# UŽIVATELSKÝ NÁVOD PROGRAMU *DYNAST*

Doc. Ing. Heřman Mann, DrSc.

# 1. Popis programu

## 1.1. Účel programu

Program DYNAST (DYNamika A STatistika) slouží k simulaci a ke statické, časové (dynamické i kinematické) a kmitočtové analýze lineárních i nelineárních dynamických soustav i k analýze jejich závislostí na různých parametrech.

Dynamické soustavy mohou být popsány nejen obyčejnými algebro-diferenciálními rovnicemi s podmíněnými a logickými výrazy, ale i blokovými nebo branovými schématy či jejich kombinacemi s rovnicemi. V případě schémat si program odpovídající rovnice sformuluje automaticky sám.

DYNAST může být využíván i jako univerzální solver (řešič) pro řešení soustav lineárních i nelineárních algebraických, obyčejných diferenciálních i algebro-diferenciálních rovnic, jako program pro řešení zobecněné úlohy vlastních čísel matic, pro Fourierovu analýzu periodických funkcí, pro vyhodnocování a zobrazování funkcí, pro výpočet derivací a integrálů funkcí, apod.

## 1.2. Vstup programu

Popis rovnic i schémat se do programu DYNAST zadává formou vstupních údajů podle pravidel vstupního jazyka tohoto programu, který je popsán v tomto návodu.

Při simulaci a analýze dynamických soustav na číslicových počítačích se s rostoucími nároky na přesnost výpočtů a rozsáhlost soustav používají stále složitější modely prvků těchto soustav. Současně se rychle rozrůstají požadavky na šíři sortimentu těchto modelů. To má ovšem za následek, že žádný simulační program nemůže být tak univerzální, aby obsahoval všechny modely, které vyžaduje každodenní praxe.

Program DYNAST se s tímto problémem vyrovnává tak, že v sobě nemá zabudovány žádné modely. Jeho vstupní jazyk je však natolik pružný, že dovoluje jednoduchým způsobem charakterizovat většinu dnes známých a užívaných modelů, ať již elektronických, regulačních, mechanických, hydraulických, pneumatických, tepelných či jiných konstrukčních prvků.

Jelikož výpočetní metody, na kterých je program DYNAST založen, vychází z iterativní linearizace nelineárních modelů, uživatel by vedle jmenovitých charakteristik musel vstupním jazykem zadávat i jejich derivace podle všech proměnných veličin, které představují prvky Jacobiho matice vektorové funkce popisující danou soustavu. Ruční výpočet těchto derivací však bývá v praxi poměrně pracný a navíc je zdrojem častých omylů. Podobně je tomu při odvozování citlivostních modelů pro citlivostní analýzu soustav.

Program DYNAST proto provádí výpočet derivací zadávaných nelineárních i podmíněných funkcí vzhledem ke všem proměnným veličinám automaticky, bez nutnosti zásahu uživatele. Výsledné derivace jsou v symbolicko-číselném tvaru.

V konstrukčních schématech se často některé konstrukční prvky vícekrát opakují. Aby uživatel programu modely takovýchto prvků nemusel v zadání úlohy vstupním jazykem popisovat vždy znovu, program umožňuje jejich modely zadávat jako tzv. "makromodely". Jednou zadaný makromodel konstrukčního prvku lze pak jednoduchým příkazem vícenásobně přiřazovat na příslušná místa analyzované soustavy. Parametry makromodelů přitom nemusí být vždy shodné, ale lze je pro každé použití různě obměňovat.

Makromodely lze využívat i ve více úrovních, tzn. že v zadání makromodelů složitých konstrukčních prvků je možno užívat makromodely prvků jednodušších, atd. Jelikož jednou vypracované makromodely lze ukládat do souborů na vnější paměti počítače, program umožňuje uživatelům vytvářet uživatelské knihovny makromodelů orientované podle jejich zájmů a potřeb. To je výhodné zejména v případě standardizovaných konstrukčních prvků, vyráběných ve velkých sériích.

## 1.3. Výstup programu

Program umožňuje i automatickou linearizaci nelineárních dynamických či statických soustav ve vypočítaném nebo zadaném klidovém či jiném pracovním bodě.

Pro lineární nebo linearizovanou soustavu program vyjádří její libovolnou racionální přenosovou funkci nebo její odezvu na počáteční podmínky, a to v semisymbolickém tvaru. Semisymbolickým tvarem racionální funkce zde rozumíme podíl dvou polynomů v p s operátorem p vyjádřeným symbolicky a s kořeny nebo koeficienty vyjádřenými numericky.

Program současně určuje časové konstanty, činitele tlumení, vlastní kmitočty, rezonanční kmitočty a fázové posuvy jednotlivých přirozených kmitočtových modů linearizovaných dynamických soustav.

Pro linearizované soustavy program dále dokáže v semisymbolickém tvaru určit i časové charakteristiky a odezvy na počáteční podmínky libovolných přenosových funkcí. V tomto případě je symbolicky vyjádřena časová proměnná t a exponenciální i trigonometrické funkce odpovídající jednotlivým přirozeným modům časových odezev.

Program umožňuje i automatickou linearizaci nelineárních dynamických či statických soustav ve vypočítaném nebo zadaném klidovém či jiném pracovním bodě. Pro periodicky ustálené průběhy program dovoluje provádět Fourierovu analýzu na základě algoritmu tzv. rychlé Fourierovy transformace.

