Technická univerzita v Liberci

Fakulta strojní

Diplomová práce

Identifikace OFF a ON-LINE
OFF and ON-LINE Identifikation

Liberec 1999

Michal Slánský

Technická univerzita v Liberci

Fakulta strojní

Katedra aplikované kybernetiky

Obor: 23 - 40 - 8 Automatické systémy řízení ve strojírenství
Zaměření: Automatizace inženýrských prací

Identifikace OFF a ON-LINE

OFF and ON-LINE Identification

Diplomant: Michal Slánský
Vedoucí diplomové práce: Prof. Ing. Miroslav Olehla, CSc.
Konzultant diplomové práce: Ing. Jan Klobouček

Rozsah práce:
Počet stran: 52
Počet příloh: 4

Datum: 27.května 1999

TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta strojní

Katedra aplikované kybernetiky Studijní rok: 1998/99

ZADÁNÍ DIPLOMOVÉ PRÁCE

Jméno a příjmení **Michal S L Á N S K Ý**

obor **23-40-8 Automatizované systémy řízení ve strojírenství**

zaměření **automatizaace inženýrských prací**

Ve smyslu zákona č. 111/1998 Sb. o vysokých školách se Vám určuje diplomová práce na téma:

Identifikace OFF & ON-LINE

Zásady pro vypracování:
(uveďte hlavní cíle diplomové práce a doporučené metody pro vypracování)

1. Seznamte se s literaturou na téma zadání.
2. Vypracujte programy pro off a on-line identifikaci pro zvolené modely (LS, EM, AR, ARMA, ARMAX, GLS).
3. Proveďte návrh modelů k řešení.
4. Porovnejte výsledky modelů pro off a on-line řešení a stanovte výhodnost modelů pro řešení při různé úrovni poruchy na výstupu a měřeném vstupu.
5. Znázorněte graficky (g(t) a f(t)) výsledky získané simulací.

KKY/ASŘ (AIP)
52 s., 12 s. příl., 4 roz. příl. - DISKETY

V 114/99 S
+4 DISKETY

ANOTACE

Tato práce se zabývá porovnáváním modelů pro on-line a off-line identifikaci.

ANOTATION

This diploma thesis deals with comparing of models for on-line and off-line identification.

Prohlášení

Prohlašuji, že jsem diplomovou práci zpracoval samostatně s použitím uvedené literatury.

V Mladé Boleslavi, dne 27.května.1999

Michal Slánský

Rád bych vyjádřil poděkování prof. Miroslavu Olehlovi ,CSc., vedoucímu diplomové práce , za cenné rady a podnětné připomínky k problémům, se kterými jsem se při zpracovávání diplomové práce setkal.

OBSAH

Seznam základních symbolů:

$u(k), x(k), z(k)$	hodnoty získané měřením
$n_2(k), n_3(k), n_4(k)$	poruchy
$z(k)$	bílý šum
$n(k)$	obecný šum
a, b, c, d, f	koeficienty přenosu
$A(z^{-1}), B(z^{-1}), \ldots$	polynomy přenosu
$g(k)$	souřadnice impulzní charakteristiky
$f(k)$	souřadnice přechodové charakt.
$\varepsilon(k), \omega(k), \varphi(k), w(k)$	pomocné proměnné
$\hat{x}(k), \hat{y}(k)$	odhady hodnot $x(k), y(k)$
λ	uroveň šumu/užitečný signál

Úvod

V současné době dochází k rozsáhlé automatizaci v řízení technologických procesů. To s sebou přináší i zvýšené nároky na modernizaci řídících systémů, které se již nesoustřeďují jen pouze na vlastní řízení procesů. Čím dál více vzrůstají nároky na včasnou diagnostiku a předcházení poruchám.

Úloha lidského činitele je však nezastupitelná, ať je úroveň automatizace jakkoliv vysoká. Operátoři musejí sledovat probíhající procesy, optimalizovat chod výroby při změně výrobních podmínek, reagovat na vznik nestandardních situací nebo poruchových stavů a zajistit tak hladký chod výroby. Neboť při jejich případné poruše by mohlo dojít ke značným materiálním škodám nebo i k haváriím ohrožujícím životy lidí. Průmyslová automatizace proto nutně vyžaduje pouze spolehlivé a důkladně ověřené technologie.

Hlavním cílem provozovatele každého zařízení je tedy bezporuchovost. V reálné praxi je však nevyhnutelné počítat se vznikem poruch technologických linek a strojů. Tento nežádoucí stav může negativně ovlivnit celý výrobní proces.

Modernizací výrobního procesu však dochází také k výraznému zvýšení složitosti zařízení a tím i množství informací, které je nutno analyzovat. Při vyhledávání poruchy je nutné, brát v úvahu velké množství snímačů, akčních členů, regulačních součástek a časových posloupností řízení apod. Jejich úlohou je v reálném čase :

- signalizovat poruchy linek a strojů (ať už akusticky, opticky či pouze výpisem chybových hlášení

- určit jejich nejpravděpodobnější příčinu
- případně navrhnout postup na jejich odstranění

V dnešní době panuje trend využívající nadřazených diagnostických počítačů. Takovýto diagnostický systém disponuje většími množnostmi pro implementaci řešení a v případě potřeby dokáže diagnostikovat dokonce více strojů nebo linek zároveň. Je ho možné využívat i na další činnosti jako je monitorování a řízení linky, administrativní práce apod.

Při využití přenosného počítače je možné provádět diagnostiku preventivní či off-line. Při diagnostice off-line se počítač připojuje pouze při vzniku poruchy a jen na dobu, kterou trvá lokalizace příčiny. Tato varianta je sice levnější, ale peníze ušetřené tímto řešením se nemusejí vyplatit. Vlastní porucha může totiž někdy být mnohem dražší a to jak v samotných nákladech na opravu, tak i v možném prostoji vlastní výroby (to může vést až k snížení prestiže dané firmy a následné ztrátě dalších zakázek).

Při řízení technologického provozu rozlišujeme pět úrovní:

- čidla a ovladače
- řízení procesů v reálném čase
- řízení výrobní linky
- řízení provozu
- ERP - celopodnikové systémy

Výzkum prvních dvou úrovní byl velkou měrou podporován hlavně vojenskými zakázkami. Z firem zabývajících se touto oblastí lze jmenovat např. Siemens, ABB, Honeywell nebo Digital Equipment, která právě na zakázkách tohoto typu vyrostla.

V poslední době se i u nás začala velká pozornost věnovat automatickému řízení a to nejen ve strojírenství.

Důkazy pro toto tvrzení jsou naše elektrárny, kde jsou tyto systémy životně důležité, ale i málo známý monitorovací systém pro vodní díla a přehrady, který využívá Povodí Odry, a. s. Ale i přes tyto příklady je hlavní nasazení těchto systémů směřováno na strojírenství výrobní a montážní linky.

Nejznámějším příkladem je na našem území firma Škoda, a.s., která po vstupu zahraničního kapitálu (firmy Volkswagen), byla postavena před nutnost zkvalitnit vlastní výrobu a dosáhnout certifikátu z řady ISO 9000. Tento certifikát se však nezabývá pouze kvalitou konečných výrobků, ale i kvalitou vlastního výrobního procesu. To s sebou logicky přineslo nutnost automatického řízení linek a také vlastní diagnostiku. Protože má firma Škoda, a. s. velkou řadu dodavatelů z naší republiky, byli i oni nuceni (pod hrozbou odebrání zakázek) přistoupit na zpřísněná měřítka kvality. Toto zvýšení nároků na kvalitu vyvolalo tedy i u nich potřebu automatických řídících systému. A tak se díky jedné firmě tento dříve spíše okrajový obor rozšířil a našel své uplatnění v naši republice.

Dnes si již mnoho podniků uvědomilo výhody automatického řízení a je známo, že při dnešních cenách za nové stroje se vlastní automatické řízení vyplatí (nejzákladnější řízení lze provozovat již s počítačích řady PC-386SX, jejichž cena dnes leží pod hranicí 5000,-Kč). Další rozšíření tohoto způsobu řízení ve všech oborech lze i nadále očekávat.

Kapitola 1.

Identifikace systémů

V současné době, kdy se rozšiřuje automatické řízení technologických procesů je třeba včasné diagnostiky (viz. Úvod). Pro co nejlepší řízení a diagnostiku je však vhodné předchozí důkladné poznání stroje či linky (dále již jen neznámého systému). Toto poznávání zahrnuje i jeho vlastní identifikaci.

Identifikace má velké výhody a úspory, protože některá zařízení jsou dnes již tak drahá, že jakékoli pokusy na nich mohou způsobit jejich nenávratné poškození. Pokud nám však daná identifikace poskytne model soustavy, je možné zkoušet jakékoli testy přímo v laboratoři jen na počítači, a posléze jen výsledky ověřit na systému bez nebezpečí poškození. Tímto postupem lze například stanovit nejlepší druh regulátoru a jeho optimální nastavení tak, aby co nejlépe vyhovoval požadavkům, které na ně klademe.

Lze tak také simulovat i některé havarijní stavy a zjišťovat odezvu na tyto vstupní signály. Předem lze tedy stanovit co se stane se systémem, když se například zasekne vstup do systému apod.

1.1. Identifikace dynamických soustav

Pod vlastním pojmem identifikace technologických soustav si lze představit zjišťování vlastností soustavy, která nám poskytuje statické a dynamické charakteristiky soustav, matematický či logický popis jejího chování atd.

Identifikace sama o sobě by však neměla žádný velký význam bez simulace a naopak. Z toho tedy plyne, že to jsou vlastně dva nedělitelné celky.

Při vytváření modelu lze postupovat dvěma způsoby:

1) analyticky, fyzikálně-matematickou analýzou objektu
2) experimentálně, na základě experimentálně získaných údajů z procesu

Při analytickém postupu se vychází ze známých fyzikálních a chemických procesů. Pro tuto metodu je však nutná velmi dobrá znalost identifikované soustavy a znalost všech procesů v ní obsažených. V případě příliš složitého systému je tento problém velmi obtížný pro nutnost zavedení mnoha doplňkových předpokladů. Tyto předpoklady jsou zaváděny až po podrobném prozkoumání systému.

K základním metodickým problémům identifikace patří:

a) klasifikace objektů z hlediska identifikace a klasifikace identifikačních metod z hlediska jejich aplikovatelnosti

b) volba a vytvoření identifikačních metod

c) plánování experimentu :

- volba a generování testovacích signálů
- určení intervalu vzorkování a kvantování signálu
- volba optimálního počtu experimentálních vzorků
- vymezení platnosti modelu

d) metody prvotního zpracování údajů

- filtrace údajů
- korelace údajů
- komprese údajů, atd.

1.2. Experimentální identifikace

Cílem této práce je porovnání různých modelů pro experimentální identifikaci.

Tato identifikace staví na údajích získaných buď přímo při vlastní funkci soustavy v normálních provozních podmínkách nebo jen při zkušebním (experimentálním) chodu. Model je tedy poté určen na základě získaných údajů ze vstupu a výstupu signálu ze soustavy.

Hlavní výhodou tohoto přístupu je, že nemusí být o daném systému nic jiného známo. Stačí nám tedy pouze naměřené hodnoty. A ani u vstupního a výstupního signálu nám nezáleží na to jakého typu je. Záleží pouze na jeho hodnotě a jejich vzájemném provázání.

Vstupní signály při identifikaci mohou být:

a) deterministické aperiodické a periodické

b) náhodné signály (stacionární, ergodické, nestacionární, neergodické)

c) pseudonáhodné signály dvouúrovňové či víceúrovňové

Experimentální metody dělíme do dvou skupin:

a) deterministické, které zanedbávají vliv poruchových signálů. K výpočtu například impulsní charakteristiky či diferenční rovnice se využívá jen tolik údajů o vstupu a výstupu, kolik je hledaných parametrů

b) statistické, umožňující kvalitativní zhodnocení chyb podle statistických hledisek

1.3. Identifikace deterministických soustav

Tato identifikace je jednorázová a je tedy použitelná jen u soustav, kde nedochází ke změnám parametrů soustavy v čase. V případě, že se projevuje i vliv náhodných veličin, provádí se identifikace buď **off-line** (dálkově), nebo **on-line** (průběžně).

V případě **off-line** metody stačí zaznamenat vstupní a výstupní data na jakémkoliv měřícím a zapisovacím zařízení a následné vyhodnocení soustavy a její identifikace probíhá až v laboratoři na počítači. V tomto případě jsou tedy data vyhodnocována až následně s časovým zpožděním. Tento způsob má svou výhodu v tom, že jsou takto získaná data možná přímo vyhodnotit a hned v laboratoři, kde jsou k tomu účelu většinou již vhodné prostředky, také okamžitě analyzovat např. na případné poruchy apod. Další výhodou tohoto způsobu je také fakt, že pro sledování několika strojů je potřebné pouze jedno měřící a zapisovací zařízení a pouze jediné vyhodnocovací zařízení, čímž se velmi snižují náklady na jeho pořízení do závodu.

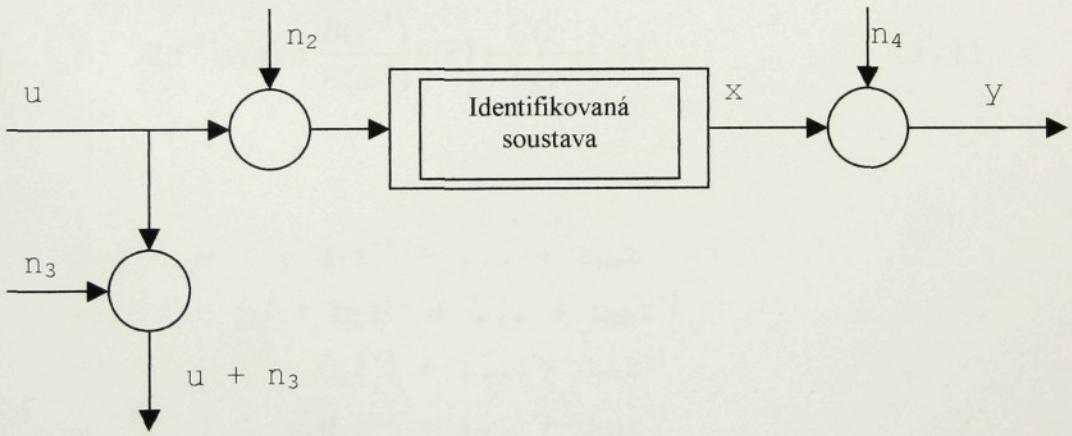
Při metodě **on-line** se využívá měření v reálném čase. To však vyžaduje řídící počítač, který je v neustálém spojení se soustavou. Tato metoda je využívána především pro soustavy v proměnlivých podmínkách, na kterých závisí parametry určovaného modelu. Tím je možno rychle reagovat na změny a adaptivně měnit například nastavení regulátorů či zapínat jiné podpůrné prostředky. Tento systém má význam i pro dlouhodobé sledování, kdy se parametry mění pomalu, ale není možno znovu provést měření (např. u spotřebičů, které již má zákazník doma).

Kapitola 2.

Určení modelu pro obecný vstupní signál

Tato práce se zabývá jen identifikací jednorozměrných lineárních soustav (SISO) se soustředěnými parametry. To znamená, že pro jednu vstupní veličinu je pouze jediná výstupní veličina. Příkladem takovéto soustavy je například elektromotor, který má jako vstupní veličinu napětí a výstupní jsou otáčky.

Pro určování těchto modelů používáme jako vstupní signál obecný vstup generovaný nebo z naměřené technologické veličiny.



Obr. 2.1.

n_2, n_3, n_4 - poruchy (na vstupu, měřeném výstupu a vstupu)
u - vstupní veličina
x, y - výstupní veličina a výstupní měřená veličina

Na obrázku 2.1. je naznačeno schéma určované soustavy a jsou zde naznačeny i možné poruchy ovlivňující vlastní měření a tím i výsledný model. Poruchy n_2 a n_4 mohou být způsobeny vlastními měřícími prvky nebo vedením k nim a porucha n_3 je zpravidla ve vlastním vedení, které předává řídící signál do soustavy do místa měření nebo místa řízení.

2.1. Struktury modelů pro výpočet koeficientů přenosu

Pro výpočet koeficientů přenosu je možno využít modely s různou strukturou. Tyto modely jsou však pouze jakýmsi zjednodušením obecného modelu. Tento obecný tvar modelu lze zapsat ve tvaru :

$$A(z^{-1})y(k) = \frac{B(z^{-1})}{F(z^{-1})}u(k) + \frac{C(z^{-1})}{D(z^{-1})}z(k) \qquad (2.1)$$

kde

$A(z^{-1}) = 1 + a_1 z^{-1} + \ldots + a_{na} z^{-1}$

$B(z^{-1}) = b_0 + b_1 z^{-1} + \ldots + b_{nb} z^{-1}$

$C(z^{-1}) = 1 + c_1 z^{-1} + \ldots + c_{nc} z^{-1}$

$D(z^{-1}) = 1 + d_1 z^{-1} + \ldots + d_{nd} z^{-1}$

$F(z^{-1}) = 1 + f_1 z^{-1} + \ldots + f_{nf} z^{-1}$

$z(k)$ ... bílý šum

$y(k)$ ... výstupní veličina

$u(k)$ ... vstupní veličina

Tento model je však pro většinu případů příliš obecný a pro účely identifikace lze některé polynomy položit rovny nule bez podstatného vlivu na výsledek.

2.2. Vlastní typy modelů

V této práci je soustředěna pozornost pouze na některé typy modelů, které jsou zjednodušením obecného modelu (2.1) z předchozí části. Jedná se modely LS, ARMA, ARMAX (EM), GLS a okrajově i o model AR.

