



Obr. 38 Příklad simulace Y1

Na dalším příkladu si ukážeme srovnání výsledků programu s experimentálně naměřenými daty. Provedeme skokovou změnu sekundárního vzduchu o 460 Nm³/h směrem nahoru. Vidíme, že koncentrace kyslíku ve spalinách poté klesla cca o 2,63%. V grafu z experimentálně naměřených dat, viz příloha [A], můžeme vidět, že na reálnou skokovou změnu sekundárního vzduchu o stejnou hodnotu 460 Nm³/h směrem nahoru, reagovala identifikovaná soustava poklesem koncentrace kyslíku přibližně z 12% na 9,11%, tedy rozdílem 2,89 procent. To je oproti našemu modelu rozdíl 0,26 procent. Je dán zřejmě tím, že aktuální model nepopisuje veškerou dynamiku soustavy, včetně případných zásahů regulace.



Obr. 39 Příklad simulace Y2

6. ZÁVĚR

Cílem diplomové práce bylo najít vhodný model, popisující chování identifikované soustavy a popsat ji pomocí matice přenosů. Tento cíl byl splněn, jak dokládá kapitola 4.7.

Teorie identifikace systémů je velmi rozsáhlý celek. Nebylo snadné sestavit stručný a výstižný vstup do problematiky v úvodních kapitolách. Teoretické informace v kapitole 2 však slouží především jako základ pro praktickou část a poskytují dostatečné množství zdrojů pro případné dohledání bližších informací.

Praktická část ukazuje, jak komplexní úlohou může identifikace systémů být. Její složitost je samozřejmě závislá na identifikovaném systému. V tomto smyslu byla řešena poměrně náročná technologie jednotky. Relativně vysoký počet vstupně-výstupních závislostí a její specifické vlastnosti, vyvolaly zvýšené nároky na řešení v každé fázi identifikační procedury. Nakonec se však povedlo tuto proceduru úspěšně dokončit. Přesnost matematického modelu byla ověřena naprogramovanou simulací v prostředí Borland Delphi. Zde bylo vytvořeno uživatelské rozhraní s možností ovlivňovat vstupní veličiny a sledovat změny, které tyto zásahy vyvolávají na výstupech.

Mezi přínosy této diplomové práce je možné zařadit vytvoření softwarového modelu jednotky, který může najít uplatnění například při výuce technických oborů. V budoucnu ho bude možné použít pro dokonalejší nastavení parametrů jednotlivých regulačních smyček (s využitím regulace v Simulinku).

Práce ukázala, že je možné využít identifikaci i u takto rozsáhlých úloh.

Zároveň práce odhaluje prostor pro další možná rozšíření. Správnost a přesnost modelu je třeba ověřit jeho srovnáním během dalších experimentálních měření a bylo by také vhodné upravit jeho parametry tak, aby se jeho přesnost ještě zvýšila. Další oblastí k rozšíření stávající implementace je například vybavení modelu algoritmy zpětných regulací. Model rovněž reprezentuje chování technologie pouze v okolí definovaného pracovního bodu a za daných pracovních podmínek (viz kap. 4.1). V případě rozšíření oblasti použitelnosti sestaveného modelu by mohl sloužit např. pro testování vyšších forem řízení (např. pomocí predictive control) u podobných technologií.

Práce dokládá, že lze identifikaci systémů použít i u velmi složitých aplikací. Je však třeba respektovat její technická omezení a přizpůsobit realizaci konkrétní technologii, která je předmětem našeho zájmu.

Věřím, že diplomová práce naplnila všechny úkoly, uvedené v jejím zadání.

SEZNAM POUŽITÉ LITERATURY

[1] KAMLER, R. Metody identifikace spojitých modelů technologických soustav. Liberec: Technická univerzita v Liberci, 1999. 146 p.

[2] NOSKIEVIČ, P. Modelování a identifikace systémů.MONTANEX a.s.,1999, ISBN 80-7225-030-02

[3] MODRLÁK, O. Modelování a diskrétní identifikace: studijní materiály. Liberec: Technická univerzita v Liberci, 2004. 31 p.

[4] MODRLÁK, O. Úvod do identifikace: studijní materiály. Liberec: Technická univerzita v Liberci, 2004. 32 p.