2.2.1. AR-model

Tento model se využívá pouze tehdy jeli vlastní vstupní signál neměřitelný, z toho také plyne jeho název AR jako AutoRegressive proces. Výsledný model neobsahuje žádný z polynomů, ve kterých je zohledněn vstupní signál a proto je pro účely simulace zcela nevhodný.

pro $F(z^{-1}) = D(z^{-1}) = C(z^{-1}) = 1$ *a* $B(z^{-1}) = 0$ *pak*

$$A(z^{-1})y(k) = z(k) \qquad (2.2.1.)$$

2.2.2. LS-model

Často využívaným modelem je model LS, také označovaný jako LR - linear regression.

pro $F(z^{-1}) = D(z^{-1}) = C(z^{-1}) = 1$ *pak*

$$A(z^{-1})y(k) = B(z^{-1})u(k) + z(k) \qquad (2.2.2.)$$

2.2.3. ARMAX-model

Tento model získal svůj název jako odvozenina z dílčích názvů pro jednotlivé části. **AR** je označení pro $A(z^{-1})y(k)$,

název **MA** (moving average) náleží poruše z(k) a konečně **X** jako exogenerous proměnná pro u(k), dohromady tedy ARMAX.

pro $F(z^{-1}) = D(z^{-1}) = 1$ *pak*

$$A(z^{-1})y(k) = B(z^{-1})u(k) + C(z^{-1})z(k) \qquad (2.2.3.)$$

2.2.4. GLS-model

Název tohoto modelu je odvozen ze slov **g**eneralized **l**east **s**queres, občas bývá použito také názvu DA, jako dynamic adjustment.

pro $F(z^{-1}) = C(z^{-1}) = 1$ *pak*

$$A(z^{-1})y(k) = B(z^{-1})u(k) + \frac{1}{D(z^{-1})}z(k) \qquad (2.2.4.)$$

2.2.5. ARMA-model a EM-model

Nejobecnějším ze všech sledovaných modelů je model ARMA nebo také EM. Tento model má dvojí označení, neboť první je vžitý název pro off-line identifikaci a druhý je nejčastěji používaný název pro on-line. Jedná se tedy o jeden jediný model. Tento model vznikl jako sloučení předchozích dvou modelů ARMAX a GLS.

pro $F(z^{-1}) = 1$ *pak*

$$A(z^{-1})y(k) = B(z^{-1})u(k) + \frac{C(z^{-1})}{D(z^{-1})}z(k) \qquad (2.2.5.)$$

Kapitola 3.

Postup řešení

Tato část je společná pro všechny předem uvedené modely, je tedy vhodné zabývat se možností výpočtu řešení pro obecný tvar. Vlastní řešení jednotlivých modelů pak je pouhým zjednodušením tohoto postupu a to na základě vlastností daného model.

3.1. Úprava modelů k řešení

Uvažujeme tedy obecný model ve tvaru (2.1.) :

$$A(z^{-1})y(k) = \frac{B(z^{-1})}{F(z^{-1})}u(k) + \frac{C(z^{-1})}{D(z^{-1})}z(k) \qquad (3.1.1.)$$

Pro vlastní řešení je potřebné tuto rovnici (3.1.1.) upravit do následujícího tvaru:

$$\frac{A(z^{-1})D(z^{-1})}{C(z^{-1})}y(k) = \frac{B(z^{-1})D(z^{-1})}{F(z^{-1})C(z^{-1})}u(k) + z(k) \qquad (3.1.2.)$$

Nyní je tedy úkolem určit vektor neznámých parametrů ω (což je vektor koeficientů pro dané polynomy):

$\omega^T = (a_1 \ldots a_{na}, b_1 \ldots b_{nb}, f_1 \ldots f_{nf}, c_1 \ldots c_{nc}, d_1 \ldots d_{nd})$

na, nb, nc, nd a nf ... tyto koeficienty vyjadřují řád daného polynomu

Chybu mezi vypočtenou a naměřenou hodnotou resp. predikcí lze určit pomocí vztahu:

$$\varepsilon(k) = y(k) - \hat{y}(k) \quad \text{, kde} \qquad (3.1.3.)$$

$$\hat{y}(k) = \frac{D(z^{-1})}{C(z^{-1})}\left[A(z^{-1})y(k) - \frac{B(z^{-1})}{F(z^{-1})}u(k)\right] \qquad (3.1.4.)$$

Pro jednodušší řešení je potřeba zavést pomocné proměnné, které mají za účel pouze usnadnit pozdější vlastní řešení vlastních modelů.

$$w(k) = \frac{B(z^{-1})}{F(z^{-1})}u(k) \qquad (3.1.5.)$$

$$v(k) = A(z^{-1})y(k) - w(k) \qquad (3.1.6.)$$

pak lze $\varepsilon(k)$ vyjádřit ve tvaru:

$$\varepsilon(k) = \frac{D(z^{-1})}{C(z^{-1})}v(k) \qquad (3.1.7.)$$

Tvar řádku matice pro soustavu rovnic má podobu:

$$\varphi^{T}(k) = -y(k-1) \ldots -y(k-na),\ u(k) \ldots u(k-nb),$$
$$-w(k-1) \ldots -w(k-nf),\ \varepsilon(k-1) \ldots \varepsilon(k-nc),$$
$$-v(k-1) \ldots -v(k-nd) \qquad (3.1.8.)$$

Hodnoty u(k), y(k) jsou získány měřením, hodnoty pomocných proměnných získáme pomocí výše uvedených vztahů (3.1.5.), (3.1.6.) a (3.1.7.).

$$w(k)=b_0u(k-1)+\ldots+b_{nb}u(k-nb)-f_1w(k-1)-\ldots- f_{nf}w(k-nf) \qquad (3.1.9.)$$

$$v(k)=y(k)+a_1y(k-1)+\ldots+a_{na}y(k-na)-w(k) \qquad (3.1.10.)$$

$$\varepsilon(k)=v(k)+d_1v(k-1)+\ldots+d_{nd}v(k-nd) \\ -c_1\varepsilon(k-1)-\ldots- c_{nc}\varepsilon(k-nc) \qquad (3.1.11.)$$

Z rovnic (3.1.8.) a (3.1.9.), lze pro ε(k) získat též obecný tvar:

$$\varepsilon(k) = y(k) - \hat{y}(k) = y(k) - \varphi^T(k)\omega \qquad (3.1.12.)$$

3.2. Vlastní výpočet

Při vlastním výpočtu koeficientů přenosu pro daný model máme většinou více rovnic než je proměnných. Přičemž počet rovnic přímo závisí na počtu měření.

Řešení je poté možno zapsat v maticovém tvaru:

$$\mathbf{y} = \phi \cdot \omega \qquad (3.2.1.)$$

kde, $\mathbf{y}^T=(y(0),y(1),y(2) \ \ldots \ y(N))$

$\omega^T=(a_1\ldots a_{na}\ ,\ b_0\ldots b_{nb},\ f_1\ldots f_{nf},\ c_1\ldots c_{nc},\ d_1\ldots d_{nd})$

pro

$$N \geq na + nb + nc + nd + nf \qquad (3.2.2.)$$

N ... je počet měření

Matice ϕ je složená z řádků, které mají tvar dle rovnice (3.1.8.).

$$\phi = \begin{bmatrix} \varphi^T(0) \\ \varphi^T(1) \\ . \\ \varphi^T(k) \\ . \\ \varphi^T(N) \end{bmatrix} \qquad (3.2.3.)$$

3.3. Úpravy pro řešení off-line

Pokud ve vztahu (3.2.2.) platí ostrá nerovnost získáme N rovnic pro (na+nb+nc+nd+nf) neznámých. Tím je tato soustava rovnic přeurčená.

Hlavní problém nastává hlavně při off-line identifikaci, kdy bývá naměřených hodnot velmi mnoho (při on-line metodě je dalšími získanými daty model pouze zpřesňován).

Pro vlastní řešení je tedy potřeba upravit matici ϕ a vektor **y**. Nejvhodnější je, upravit matici ϕ na čtvercový tvar vynásobením zleva transponovanou maticí ϕ^T. Výsledná určovaná rovnice má potom tvar:

$$\phi^T.\mathbf{y} = \phi^T.\phi.\omega \qquad (3.2.4.)$$

Další možná a použitá úprava v této práci je využití iteračních kroků na základě zobecněné metody nejmenších čtverců. Tato metoda využívá možnosti filtrovat naměřené hodnoty u(k) a y(k) vypočteným filtrem a dále identifikovat

takto upravené hodnoty. Tím lze získat přesnější odhad koeficientů modelu.

Tato metoda považuje korelovanou chybu n(k) za výstup filtru s přenosem $C(z^{-1})/D(z^{-1})$, na jehož vstupu působí bílý šum z(k). Pak platí:

$$n(k) = \frac{C(z^{-1})}{D(z^{-1})} z(k) \qquad (3.2.5.)$$

Hledaná rovnice lze potom zapsat ve tvaru:

$$A(z^{-1})y(k) - B(z^{-1})u(k) = n(k) \qquad (3.2.6.)$$

Po dosazení a úpravě dostáváme tedy rovnici v této podobě:

$$A(z^{-1})\frac{D(z^{-1})}{C(z^{-1})} y(k) - B(z^{-1})\frac{D(z^{-1})}{C(z^{-1})} u(k) = z(k) \qquad (3.2.7.)$$

Nyní je znovu nejvhodnějším řešením substituce. Nabízejí se zde rovnou dvě možnosti:

a) $A^*(z^{-1}) = A(z^{-1})\frac{D(z^{-1})}{C(z^{-1})}$ a $B^*(z^{-1}) = B(z^{-1})\frac{D(z^{-1})}{C(z^{-1})}$, potom ale velmi rychle roste řád polynomů $A^*(z^{-1})$ a $B^*(z^{-1})$.

b) $y^*(k) = \frac{D(z^{-1})}{C(z^{-1})} y(k)$ a $u^*(k) = \frac{D(z^{-1})}{C(z^{-1})} u(k)$, tímto postupem nedochází ke zvyšování řádu polynomů $A(z^{-1})$ a $B(z^{-1})$, ale pouze k zpřesňování jejich odhadů z přepočtených dat y(k) a u(k). Tato metoda byla využita i v této práci.

3.4. Vlastní postup řešení off-line

Při použití této varianty identifikace je velmi důležité sestavit přeurčenou matici ϕ a dále postupovat dle předchozího navrženého postupu za využití varianty b) pro iterace.

3.5. Úpravy pro řešení on-line

Nejvhodnějším postupem pro tuto metodu je také využití zobecněné metody nejmenších čtverců či metody rozšířené matice. Pro tuto práci byla vybrána metoda rozšířené matice pro své použití pro všechny sledované modely.

Při této metodě se předpokládá (jako při off-line identifikaci), že šum lze vyjádřit ve tvaru:

$$n(k) = \frac{C(z^{-1})}{D(z^{-1})} z(k) \qquad (3.5.1.)$$

Pak může být soustava popsána vztahy:

$$y(k) = x(k) + n(k) \quad \text{, kde} \quad \sum_{i=0}^{n} a_i x(k-i) = \sum_{i=0}^{m} b_i u(k-i) \qquad (3.5.2,3)$$

čili

$$\sum_{i=0}^{n} a_i y(k-i) - \sum_{i=0}^{m} b_i u(k-i) = \sum_{i=0}^{n} a_i n(k-i) \qquad (3.5.4.)$$

$$r(k) = \sum_{i=0}^{n} a_i n(k-i) \qquad (3.5.5.)$$

kde výstup šumu je reprezentován r(k), které může být určeno z bílého šumu z(k) podle vztahu:

$$r(k)+\sum_{i=1}^{p} d_i r(k-i) = z(k)+\sum_{i=1}^{q} c_i z(k-i) \qquad (3.5.6.)$$

Při a_0=1 pak

$$y(k)=\sum_{i=1}^{n} -a_i y(k-i)+\sum_{i=0}^{m} b_i u(k-i)+\sum_{i=1}^{q} c_i z(k-i)+\sum_{i=1}^{p} d_i r(k-i)+z(k) \qquad (3.5.7.)$$

v maticovém zápisu má rovnice (3.5.7.) potom tvar:

$$y=\begin{bmatrix} U & Y & Z & R \end{bmatrix}\begin{bmatrix} +b \\ -a \\ +c \\ -d \end{bmatrix}+z=\Omega\omega+z \qquad (3.5.8.)$$

$$\omega=\left(\Omega^T\Omega\right)^{-1}\Omega^T x \qquad (3.5.9.)$$

Protože hodnoty r(k) a z(k) jsou neznámé a nedají se získat měřením nahrazují se jejich odhadem po každé aproximaci parametru c.

$$\hat{r}(k)=y(k)+\sum_{i=1}^{n} a_i y(k-i)-\sum_{i=0}^{m} b_i u(k-i) \qquad (3.5.10.)$$

$$\hat{z}(k)=\hat{r}(k)+\sum_{i=1}^{p} d_i \hat{r}(k-i)-\sum_{i=1}^{q} c_i \hat{z}(k-i) \qquad (3.5.11.)$$

3.6. Vlastní postup řešení on-line

Při této metodě přicházejí data průběžně a tak se také vyhodnocují. Nejprve jsou určeny hlavní koeficienty na jejichž základě se počítají další. Pokud jsou tedy již všechny koeficienty známy, dochází v následném kroku k jejich zpřesňování.

Hlavní nevýhodou této metody je to, že odchylka $\varepsilon(k)$ je použita až pro následný řádek $k+1$, čímž se do celého řešení vnáší chyba. Ta se však odstraní po velkém počtu měření, kdy rozdíly mezi dvěmi sousedními odhady modelu jsou již zanedbatelné tak, že $\varepsilon(k) \approx \varepsilon(k+1)$.

Kapitola 4.

Výchozí předpoklady

Pro možnost porovnání modelů je nutné identifikovaný model předem znát. Já jsem pro potřeby této práce zvolil shodný model s použitou literaturou, a to hlavně z důvodů možnosti porovnání výsledků. Volený identifikovaný model nemá na výsledky této práce žádný vliv. Lze zobecnit.

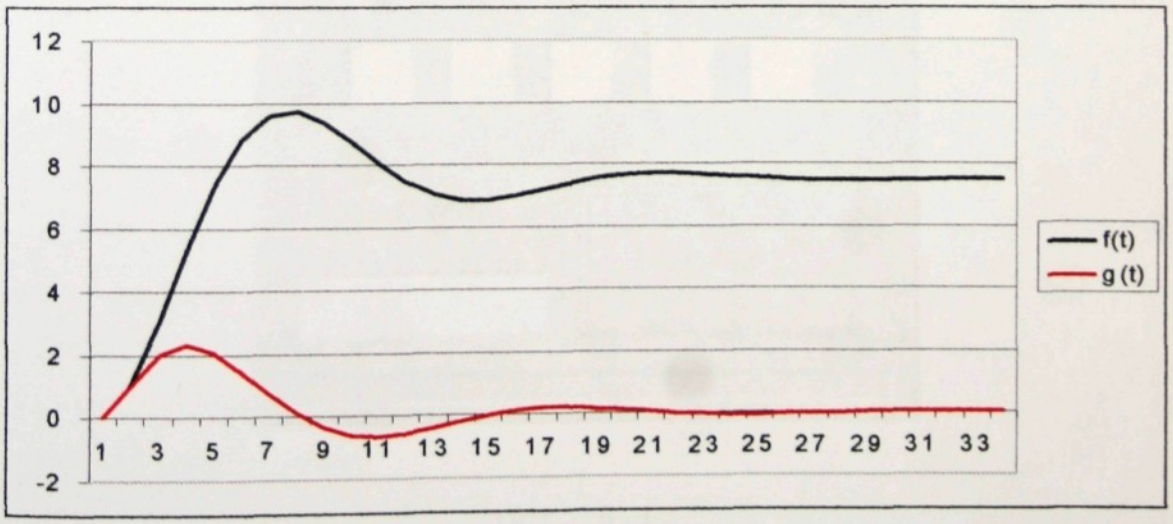
Přenosu tohoto systému má tvar:

$$G = \frac{1+0{,}5.z^{-1}}{1-1{,}5.z^{-1}+0{,}7.z^{-2}} \qquad (4.1.)$$

Z tohoto přenosu plyne, že hledané koeficienty při identifikaci by měly mít tyto hodnoty:

$a_0=1$, $a_0=-1{,}5$, $a_0=0{,}7$ a $b_0=1$, $b_1=0{,}5$

Pro tento model má přechodová a impulzní charakteristika následující podobu:

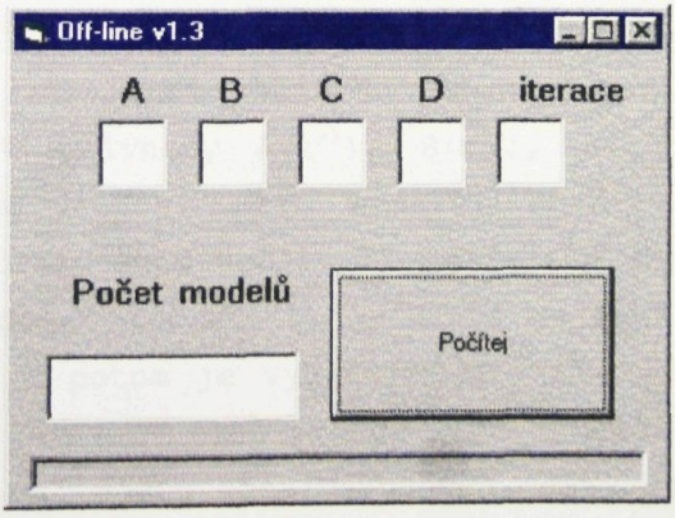


Graf 4.1.

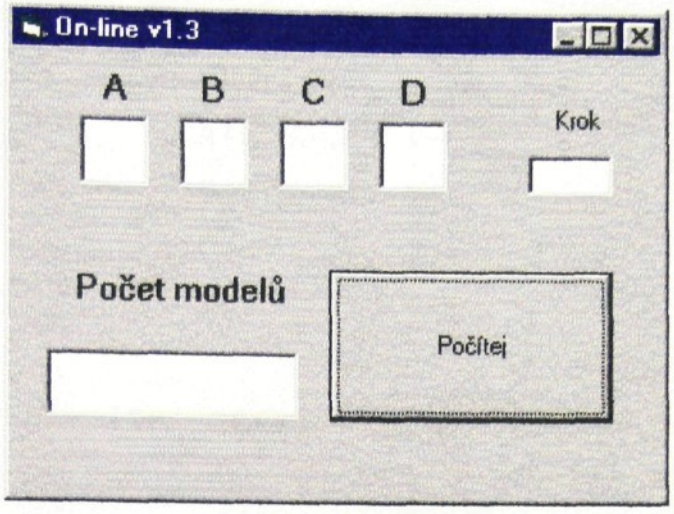
S těmito hodnotami jsou tedy následující vyšlé modely porovnávány. Pro větší přehlednost je ve všech grafech zachováno barevné označení těchto křivek: černě přechodová a červeně impulzní charakteristika. Nadále již budou obě charakteristiky od sebe odděleny z důvodu větší přehlednosti a lepší orientace. Rozdělení obou grafů má také tu výhodu, že obě charakteristiky jsou nestejně velké, a tak by mohlo docházet ke zkreslení výsledků.

4.1. Identifikační programy

Pro potřeby identifikace byly vypracovány programy ve Visual Basicu 5.0. Tyto programy hledají koeficienty přenosu pomocí výše uvedeného postupu a výsledky jsou ukládány do textového souboru.



Obr. 4.1.



Obr 4.2.

Výhodou práce s Visual Basicem je snadné vytvoření programu pro systém Windows 95/8 nebo Windows NT.

Vlastní programy jsou koncipovány tak, aby jejich ovládání bylo intuitivní. Program si sám na začátku vyžádá potřebné informace a samotné spuštění je pak provedeno pouze stiskem jediného tlačítka (viz obr. 4.1.a 4.2.).

V okně programu jsou při jeho běhu znázorňovány všechny potřebné údaje o průběhu identifikace. To znamená v obou případech je zde vidět velikost polynomů právě určovaného modelu (pro polynomy $A(z^{-1})$, $B(z^{-1})$, $C(z^{-1})$, $D(z^{-1})$), celkový a zbývající počet modelů. Rozdíly jsou pouze v jediném zobrazovaném údaji. Pro off-line identifikaci je znázorňován počet iteračních kroků (jejichž množství je možno omezit a potom je vybrán model, který měl nejmenší čtverec odchylek). V okně pro on-line je znázorňován krok, který odpovídá počtu analyzovaných dat se vstupní a výstupní veličinou.

Výpis zdrojového kódu pro jednotlivé programy je umístěn v příloze č.1. a č.2..

Oba programy jsou předem nastaveny na výpočet 900 modelů a to pro velikosti polynomů:

$A(z^{-1})$ ... 1-6

$B(z^{-1})$ ... 1-6

$C(z^{-1})$ ... 0-4

$D(z^{-1})$ ... 0-4

Toto nastavení lze opravit pouze přímo ve vlastním programu (a to jen z důvodu pro snadnější manipulaci). Tím by měla být data dostatečně analyzována a mělo by být vytvořeno dostatečné množství modelů pro rozbor identifikovaného systému. Toto nastavení je vytvořeno tak, aby soustava byla identifikována menšími i většími polynomy, než jí ve skutečnosti tvoří.

Nevýhodou takto sestaveného programu je jedině délka doby trvání jejího chodu. Pro potřeby této práce byly modely počítány na dvou počítačích třídy Pentium, a zde se doba výpočtu pro 1000 hodnot o vstupu a výstupu při identifikaci již uvedených 900 modelů pohybovala okolo 8 hodin (na rychlejším z nich s procesorem AMD K6-166Mhz a 32MB RAM).

Při výpočtu metodou off-line pro identifikaci systému byl nastaven maximální počet iterací na hodnotu **31**. Iterační cyklus byl přerušen pokud se všechny určované koeficienty v následujícím kroku nezměnily o více než 0,001 (splňovalo 75% všech počítaných modelů). Pokud ani při dosažení tohoto počtu iterací nebylo nalezeno optimální řešení, byla vybrána z celkového množství iteračních modelů ta varianta, která měla nejmenší kvadrát odchylek.

Kapitola 5.

Vyhodnocování vlastního měření

Jak již bylo výše uvedeno, dané programy počítají z jedné sady dat 900 modelů. Z tohoto počtu je vždy 36 modelů typu LS, 144 typu ARMAX a stejný počet typu GLS. Na ARMA-model tedy připadá zbývajících 576 modelů.

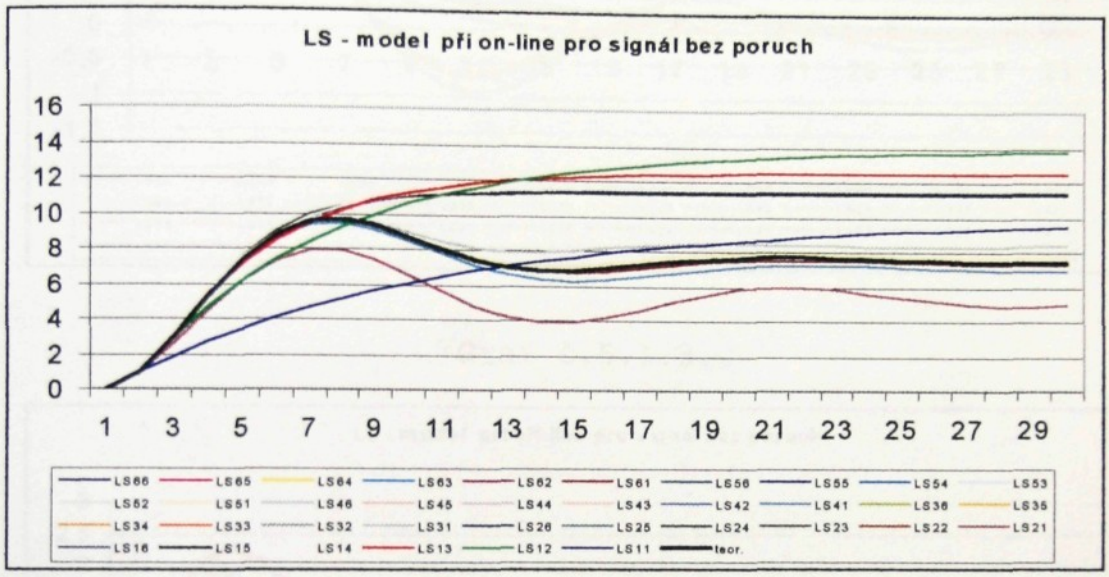
Dané modely byly testovány na předem připravených 5 různých souborech se vstupním a výstupním signálem, každým o maximální délce 1000 hodnot. Dva z těchto souborů byly zatíženy chybou na vstupu, dva na výstupu a poslední byl bez poruch. Přičemž se i soubory, které byly zatíženy chybou, od sebe lišily právě velikostí poměru úrovně šumu vůči užitečnému signálu (λ). V prvním případě byla tato úroveň λ_1 rovna 1, v druhém je λ_2 rovno 5.

Z toho plyne, že pro potřeby této práce bylo spočítáno 4500 modelů pro jednu metodu identifikace (on/off-line). Což dohromady činí 9000 modelů, které byly porovnávány s impulzní a přechodovou charakteristikou dle grafu 4.1. (což činí 18 000 porovnávání).

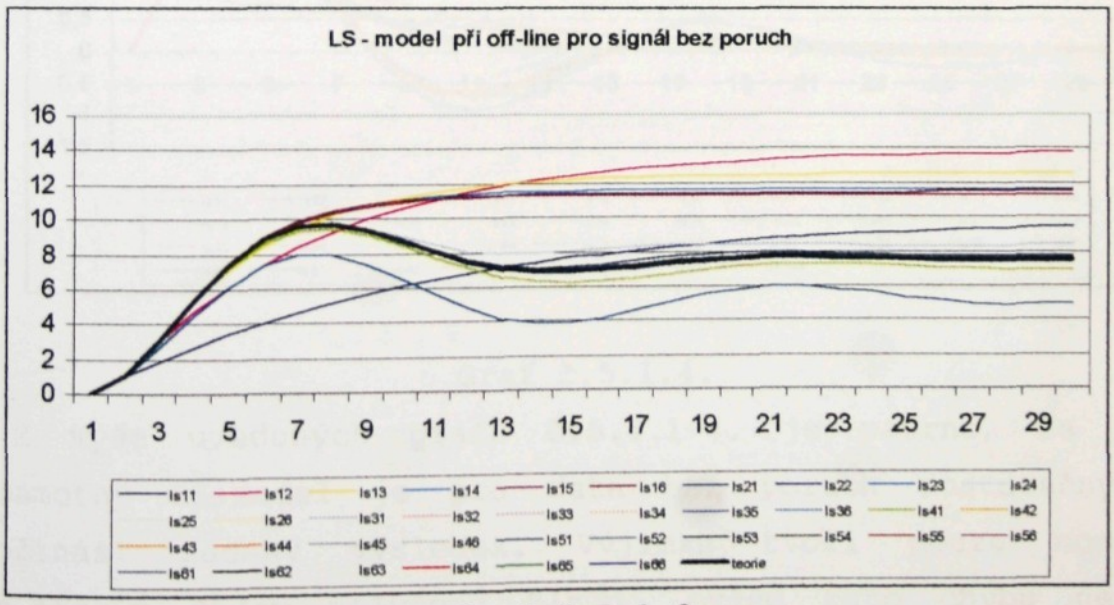
Je jasné, že takovéto obrovské množství dat by bylo neúnosné vkládat do této práce, a tak byly vybrány pouze charakteristické případy.

5.1. LS-model

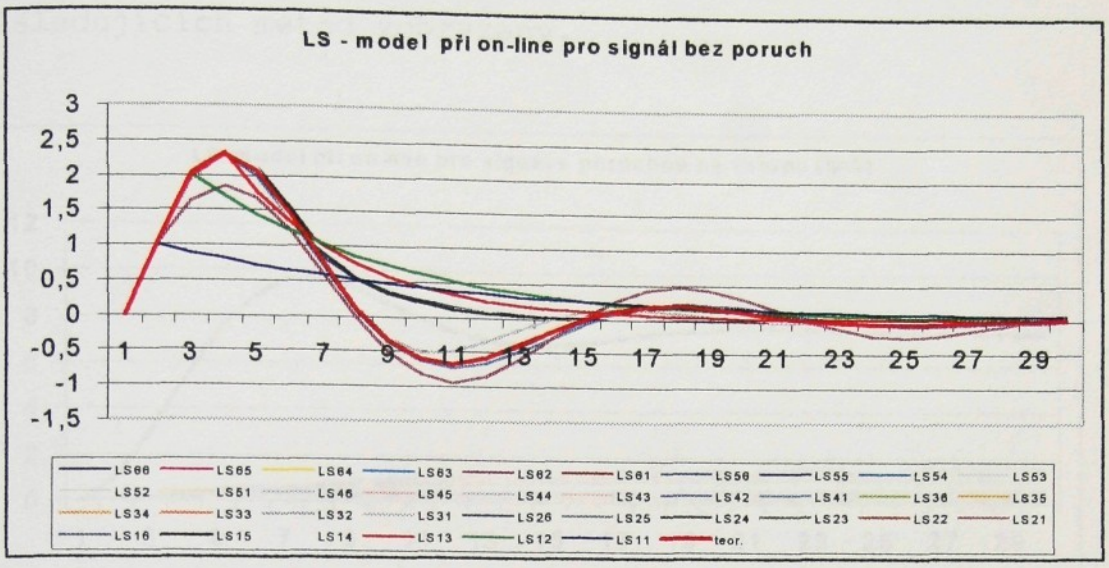
Označení LS-modelů v grafech vychází z velikosti jednotlivých polynomů. První číslice tedy znamená velikost polynomu $A(z^{-1})$, druhá polynomu $B(z^{-1})$.



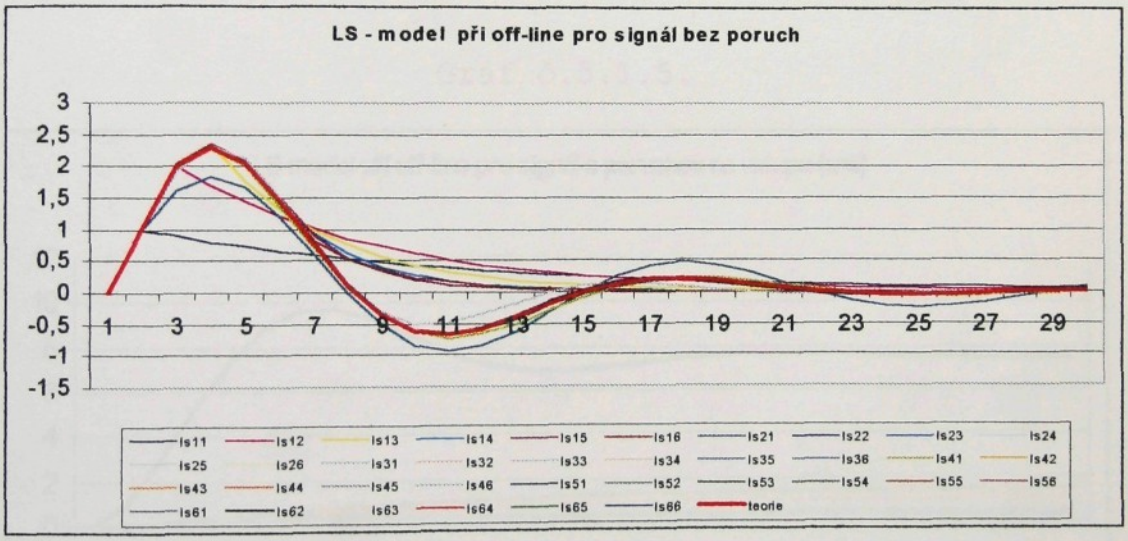
Graf č.5.1.1.



Graf č.5.1.2.



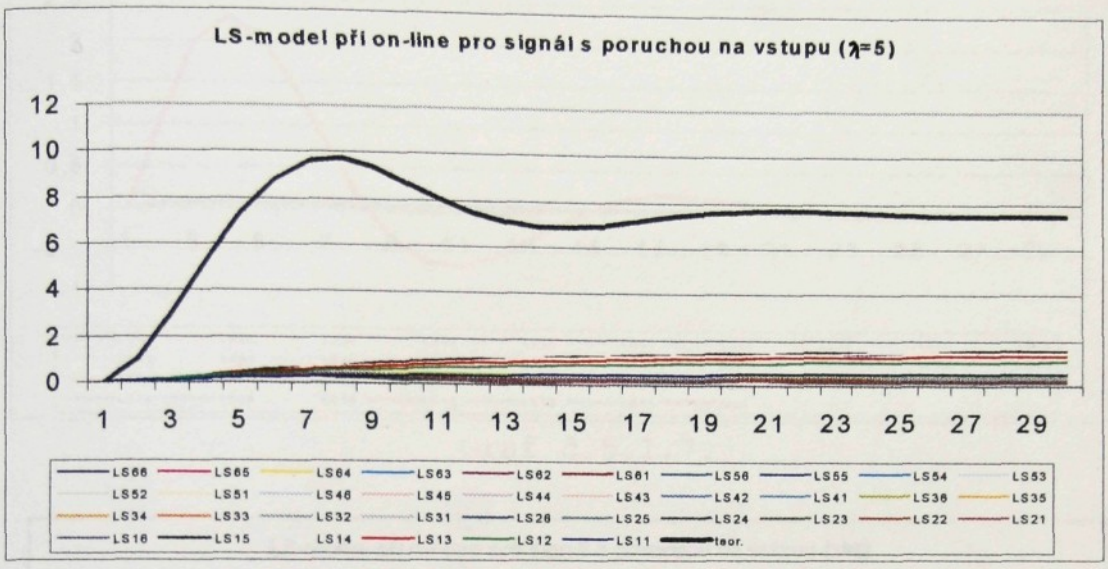
Graf č.5.1.3.



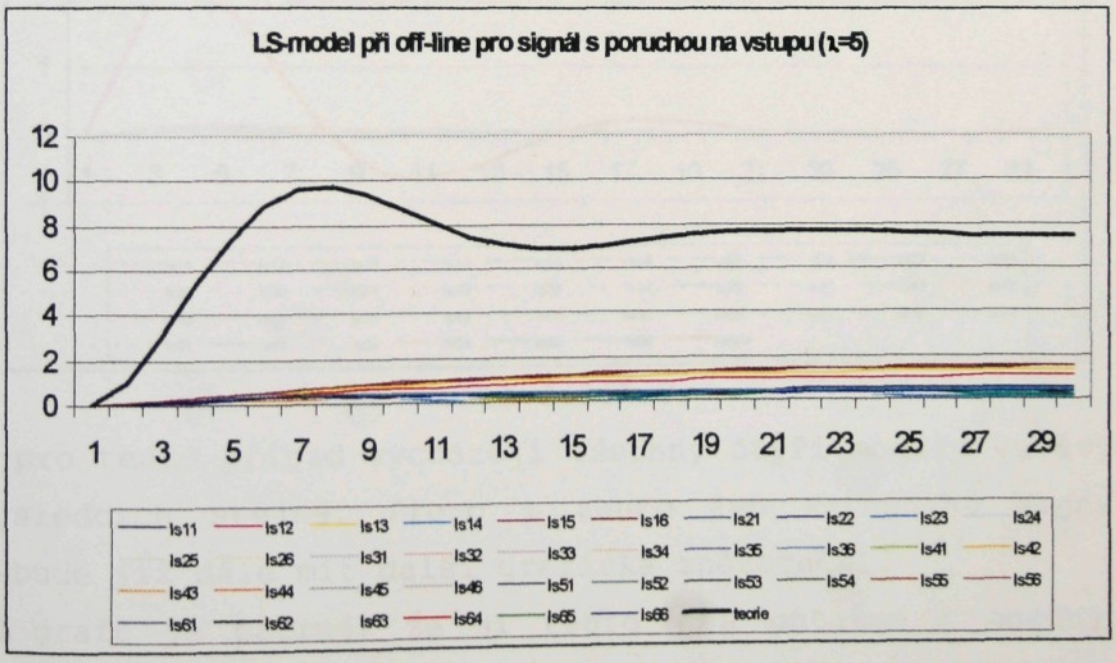
Graf č.5.1.4.