[5] ŠTĚPÁN, M. Nelineární aproximace signálů, implementace algoritmu pro nelineární odhad parametrů v Matlabu [HTML dokument] [cit. 2008-10-05]. FEKT VUT v Brně. Dostupný z: <http://www.elektrorevue.cz/clanky/03017/index.html>

[6] PÍŠA, P. Matlab, laboratoř nejen pro matematiky [online] [cit. 2008-10-05] Dostupné z: <http://cmp.felk.cvut.cz/~pisa/Public/ST_matlab.html>

[7] ŠVARC, I. Teorie automatického řízení: obor inženýrská informatika a automatizace. Brno: VUT v Brně, 2003. 89 p.

[8] SCHLEGEL, M. Laplaceova transformace: úvod [online] [cit. 2008-10-01] Dostupné z: <http://control.zcu.cz/~mcech/TS-SM/Laplaceova_transformace.pdf>

[9] DUŠEK, F., HONC,D. Identifikace a linearizace v Simulinku. Pardubice: Univerzita Pardubice,7 p.

[10] MÁŠA, V.,URBAN L. Termická účinnost a měření na experimentální jednotce pro spalování biomasy, Brno, 2007, článek pro odborný seminář Energie z biomasy VII. 6 p.

[11] MÁŠA, V.,URBAN L. Nový typ zařízení pro spalování biomasy a fytomasy II. Zvyšování termické účinnosti a řízení. Brno, 2006, 6 p.

[12] Urban L., Masa V., Pavlas M. and Stehlik P., Novel type of technology for biomass utilization, In:10th Conference on Process Integration, Modelling and Optimisation for Energy Saving and Pollution Reduction PRES 2007, Ischia Porto, Italy, ed Jiri Klemeš, Chemical Engineering Transactions, Volume 12, pp 465– 470, (2007)

[13] PÍSEK, S. Delphi – podrobný průvodce. Praha: Grada Publishing a.s.,2002, ISBN 80-247-0547-8, 323 p.

[14] ŠEJBL, R. Fyzikální analogie popisů modelů spojité simulace. Pardubice, 2006. 66p. Diplomová práce na Dopravní fakultě Jana Pernera, Katedře informatiky v dopravě. Vedoucí diplomové práce Ing. Oktavián Strádal, Ph.D.

[15] WWW stánky Odhad parametrů modelu [online] [cit. 2008-10-01] Dostupné z: <http://www.lermo.cz/pdf/YMMO_pr_10.pdf >

[16] WWW stánky Státnice ČVUT-FEL 2006 Bc [online] [cit. 2008-10-01] Dostupné z: <http://www.bakal06.chytrak.cz/2006_K37.pdf>

[17] WWW stánky Borland Delphi [online] [cit. 2008-10-15] Dostupné z: <http://i-extra.net/programovani/delphi/borland-delphi>

[18] WWW stánky Projekt MPO – Prototyp jednotky o výkonu 1 až 3 MW pro energetické využití různých druhů biomasy a fytomasy [online] [cit. 2008-10-01] Dostupné z: <http://www.upei.fme.vutbr.cz/cs/struktura/sessv-projekt-prototyp-jednotky-o-vykonu-1-az-3-mw-pro-energeticke-vyuziti-ruznych-druhu-biomasy-a-fytomasy.php>

[19] WWW stránky Analysis of the AIC Statistic for Optimal Detection of Small Changes in Dynamic Systems [online] [cit. 2008-10-06]
Dostupné z:
<http://www.chemengr.ucsb.edu/~ceweb/faculty/seborg/pdfs/JSC_ACC04_pub.pdf>

[20] WWW stránky The MIT Press - The Minimum Description Length Principle [online] [cit. 2008-10-06] Dostupné z:
< http://mitpress.mit.edu/catalog/item/default.asp?ttype=2&tid=11155>

[21] WWW stránky Zone.ni.com [online] [cit. 2008-10-06] Dostupné z: < přeloženo z http://zone.ni.com/devzone/cda/tut/p/id/4028#toc2>

SEZNAM PŘÍLOH

[A] MACALÍK, RADEK: *Predzpracovana_data.zip* [CD-ROM]. Brno 2008.

[B] MACALÍK, RADEK: Modely_Simulink.zip [CD-ROM]. Brno 2008.

[C] MACALÍK, RADEK: *MNC_Pascal.zip* [CD-ROM]. Brno 2008.

[D] MACALÍK, RADEK: *Program_Vizualizace.zip* [CD-ROM]. Brno 2008.

[E] Poster – Technologie jednotky (poster.pdf) [CD-ROM]. Brno 2008.