Z výše uvedených grafů č.5.1.1-4. je patrné, že již samotný LS-model je pro data bez poruch dostatečný a přináší žádaný výsledek. Výjimku tvoří pouze modely s nízkým řádem polynomu $B(z^{-1})$, ovšem tuto chybu nelze odstranit ani vyššími řády polynomů $C(z^{-1})$ a $D(z^{-1})$ (toto tvrzení bylo potvrzeno výsledky u určovaných modelů). A

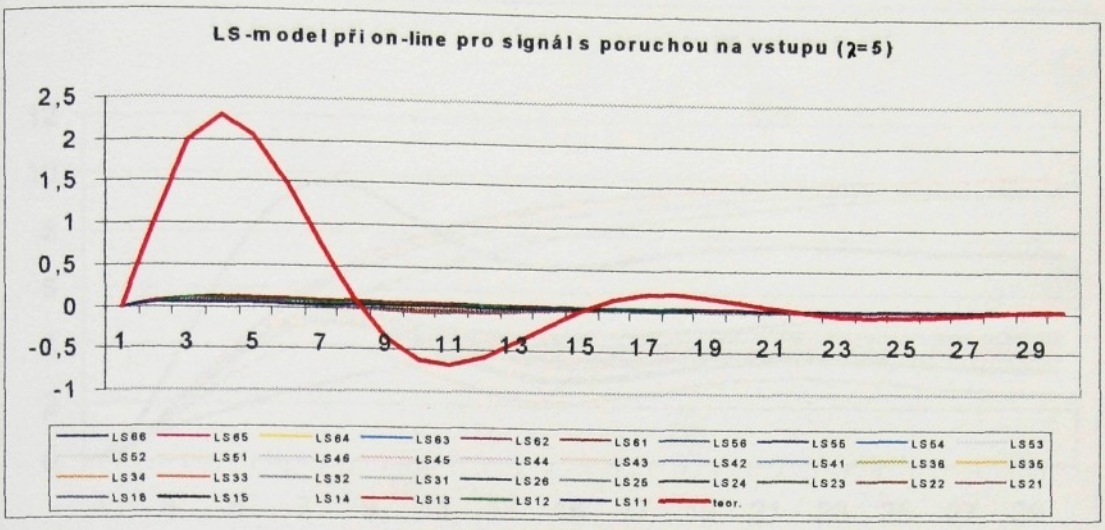
proto další grafy s těmito daty bez poruch již nebudou u následujících metod zobrazeny.



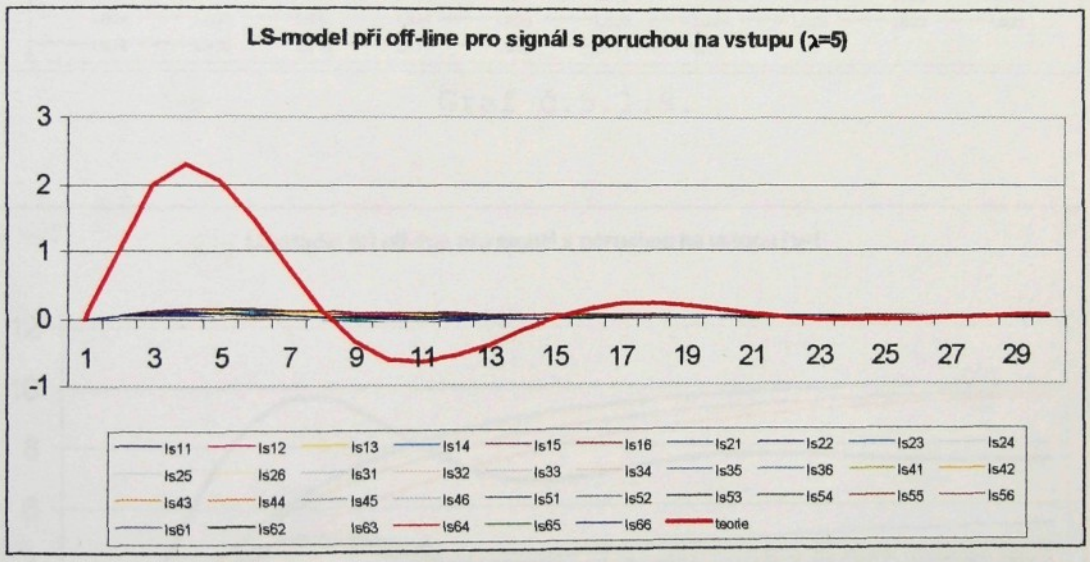
Graf č.5.1.5.



Graf č.5.1.6.



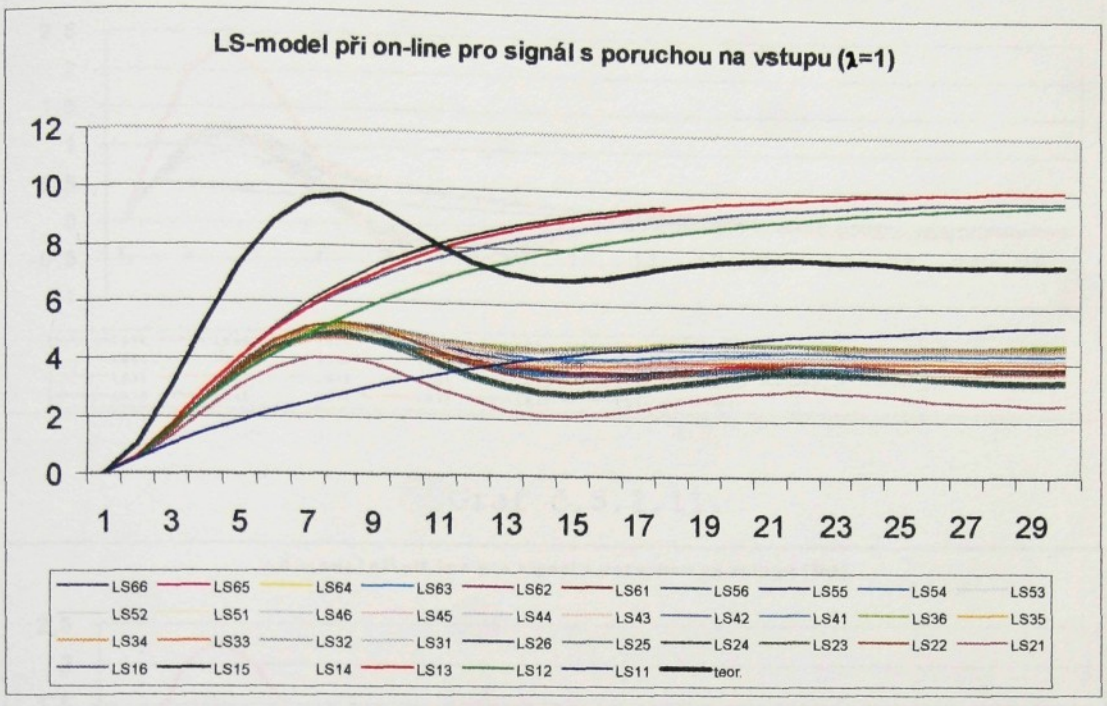
Graf č.5.1.7.



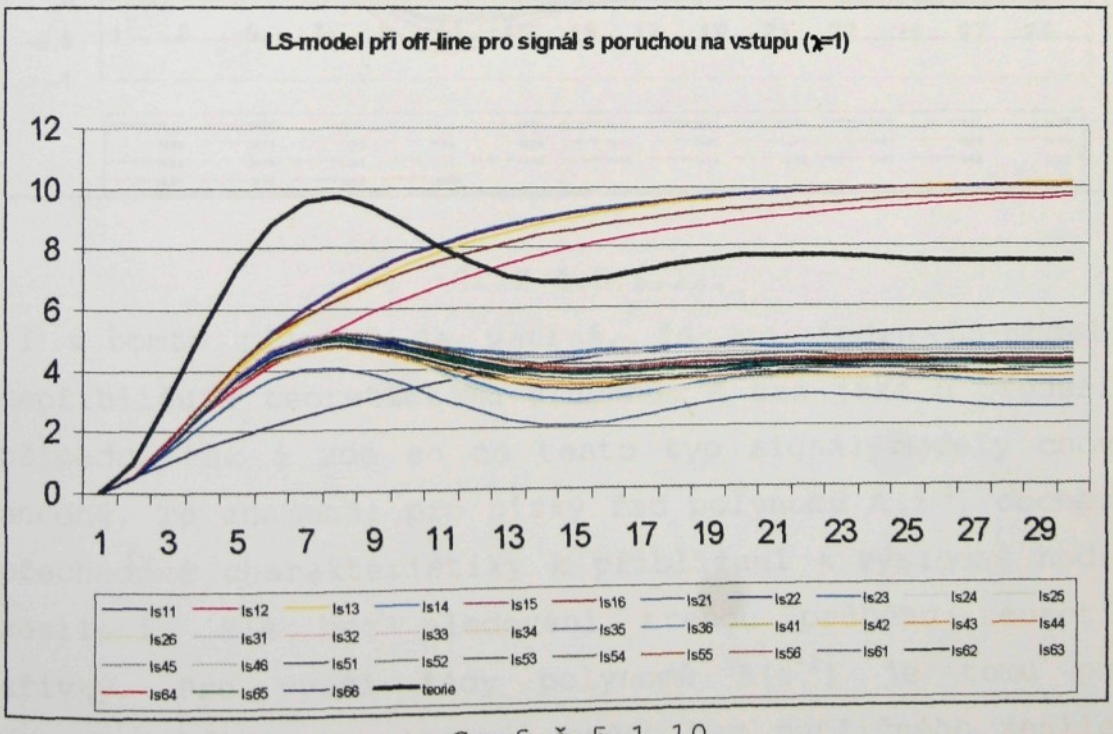
Graf č.5.1.8.

I pro tento případ vycházejí všechny čtyři modely ve svých výsledcích stejně. Proto i tento typ vstupního signálu nebude již dále mít další grafické znázornění.

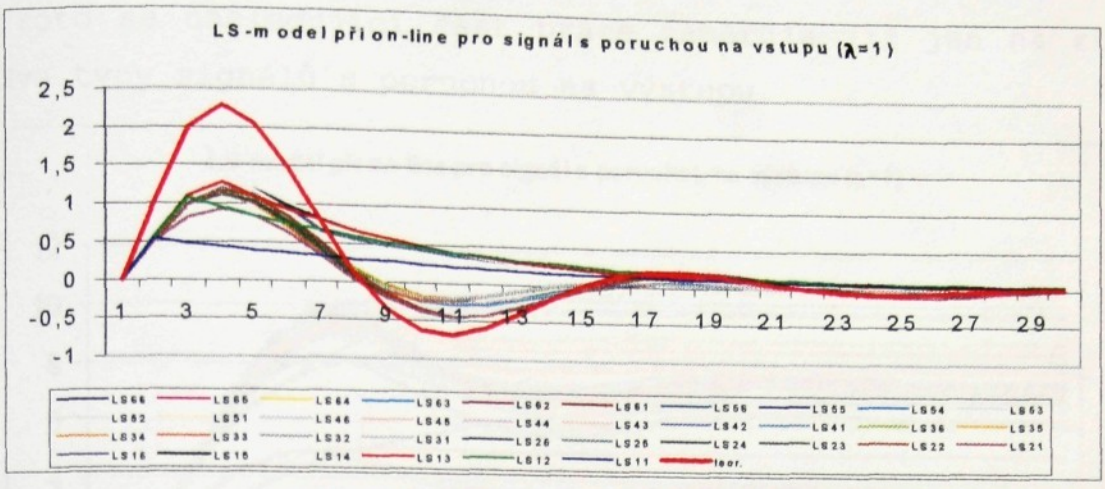
Z grafů je patrné, že si tento (a potažmo i obecný) model nedokáže poradit s velkou chybou na vstupu. Koeficienty polynomu $B(z^{-1})$ pro daný signál nabývají hodnot menších než 0,02 oproti určovanému koeficientu, který má hodnotu 1.



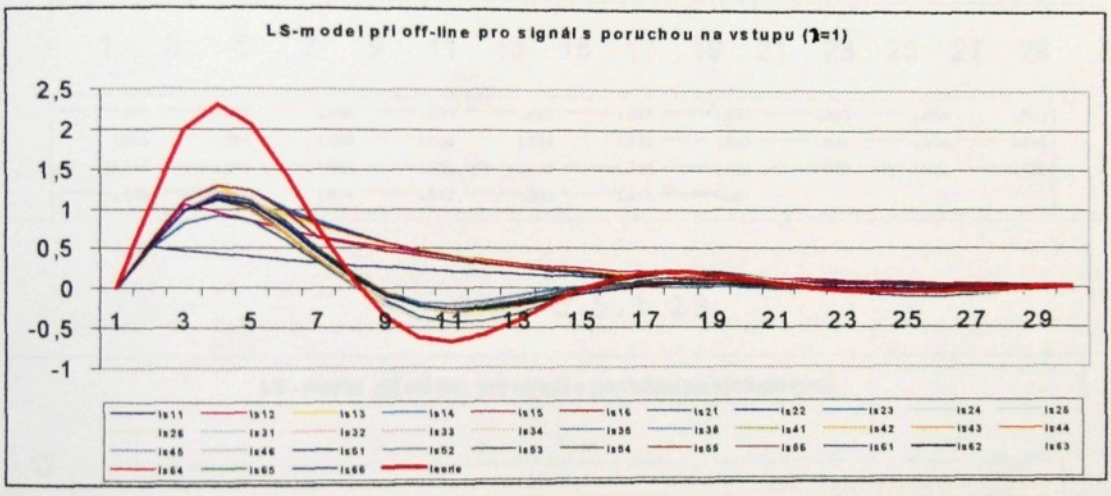
Graf č.5.1.9.



Graf č.5.1.10.



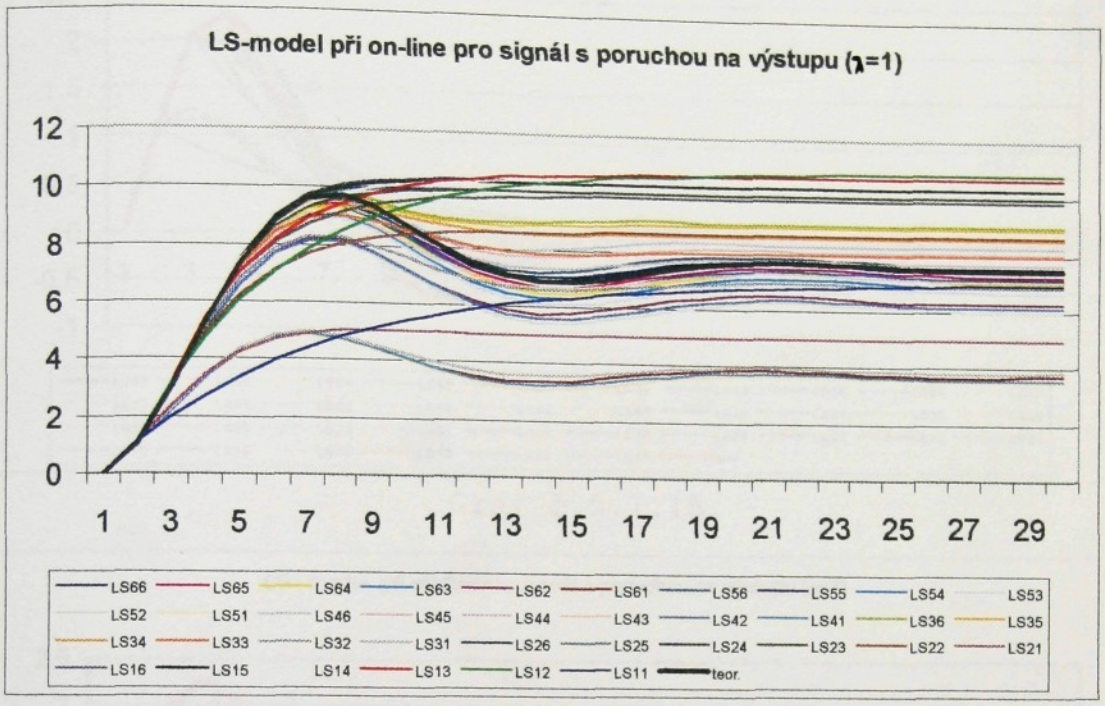
Graf č.5.1.11.



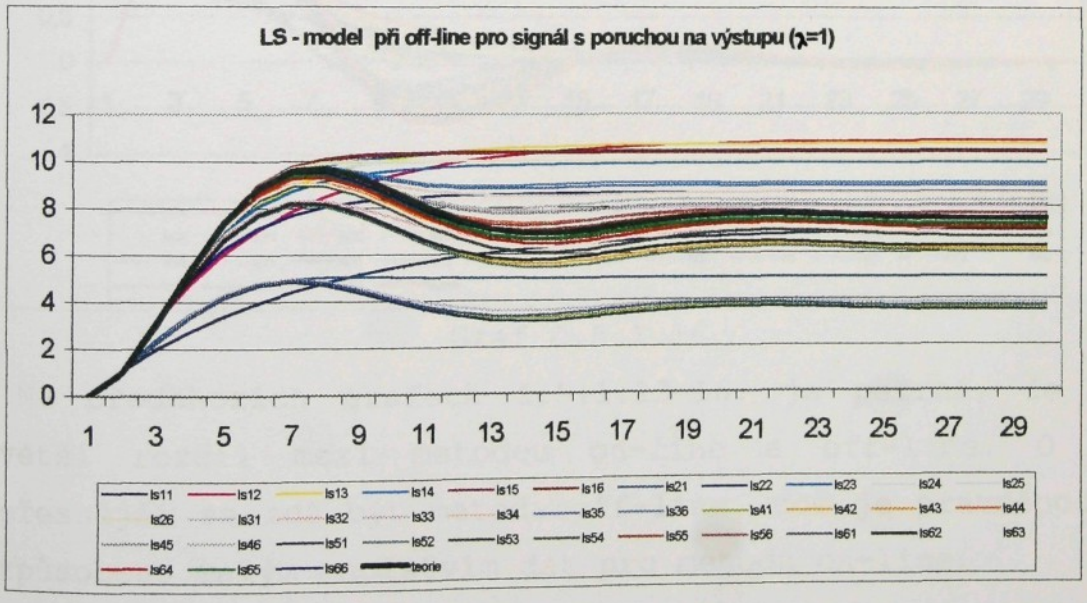
Graf č.5.1.12.

I v tomto případě je patrné, že ani jeden LS-model se nepřibližuje teoretickému průběhu. A tak jako u předešlého případu, tak i zde se na tento typ signál modely chovají shodně. To znamená: pro nízký řád polynomu $A(z^{-1})$ dochází u přechodové charakteristiky k přiblížení k výsledné hodnotě zesílení, ale bez sledování trendu průběhu teoretické křivky. Pro vyšší řády polynomů $A(z^{-1})$ je tomu právě naopak. Průběh je sledován ovšem bez patřičného zesílení.

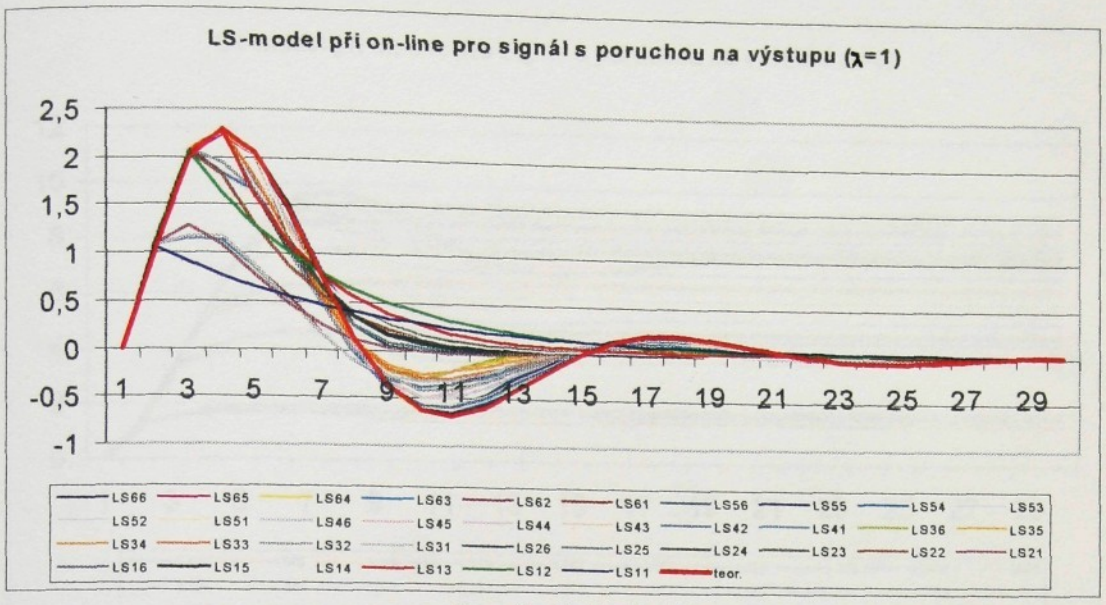
Proto se následující část práce zaměřuje již jen na zbylé dva typy signálů s poruchou na výstupu.



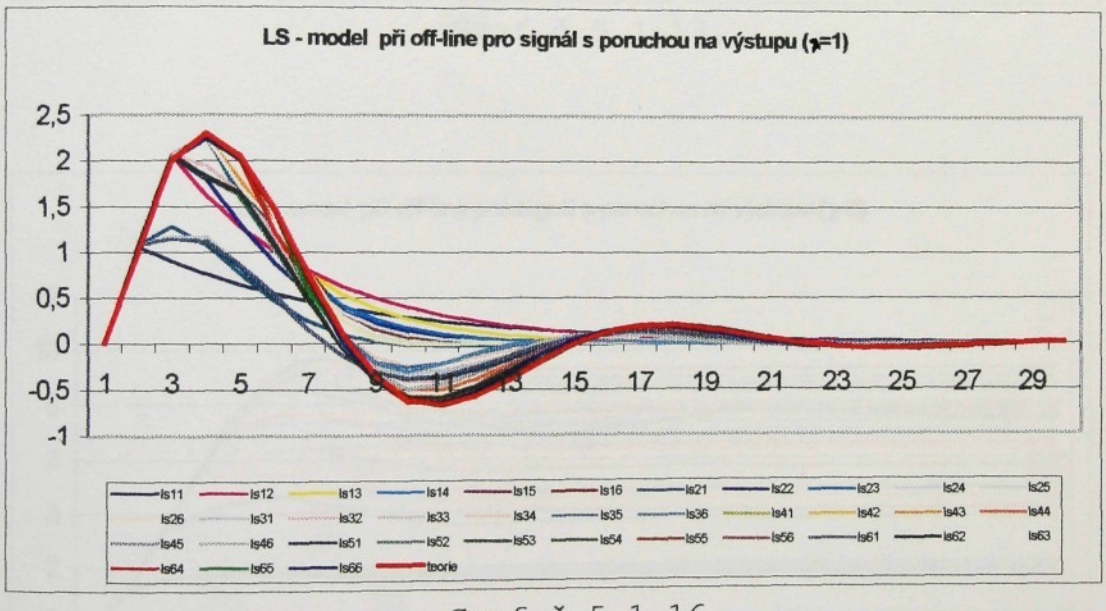
Graf č.5.1.13.



Graf č.5.1.14.



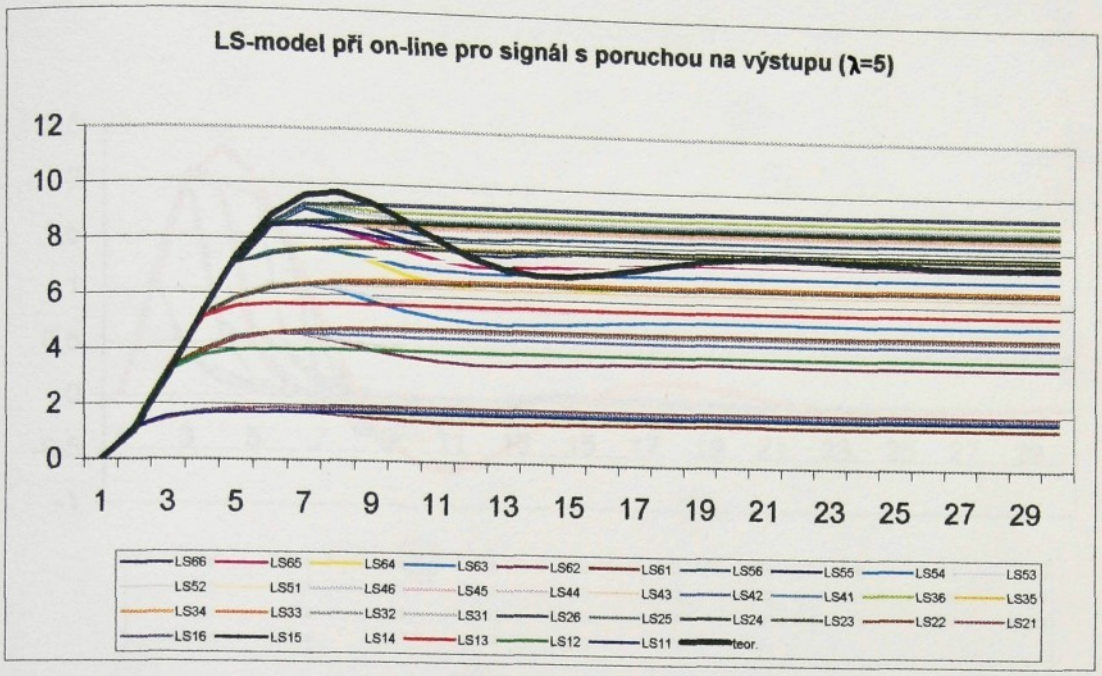
Graf č.5.1.15.



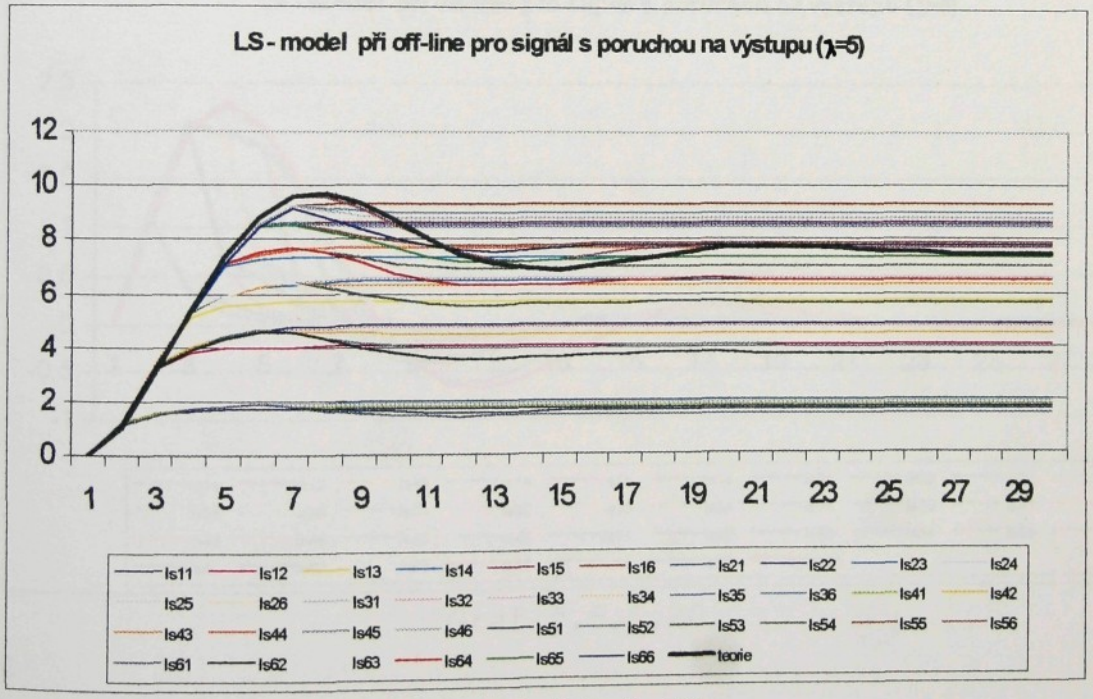
Graf č.5.1.16.

Na předchozích grafech č.5.1.13-16. je patrné, že není větší rozdíl mezi metodou on-line a off-line. O něco přesnější se zdá být metoda off-line, což je pravděpodobně způsobeno malým množstvím dat pro metodu on-line.

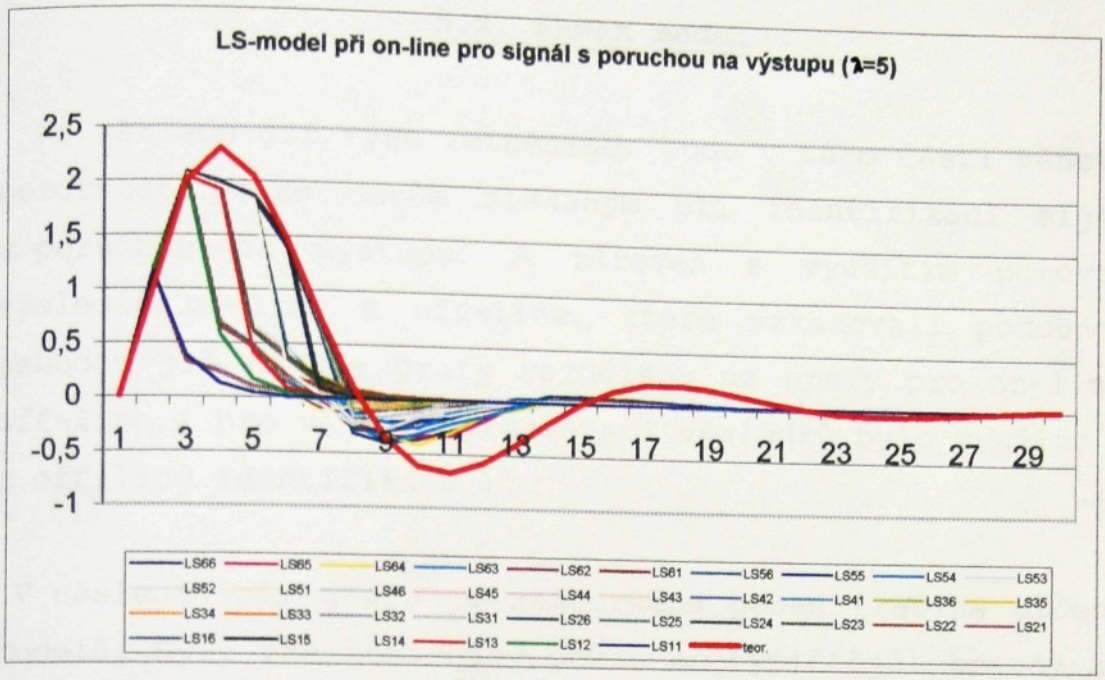
Proto bude zajímavější pozorovat rozdíly mezi jednotlivými modely než metodami.



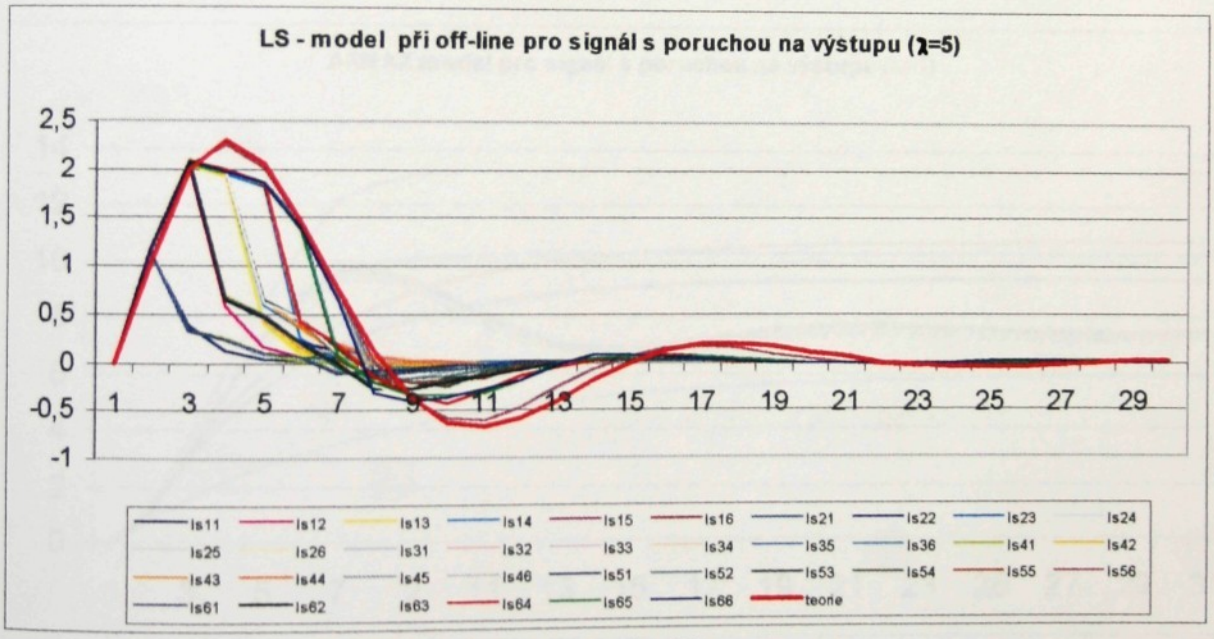
Graf č.5.1.17.



Graf č.5.1.18.



Graf č.5.1.19.



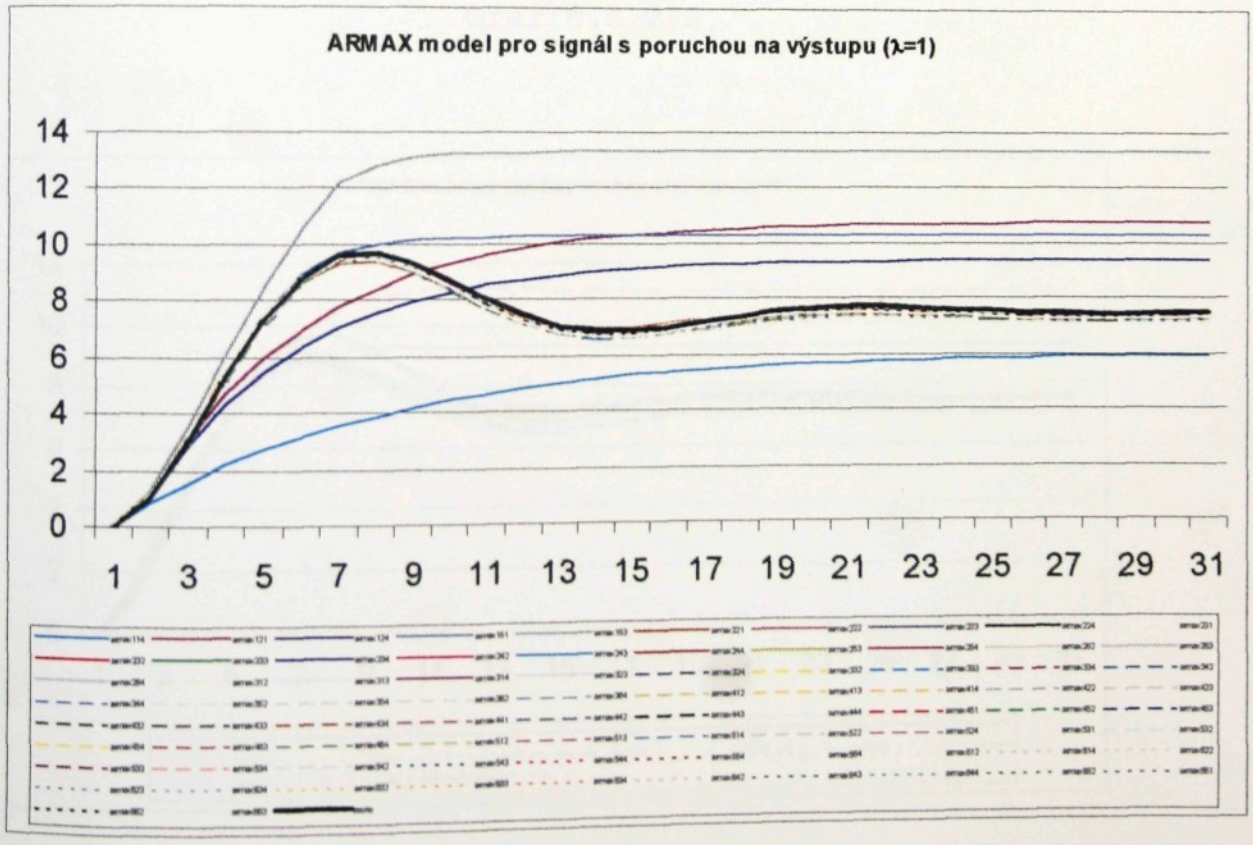
Graf č.5.1.20.

Také pro tento soubor dat platí předchozí hodnocení (viz. str. 39).

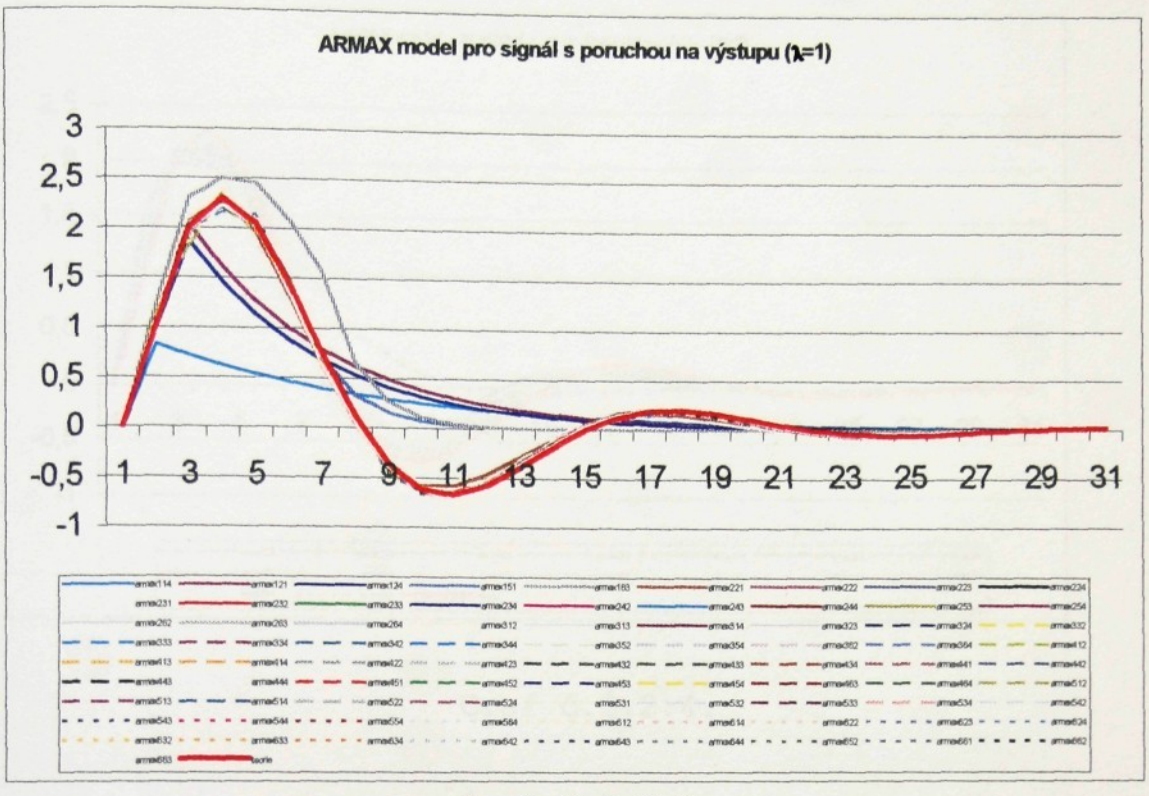
5.2. ARMAX model

Na základě již výše zmíněného, bude v této části věnována pozornost pouze datům získaným při identifikaci signálu s poruchou na výstupu. A zároveň s využitím porovnání výsledků on-line a off-line, které vzkazovali podobnost, nebudou již nadále grafy rozděleny na grafy pro on-line a off-line (pro vlastní prezentaci výsledků bylo využito dat z off-line identifikace).

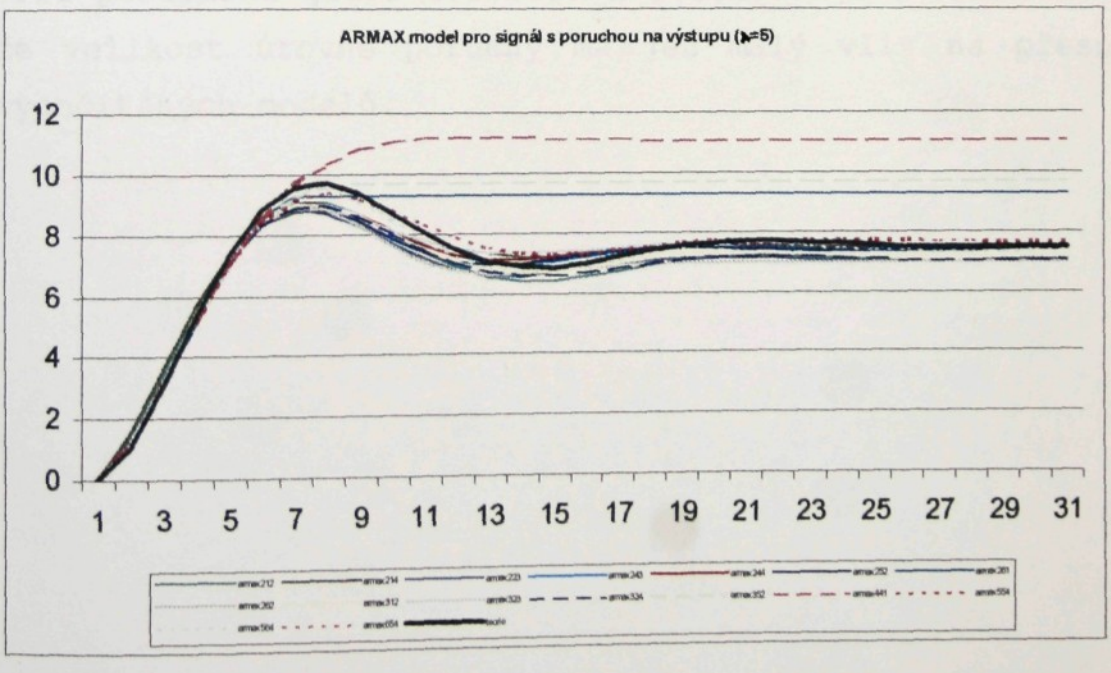
V následujícím grafu je znázorněna pouze třetina určených modelů. Byly zde ponechány pouze nejtypičtější trendy a to v poměru k jejich zastoupení.



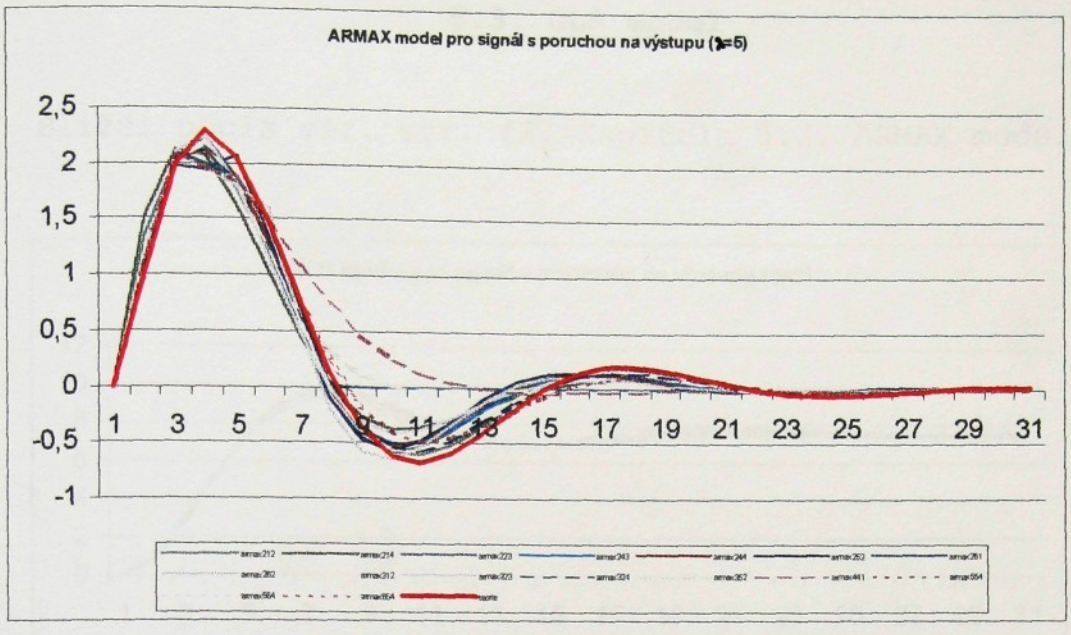
Graf č.5.2.1.



Graf č.5.2.2.



Graf č.5.2.3.



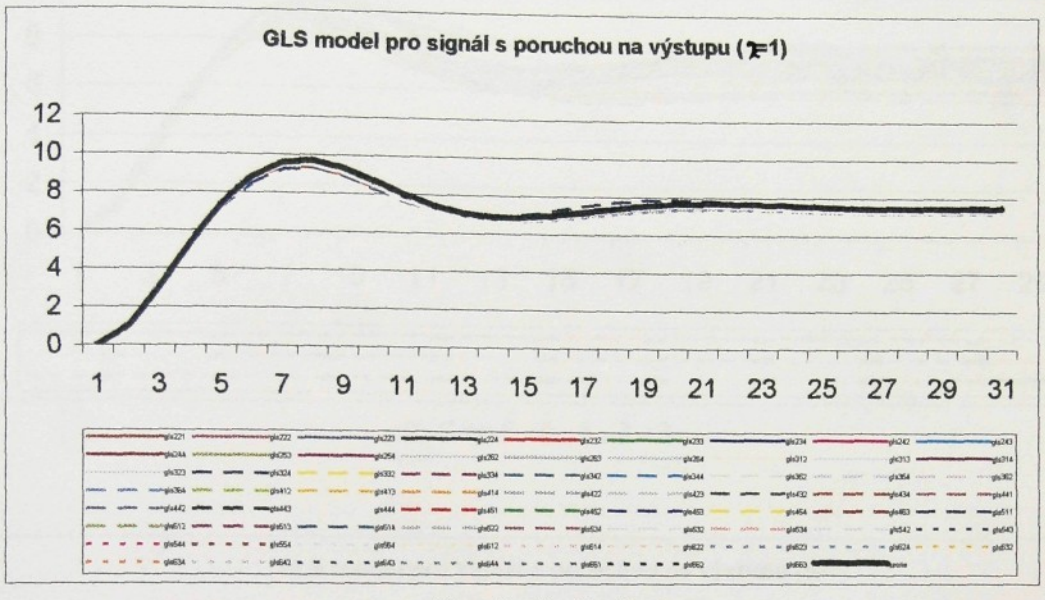
Graf č.5.2.4.

Pro grafy č.5.2.3-4. bylo zvoleno mnohem menší množství modelů a to jen z důvodů větší názornosti.

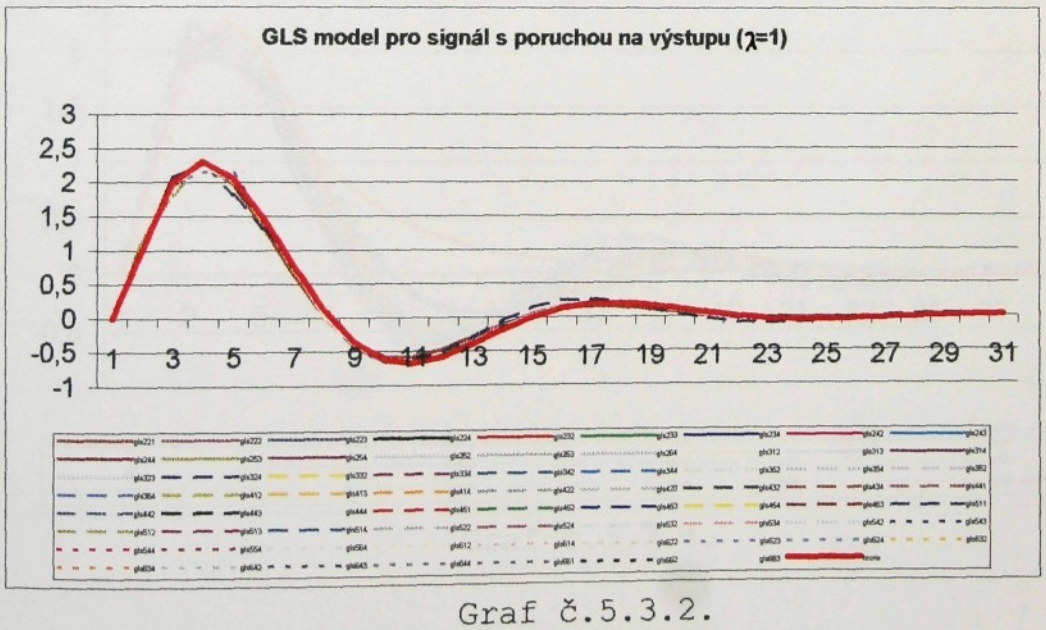
Při porovnání grafu č.5.2.1. a č.5.2.3. je názorně vidět, že velikost úrovně poruchy má jen malý vliv na přesnost vypočítaných modelů.

5.3. GLS model

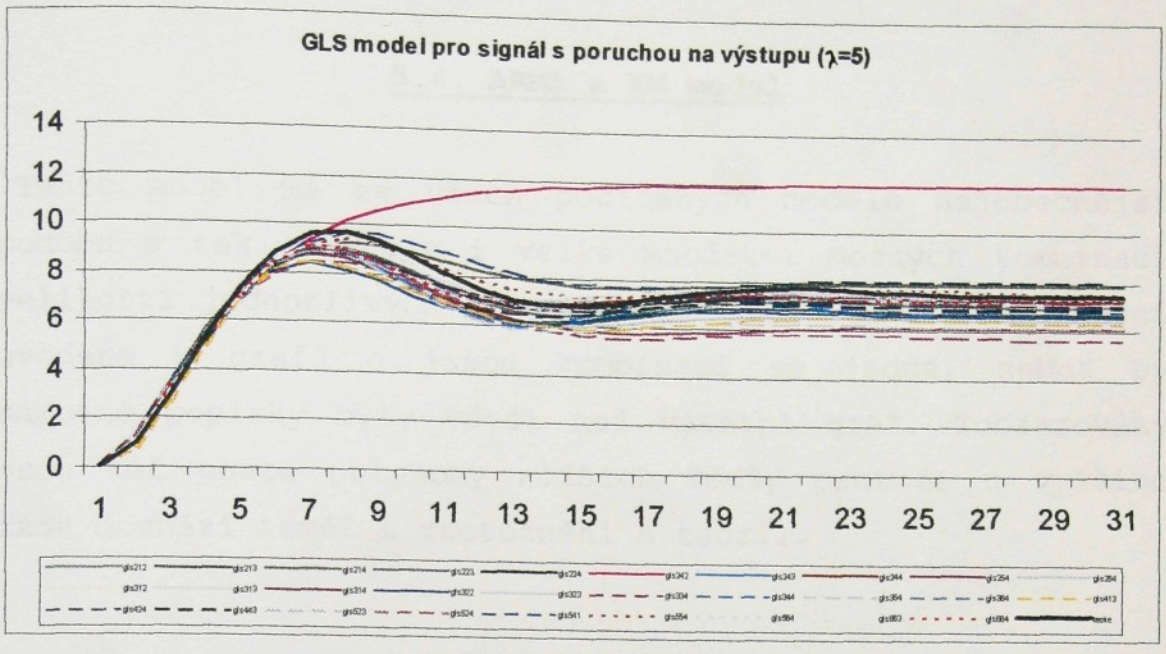
Bližší popis viz. str. 42. Kapitola 5.2. ARMAX model.



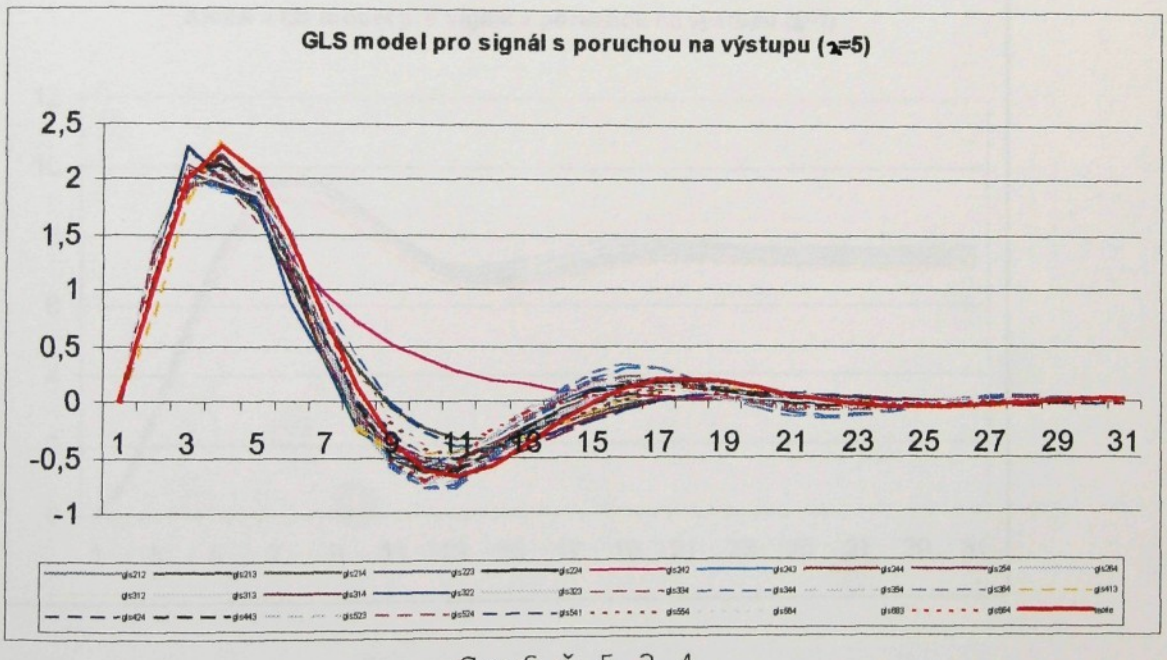
Graf č.5.3.1.



Graf č.5.3.2.



Graf č.5.3.3.

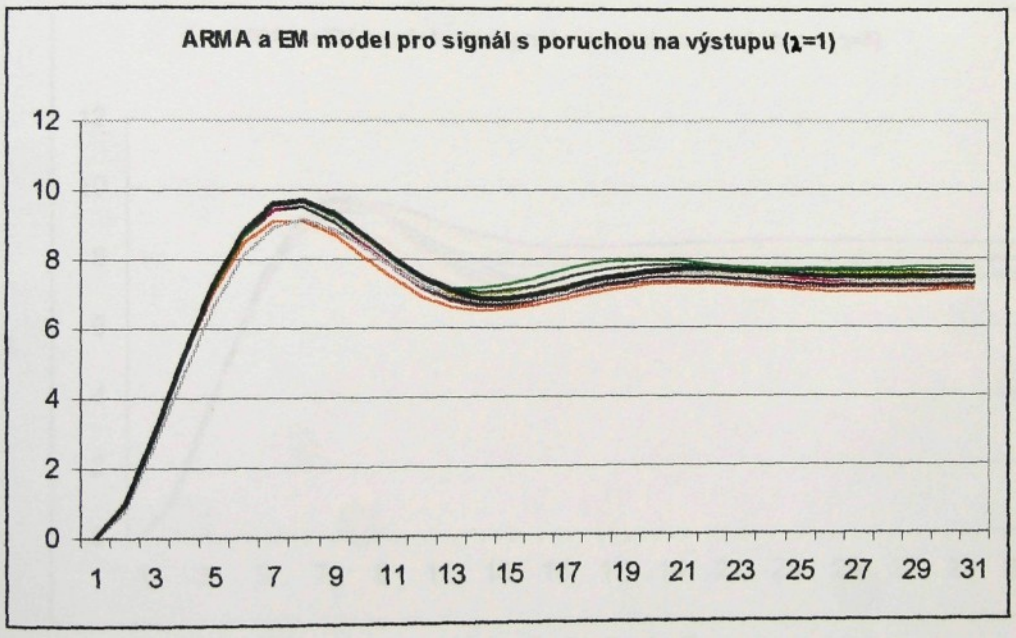


Graf č.5.3.4.

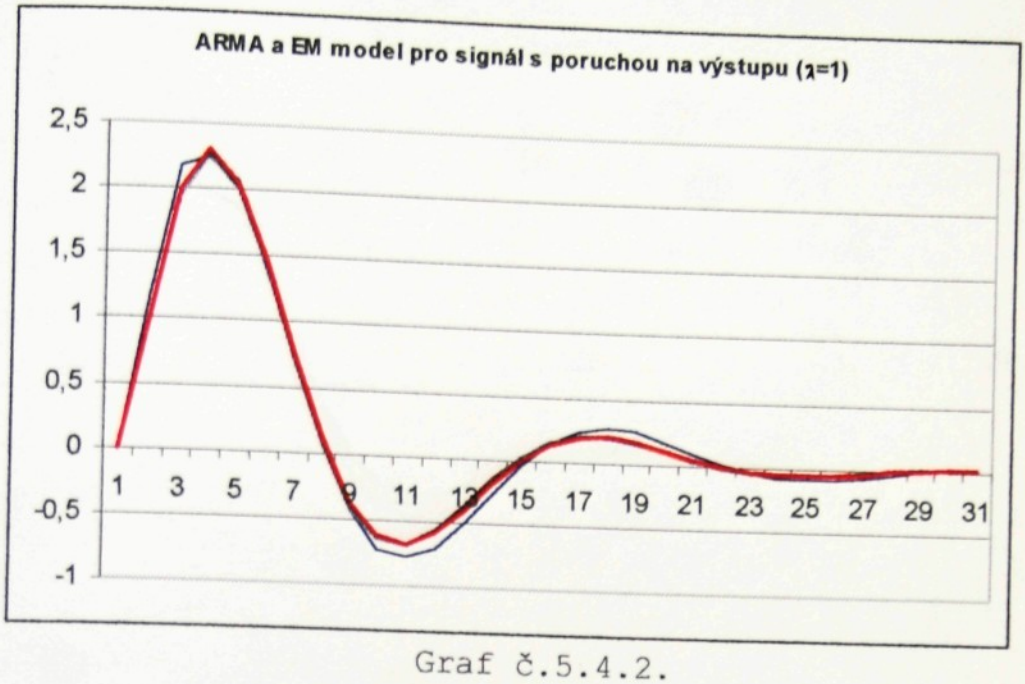
Již na první pohled je patrné, že metody GLS a ARMAX dosahují přibližně stejných výsledků.

5.4. ARMA a EM model

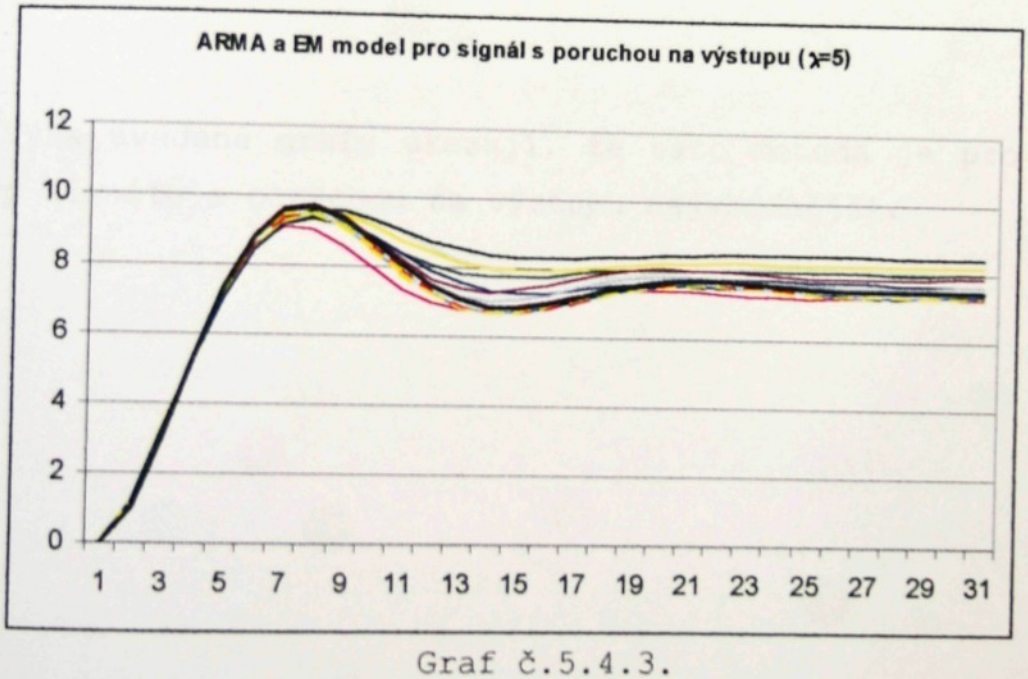
Tento model má ze všech počítaných modelů nejobecnější podobu a tak existuje i velké množství možných kombinací velikostí jednotlivých polynomů. Z tohoto důvodu také není uvedeno u grafů o jakou kombinaci se jedná, neboť by samotné popisky byly větší než vlastní graf. Zobrazovány jsou též pouze polynomy nižších řádů, protože u vyššího řádu dochází téměř k ztotožnění s teorií.



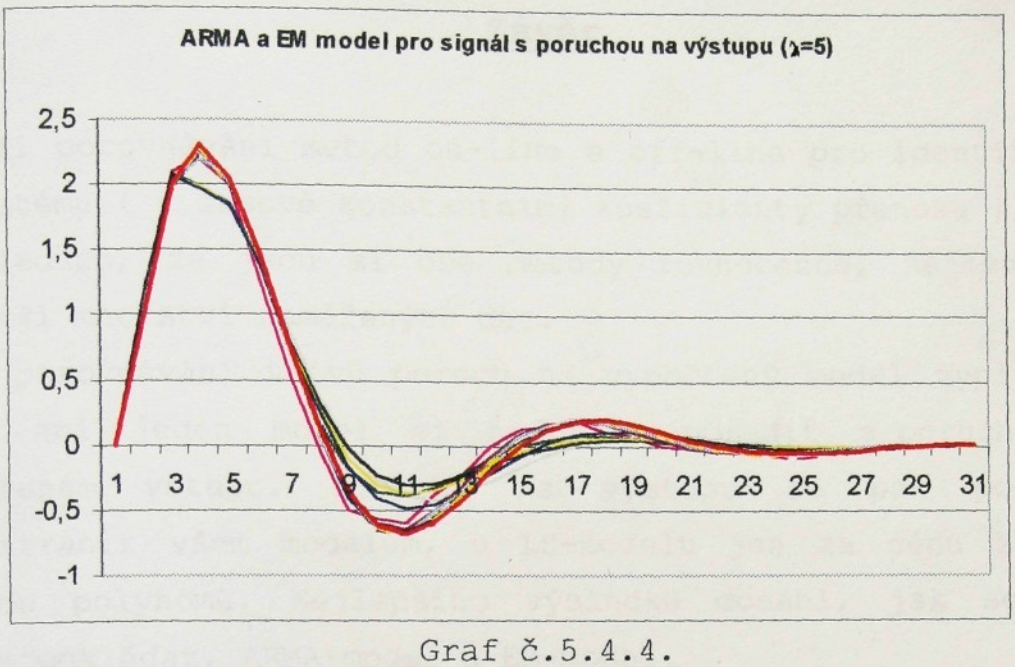
Graf č.5.4.1.



Graf č.5.4.2.



Graf č.5.4.3.



Graf č.5.4.4.

Výše uvedené grafy ukazují, že tato metoda je pro tento typ signálu s poruchou na výstupu nejvhodnější.

Závěr

Při porovnávání metod on-line a off-line pro identifikaci systému (s časově konstantními koeficienty přenosu), bylo shledáno, že jsou si obě metody rovnocenné, zejména pro větší množství naměřených dat.

Z porovnávání vlivu poruch na vypočtený model vyplynulo, že ani jeden model si nedokáže poradit s poruchou na měřeném vstupu. Poruchu na výstupu se pak podařilo odstranit všem modelům, u LS-modelu jen za cenu zvýšení řádu polynomů. Nejlepšího výsledku dosáhl, jak se dalo předpokládat, ARMA-model a EM-model.

Z výše uvedeného plyne, že při identifikaci neznámého systému v praxi je třeba dát největší pozor na možnost vzniku poruch především na měřeném vstupu, chyba na výstupu může být odfiltrována.

Pro praktické měření je třeba také co nejlépe odhadnout řády jednotlivých polynomů, neboť při špatném určení může dojít k nestabilitě určeného systému.

Seznam použité literatury:

Olehla,M.: Identifikace technologických soustav.
TU Liberec 1997

Olehla,M. - Věchet,V. - Olehla,J.: Řešení úloh matematické statistiky ve Fortranu.NADAS Praha 1982

Seznam příloh:

Příloha č. 1 : Výpis programu pro off-line identifikaci

Příloha č. 2 : Výpis programu pro on-line identifikaci

Příloha č. 3 : Dvě diskety obsahující instalační program pro program na off-line identifikaci

Příloha č. 4 : Dvě diskety obsahující instalační program pro program na on-line identifikaci

Prohlášení k využívání výsledků diplomové práce

Jsem si vědom toho, že diplomová práce je majetkem školy, a že s ní nemohu sám bez svolení školy disponovat, že kopie diplomové práce může být zapůjčena či objednána za účelem využití jejího obsahu.

Beru na vědomí , že po pěti letech si mohu diplomovou práci vyžádat v Univerzitní knihovně TU v Liberci, kde bude uložena.

27.5.1999

Michal Slánský
Jana Palacha 1217
Mladá Boleslav 293 01

Příloha č.1.

Výpis programu pro off-line identifikaci

```
' Globální definování vektoru a matic
Dim p_matice() As Double
Dim k_matice() As Double
Dim k_vektor() As Double
Dim u(), y(), e(), w(), v() As Double
Dim ypom() As Double

'   Globálně definované vektory pro nalezený systém
Dim Ua_pocet As Integer
Dim Ua() As Double
Dim Uam() As Double
Dim Ub_pocet As Integer
Dim Ub() As Double
Dim Ubm() As Double
Dim Uc_pocet As Integer
Dim Uc() As Double
Dim Ud_pocet As Integer
Dim Ud() As Double
Dim Uf_pocet As Integer
Dim Uf() As Double

Dim Pocet_Mereni As Integer
Dim Zdroj_dat As String
Dim Adresar_vysledku As String
Dim Max_iteraci As Integer

Private Sub Form_Load()
 Zdroj_dat = InputBox("Zadej cestu i s názvem souboru s daty", "Dotaz")
 Pocet_Mereni = InputBox("Zadej počet analyzovaných dat", "Dotaz")
 Adresar_vysledku = InputBox("Zadej cestu s názvem adrtesáře pro ukládaní výsledků (např. c:\data )", "Dotaz")
 Max_iteraci = InputBox("Zadej maximální počet iterací", "Dotaz")
End Sub

Sub Nacti_vst_data()
Dim i As Integer
Dim K, l As Double

 Open Zdroj_dat For Input As #1

 ReDim u(Pocet_Mereni)
 ReDim y(Pocet_Mereni)
  For i = 1 To Pocet_Mereni
    Input #1, u(i), y(i), K, l
  Next
 Close #1

End Sub

Sub Vypocet_k_matice(sirka As Integer, vyska As Integer)
Dim i, j As Integer
Dim tisk_retezec As String

 ReDim k_matice(sirka + 1, sirka)

 For i = 1 To sirka
  For j = 1 To sirka
   k_matice(i, j) = 0
   For K = 1 To vyska
    k_matice(i, j) = k_matice(i, j) + p_matice(i, K) * p_matice(j, K)
   Next
  Next
```

```
 Next

 For j = 1 To sirka
  k_matice(sirka + 1, j) = 0
  For K = 1 To vyska
   k_matice(sirka + 1, j) = k_matice(sirka + 1, j) + y(K) * p_matice(j, K)
  Next
 Next

 Call Reseni_k_matice(sirka + 1)

 End Sub

 Sub Reseni_k_matice(n As Integer)
  Dim i, j, K As Integer
  Dim pom() As Double
  Dim podil As Double
  Dim tisk_retezec, pod As String

  ReDim pom(n, n - 1)
  ReDim k_vektor(n - 1)

  For i = 1 To n
   For j = 1 To n - 1
    pom(i, j) = k_matice(i, j)
   Next
  Next

 ' výpocet trojúhelníkové matice
  For i = 1 To n - 2
   For K = i To n - 2
    podil = pom(i, K + 1)
    For j = i To n
     If pom(i, i) <> 0 Then
      pom(j, K + 1) = pom(j, K + 1) - pom(j, i) * podil / pom(i, i)
     End If
    Next
   Next
  Next

  ' zpetná redukce na diagonální vysledny vektor
  For i = 1 To n - 1 Step 1
   podil = pom(n, n - i)
   For j = 1 To i - 1
    podil = podil - pom(n - j, n - i) * k_vektor(n - j)
   Next
   If pom(n - i, n - i) <> 0 Then
    k_vektor(n - i) = podil / pom(n - i, n - i)
   Else
    k_vektor(n - i) = 0
   End If
  Next

 End Sub

 Sub LS_model(Pocet As Integer, pa As Integer, pb As Integer)
 Dim i, j As Integer
 Dim pom As Double
 Dim tisk_retezec As String

  Ua_pocet = pa
  Ub_pocet = pb
  ReDim Ua(Ua_pocet)
  ReDim Ub(Ub_pocet)
  ReDim Uam(Ua_pocet)
  ReDim Ubm(Ub_pocet)
  ReDim p_matice(Ua_pocet + Ub_pocet, Pocet)
```

```
For i = 1 To Ub_pocet Step 1
 For j = 1 To Pocet
  If j - i + 1 > 0 Then
   p_matice(Ub_pocet - i + 1, j) = u(j - i + 1)
  Else
   p_matice(Ub_pocet - i + 1, j) = 0
  End If
 Next
Next
For i = Ub_pocet + 1 To Ua_pocet + Ub_pocet Step 1
 For j = 1 To Pocet
  If j - i + 1 + Ub_pocet > 0 Then
   p_matice(2 * Ub_pocet + Ua_pocet - i + 1, j) = -y(j - i + Ub_pocet)
  Else
   p_matice(2 * Ub_pocet + Ua_pocet - i + 1, j) = 0
  End If
 Next
Next

Call Vypocet_k_matice(Ua_pocet + Ub_pocet, Pocet)

' vlození výsledku do vektoru
For i = Ub_pocet To 1 Step -1
 Ub(Ub_pocet - i) = k_vektor(i)
Next
 Ua(1) = 1
For i = Ub_pocet + Ua_pocet + 1 To Ub_pocet + 2 Step -1
 Ua(Ua_pocet + Ub_pocet - i + 2) = k_vektor(i - 1)
Next

End Sub

Sub vyp_ypom_e(a1 As Integer, b1 As Integer, c1 As Integer, d1 As Integer)
Dim i, j As Integer
ReDim ypom(Pocet_Mereni)

  For i = 1 To Pocet_Mereni
  ypom(i) = 0
   For j = 1 To a1
    If i - j > 0 Then
     ypom(i) = ypom(i) - Ua(j) * y(i - j)
    Else
     ypom(i) = ypom(i) - Ua(j) * 0
    End If
   Next
   ypom(i) = ypom(i) + Ub(0) * u(i)
   For j = 1 To b1 - 1
    If i - j > 0 Then
     ypom(i) = ypom(i) + Ub(j) * u(i - j)
    Else
     ypom(i) = ypom(i) + Ub(j) * 0
    End If
   Next
   For j = 1 To c1
    If i - j > 0 Then
     ypom(i) = ypom(i) + Uc(j) * e(i - j)
    Else
     ypom(i) = ypom(i) + Uc(j) * 0
    End If
   Next
   For j = 1 To d1
    If i - j > 0 Then
     ypom(i) = ypom(i) - Ud(j) * v(i - j)
    Else
     ypom(i) = ypom(i) - Ud(j) * 0
    End If
```

```
    Next
   Next
ReDim e(Pocet_Mereni)

 For i = 1 To Pocet_Mereni

  e(i) = CDbl(y(i)) - CDbl(ypom(i))
 Next

End Sub

Sub vyp_v(a1 As Integer, b1 As Integer)
 Dim i, j As Integer
 ReDim v(Pocet_Mereni)

 For i = 1 To Pocet_Mereni
  v(i) = y(i)
  For j = 1 To a1
   If i - j > 0 Then
    v(i) = v(i) + Ua(j) * y(i - j)
   End If
  Next
  v(i) = v(i) - Ub(0) * u(i)
  For j = 1 To b1 - 1
   If i - j > 0 Then
    v(i) = v(i) - Ub(j) * u(i - j)
   End If
  Next
 Next

End Sub

Sub Pridavne_vypocty(a As Integer, b As Integer, c As Integer, d As Integer)
 Call vyp_v(a, b)
 Call vyp_ypom_e(a, b, c, d)
End Sub

Sub uprava_u_y(C_velikost As Integer, D_velikost As Integer)
Dim u2() As Double
Dim y2() As Double

ReDim u2(Pocet_Mereni)
ReDim y2(Pocet_Mereni)

For i = 1 To Pocet_Mereni
 u2(i) = u(i)
 y2(i) = y(i)
  For K = 1 To D_velikost
   If i - K > 0 And D_velikost > 0 Then
    u2(i) = u2(i) + Ud(K) * u(i - K)
    y2(i) = y2(i) + Ud(K) * y(i - K)
   End If
  Next
Next
For i = 1 To Pocet_Mereni
 For K = 1 To C_velikost
  If i - K > 0 And C_velikost > 0 Then
   u2(i) = u2(i) - Uc(K) * u(i - K)
   y2(i) = y2(i) - Uc(K) * y(i - K)
  End If
 Next
Next

 For i = 1 To Pocet_Mereni
  u(i) = u2(i)
  y(i) = y2(i)
 Next
```

```
End Sub

Private Sub SRun_Click()
Dim i, j, K As Integer
Dim sss As String
Dim Ia As Integer
Dim Ib As Integer
Dim Ic As Integer
Dim Id As Integer
Dim sdn, sstr As String
Dim podminka As Boolean
Dim iterace As Integer
Dim se, semin As Double
Dim Ubbest() As Double
Dim Uabest() As Double
Dim coun As Integer
Dim Ia_dolni, Ia_horni As Integer
Dim Ib_dolni, Ib_horni As Integer
Dim Ic_dolni, Ic_horni As Integer
Dim Id_dolni, Id_horni As Integer
Dim coun_max As Integer
 Ia_dolni = 1
 Ia_horni = 6
 Ib_dolni = 1
 Ib_horni = 6
 Ic_dolni = 0
 Ic_horni = 4
 Id_dolni = 0
 Id_horni = 4

 coun_max = (Ia_horni - Ia_dolni + 1) * (Ib_horni - Ib_dolni + 1) * (Id_horni - Id_dolni + 1) * (Ic_horni - Ic_dolni + 1)
 coun = coun_max
 T2.Text = coun_max & " / " & coun_max
 T2.Refresh

 For Ia = Ia_horni To Ia_dolni Step -1     '6-1
 For Ib = Ib_horni To Ib_dolni Step -1     '6-1

   sdn = Adresar_vysledku & "\OFF_vysl" & Ia & "_" & Ib & ".dat"
   Open sdn For Output As #10

   Call Nacti_vst_data
   Call LS_model(Pocet_Mereni, Ia, Ib)
   ReDim Uabest(Pocet_Mereni)
   ReDim Ubbest(Pocet_Mereni)

 For Ic = Ic_dolni To Ic_horni    '0-6
 For Id = Id_dolni To Id_horni    '0-6

    ReDim Ud(Id)
    ReDim Uc(Ic)

Ta.Text = Ia
Ta.Refresh
Tb.Text = Ib
Tb.Refresh
Tc.Text = Ic
Tc.Refresh
Td.Text = Id
Td.Refresh
Ti.Text = iterace
Ti.Refresh

   Call Nacti_vst_data
   Call LS_model(Pocet_Mereni, Ia, Ib)
   ReDim p_matice(Ia + Ib + Ic + Id, Pocet_Mereni)
```

```
 podminka = True
 iterace = 0
 Bar = 0

While (podminka And Not (iterace > Max_iteraci))

 Call Pridavne_vypocty(Ia, Ib, Ic, Id)
 iterace = iterace + 1

 Ti.Text = iterace
 Ti.Refresh

 For i = 1 To Ib
  For j = 1 To Pocet_Mereni
    If j - i + 1 > 0 Then
      p_matice(i, j) = u(j - i + 1)
    Else
     p_matice(i, j) = 0
   End If
  Next
 Next

 For i = 1 To Ia
  For j = 1 To Pocet_Mereni
   If j - i > 0 Then
    p_matice(Ib + i, j) = -y(j - i)
   Else
    p_matice(Ib + i, j) = 0
   End If
  Next
 Next

 For j = 1 To Id
  For K = 1 To Pocet_Mereni
   If K - j > 0 Then
    p_matice(Ia + Ib + j, K) = -v(K - j)
   Else
    p_matice(Ia + Ib + j, K) = 0
   End If
  Next
 Next

 For j = 1 To Ic
  For K = 1 To Pocet_Mereni
   If K - j > 0 Then
    p_matice(Ia + Ib + Id + j, K) = e(K - j)
   Else
    p_matice(Ia + Ib + Id + j, K) = 0
   End If
  Next
 Next

 Call Vypocet_k_matice(Ia + Ib + Ic + Id, Pocet_Mereni)

 For K = 1 To 6
  If K > Ib Then

  Else
   Ubm(K - 1) = Ub(K - 1)
   Ub(K - 1) = k_vektor(K)
  End If
 Next

 For K = 1 To 6
  If K > Ia Then

  Else
```

```
      Uam(K) = Ua(K)
      Ua(K) = k_vektor(K + Ib)
     End If
    Next

    For K = 1 To 6
     If K > Id Then

     Else
      Ud(K) = k_vektor(K + Ib + Ia)
     End If
    Next

    For K = 1 To 6
     If K > Ic Then

     Else
      Uc(K) = k_vektor(K + Ib + Ia + Id)
     End If
    Next

  podminka = True
  For K = 1 To Ia
   podminka = podminka And (Ua(K) - Uam(K) < 0.001)
  Next
  For K = 1 To Ib
   podminka = podminka And (Ub(K - 1) - Ubm(K - 1) < 0.001)
  Next
  podminka = Not podminka

  se = 0
  For i = 1 To Pocet_Mereni
   se = se + e(i) * e(i)
  Next
  If semin > se Or iterace = 1 Then
   semin = se
   For K = 1 To 6
     If K > Ib Then
     Else
      Ubbest(K - 1) = k_vektor(K)
     End If
    Next
    For K = 1 To 6
     If K > Ia Then
     Else
      Uabest(K) = k_vektor(K + Ib)
     End If
    Next
  End If
  Call uprava_u_y(Ic, Id)
  Bar = CInt(100 * (iterace / (Max_iteraci + 1)))

 Wend ' Konec iterace

 Bar = 100

 coun = coun - 1
 T2.Text = coun & "  / " & coun_max
 T2.Refresh

   Print #10, "b-est>"
   For K = 1 To 6
    If K > Ib Then
     Print #10, " "
    Else
     If iterace > Max_iteraci Then Ub(K - 1) = Ubbest(K - 1)
     Print #10, Ub(K - 1)
```

```
    End If
  Next

  Print #10, "a-est>"
  For K = 1 To 6
    If K > Ia Then
     Print #10, " "
    Else
     If iterace > Max_iteraci Then Ua(K) = Uabest(K)
     Print #10, Ua(K)
    End If
  Next

  Print #10, "d>"
  For K = 1 To 6
    If K > Id Then
     Print #10, " "
    Else
     Print #10, k_vektor(K + Ib + Ia)
    End If
  Next

  Print #10, "c>"
  For K = 1 To 6
    If K > Ic Then
     Print #10, " "
    Else
     Print #10, k_vektor(K + Ib + Ia + Id)
    End If
  Next

  Print #10, "Pocet iterace" & ">" & iterace

Next
Next

 Close #10
Next
Next

 If MsgBox("KONEC  ?", vbOKCancel, "Dotaz") = vbOK Then End
End Sub
```

Příloha č.2.

Výpis programu pro on-line identifikaci

```
Dim row(40) As Double
Dim b(30, 30), a(30, 30) As Double
Dim kr, krc, kn As Integer

Dim Pocet_Mereni As Integer
Dim Zdroj_dat As String
Dim Adresar_vysledku As String
Dim Krok_vypisu As Integer

Private Sub Form_Load()
 Zdroj_dat = InputBox("Zadej cestu i s názvem souboru s daty", "Dotaz")
 Pocet_Mereni = InputBox("Zadej pocet analyzovaných dat", "Dotaz")
 Krok_vypisu = InputBox("Zadej pocet analyzovaných dat, po kterých bude následovat zápis do souboru.",
"Dotaz")
 Adresar_vysledku = InputBox("Zadej cestu s názvem adrtesáre pro ukládaní výsledků (např. c:\data )", "Dotaz")
End Sub

Sub Reduce(n As Integer, k As Integer, init As Integer)
 Dim i, j As Integer
 Dim c, s, d As Double

 ' algoritmus reduce
 If init = 0 Then
  kr = 0
  krc = 0
  For i = 1 To n + 1
   For j = 1 To n + 1
    a(i, j) = 0
   Next
  Next
 End If

 For i = 1 To n
  If Abs(row(i)) > 0.000001 Then
    d = Sqr(row(i) * row(i) + a(i, i) * a(i, i))
    c = a(i, i) / d
    s = row(i) / d
    For j = i To n + 1
     d = c * row(j) - s * a(i, j)
     a(i, j) = s * row(j) + c * a(i, j)
     row(j) = d
    Next
  End If
 Next

 krc = krc + 1
 kr = kr + 1

 For i = 1 To n
  For j = 1 To n + 1
   b(i, j) = a(i, j)
  Next
 Next

 i = n
 While (i >= 1)
  If b(i, i) = 0 Then
   b(i, i) = 0.000001
  End If
  row(i) = b(i, n + 1) / b(i, i)
  j = i - 1
```

```
 While (j >= 1)
  b(j, n + 1) = b(j, n + 1) - b(j, i) * row(i)
  j = j - 1
 Wend
 i = i - 1
Wend

End Sub

Private Sub SRun_Click()
Dim nt, mt, mz, mr, mx, my, n, i, j As Integer
Dim k As Integer
Dim init As Integer
Dim mt1, mt2, mt3, mtz, mtzr, mtzrx, mtzrxy As Integer
Dim ii, mm, ns, mtz1, mtzr1, mtzrx1, mtzrxy1 As Integer
Dim nc, kr As Integer
Dim r, z As Double
Dim y(50) As Double

Dim ss As String

Dim Ia, Ib, Ic, Id As Integer
Dim pom1, pom2 As Double
Dim coun As Integer
Dim Ia_dolni, Ia_horni As Integer
Dim Ib_dolni, Ib_horni As Integer
Dim Ic_dolni, Ic_horni As Integer
Dim Id_dolni, Id_horni As Integer
Dim coun_max As Integer
 Ia_dolni = 1
 Ia_horni = 6
 Ib_dolni = 1
 Ib_horni = 6
 Ic_dolni = 0
 Ic_horni = 4
 Id_dolni = 0
 Id_horni = 4

 coun_max = (Ia_horni - Ia_dolni + 1) * (Ib_horni - Ib_dolni + 1) * (Id_horni - Id_dolni + 1) * (Ic_horni - Ic_dolni
+ 1)
 coun = coun_max
 T2.Text = coun_max & " / " & coun_max
 T2.Refresh

 For Ia = Ia_horni To Ia_dolni Step -1     '6-1
 For Ib = Ib_horni To Ib_dolni Step -1     '6-1

   sdn = Adresar_vysledku & "\ON_vysl" & Ia & "_" & Ib & ".dat"
   Open sdn For Output As #10

 For Ic = Ic_dolni To Ic_horni    '0-6
 For Id = Id_dolni To Id_horni    '0-6


Ta.Text = Ia
Ta.Refresh
Tb.Text = Ib
Tb.Refresh
Tc.Text = Ic
Tc.Refresh
Td.Text = Id
Td.Refresh

   nc = 0
   kr = 0
   Open Zdroj_dat For Input As #1
   mt = 0
```

```
mz = Ic
mr = Id
mx = Ib
my = Ia
n = Pocet_Mereni
k = Krok_vypisu

nt = 0
init = 0
mt1 = mt + 1
mt2 = mt - 1
mt3 = mt - 2
mtz = mt + mz
mtzr = mtz + mr
mtzrx = mtzr + mx
mtzrxy = mtzrx + my
mtz1 = mtz + 1
mtzr1 = mtzr + 1
mtzrx1 = mtzrx + 1
mtzrxy1 = mtzrxy + 1
For i = 1 To mtzrxy1
 y(i) = 0
Next
ns = 0

Do

 Tk.Text = ns
 Tk.Refresh

 Input #1, y(mtzrx), y(mtzrxy1), pom1, pom2

 ns = ns + 1
 For i = 1 To mtzrxy1
  row(i) = y(i)
 Next
 Call Reduce(mtzrxy, k, init)
 init = 1
 nt = nt + 1

 If nt = k Then
  ' vypis vysledku
  nt = 0
  Print #10, ns
  For i = 1 To mt
   Print #10, row(i)
  Next
  Print #10, "C>"
  For i = mt1 To mtz
   Print #10, row(i)
  Next
  Print #10, "D>"
  For i = mtz1 To mtzr
   Print #10, row(i)
  Next
  Print #10, "B>"
  For i = mtzr1 To mtzrx
   Print #10, row(i)
  Next
  Print #10, "A>"
  For i = mtzrx1 To mtzrxy
   Print #10, -row(i)
  Next
  Print #10, 1
  Print #10, " "

 End If
```

```
    'vypocet r
    r = 0
    For i = mtzr1 To mtzrxy
     r = r + y(i) * row(i)
    Next
    For i = 1 To mt
     r = r + y(i) * row(i)
    Next
    r = r - y(mtzrxy1)

    'vypocet z
    z = 0
    For i = mt1 To mtzr
     z = z + y(i) * row(i)
    Next
    z = z - r

    'posunuti
    For i = mt1 To mtzrxy
     y(i) = y(i + 1)
    Next
    If mr > 0 Then
     y(mtzr) = r
    End If
    If mz > 0 Then
     y(mtz) = z
    End If

   Loop Until (ns >= n)
   Close #1

 coun = coun - 1
 T2.Text = coun & "  / " & coun_max
 T2.Refresh

Next
Next

 Close #10
Next
Next

 If MsgBox("KONEC  ?", vbOKCancel, "Dotaz") = vbOK Then End
End Sub
```