Během výpočtu časových odezev dynamických soustav programu podle potřeby určuje výskyt tzv. událostí, tj. časových okamžiků, v nichž nastane určitý, předem definovaný stav odezev. Současně dokáže určit i časové intervaly, které mezi těmito událostmi uplynuly. Toho lze v programu využít nejen k vyhodnocování časových průběhů, ale i k řízení běhu dalších výpočtů, k řízení změn v konfiguraci analyzované soustavy (např. spínačů) apod.

Program výsledky numerických výpočtů nejen tabeluje, ale i graficky znázorňuje prostřednictvím abecedně-číselných znaků v podobě tzv. semigrafů. Stejným způsobem znázorňuje histogramy kmitočtových spekter. U grafů lze volit jejich šířku podle média, na němž se zobrazují (např. široký graf na tiskárně, úzký na obrazovce apod.). Měřítka všech grafů program volí automaticky tak, aby byl co nejlépe využit jejich rozsah. Jelikož program výsledky současně ukládá do souborů ve vnější paměti, lze je odtud využívat pro grafické znázornění na grafickém displeji, na souřadnicovém zapisovači apod.

Většina parametrů pro řízení výpočtů je v programu zvolena implicitně, uživatel však má možnost jejich hodnotu podle potřeby měnit.

## 1.4. Použité metody

Způsob automatické formulace rovnic popisujících branová schémata, který je v programu DYNAST použit, je založen na modifikované metodě uzlových napětí [12]. Tato metoda byla rozšířena i na bloková schémata [13]. Tím, že program formuluje všechny rovnice branového schématu současně, odpadají potíže s tzv. rychlými nebo algebraickými smyčkami obvyklé u běžných simulačních programů.

V případě **časové analýzy** program používá formulaci v podobě soustavy obyčejných obecně nelineárních algebro-diferenciálních rovnic 1. řádu v implicitním tvaru

$$F(x(t), \dot{x}(t), t) = 0 \qquad (1.1)$$

kde x(t) je vektor tzv. primárních proměnných, $\dot{x}(t)$ je vektor derivací primárních proměnných podle nezávisle proměnné t, F(.) je vektorová funkce.

V případě statických soustav nebo **statické analýzy** dynamických soustav ovšem $\dot{x}(t) = 0$ a (l.l) tak degeneruje na soustavu nelineárních algebraických rovnic

$$F(x(t), t) = 0 \qquad (1.2)$$

K řešení soustavy ( 1.1 ) při časové analýze se používá implicitní vícekroková **integrační metoda** charakterizovaná lineárním polynomiálním vzorcem

$$x_{n+1} = h_{b-1}\dot{x}_{n+1} + \sum_{i=0}^{r-1} a_i x_{n-1} \qquad (1.3)$$

kde *n* je index diskretizované nezávisle proměnné *t*.

Řád metody *r*, délka integračního kroku *h* a koeficienty metody $a_i$ a $b_i$ se během integrace optimalizují v závislosti na průběhu řešení automaticky tak, aby si výpočet při respektování přípustné zbytkové chyby integrace vyžádal co nejkratší dobu [10]. Koeficienty $a_i$ a $b_i$ jsou přitom voleny tak, aby metoda byla numericky stabilní i pro dynamické soustavy s velkým rozptylem časových konstant. Řád metody se během integrace mění v intervalu od 1 do 6.

V programu se soustava rovnic (1.1) formuluje prostřednictvím tzv. "matic-razítek" jednotlivých branových i blokových prvků přímo v časově diskretizovaném a inkrementálně linearizovaném tvaru

$$\left[\left(\frac{\partial F}{\partial x}\right)_n^{(k)} + \gamma\left(\frac{\partial F}{\partial \dot{x}}\right)_n^{(k)}\right]\Delta x_n^{k+1} = -F(x_n^k, \dot{x}_n^k, t_n) \qquad (1.4)$$

kde v hranaté závorce vystupují jakobiány vektorové funkce F(.) z levé strany (1.1) vzhledem k $x$ a $\dot{x}$, což jsou aproximace vektorů $x(t_n)$ a $\dot{x}(t_n)$ v k-té iteraci řešení soustavy (1.4). Integrační činitel $\gamma = hb$ je závislý jak na délce integračního kroku *n*, tak i na řádu integrační metody *r*. $\Delta x_n^{(k+1)} = x_n^{(k+1)} - x_n^{(k)}$ je oprava aproximace vektoru $x(t)$ získaná v k-té iteraci. Po zkonvergování iterací se počáteční odhad řešení v dalším kroku volí na základě vzorce (1.3).

Pro ustálené periodické průběhy získané při časové analýze program dovoluje provádět jejich **Fourierovu analýzu** na základě algoritmu tzv. rychlé Fourierovy transformace.

V případě analýzy nelineárních statických soustav a statické analýzy dynamických soustav je druhý člen hranaté závorky v (1.4) nulový a použitá integrační metoda přejde na známou metodu Newtona-Raphsona pro iterativní **řešení soustav nelineárních algebraických rovnic** (1.2).

Popsaná integrační metoda se však využívá nejen při časové analýze dynamických soustav, ale i při analýze parametrické, tj. při analýze závislosti statických soustav nebo statického řešení dynamických soustav na různých parametrech samotných soustav nebo na parametrech jejich okolí (např. na teplotě). Nezávisle proměnná t potom představuje měnící se parametr.

Linearizovaná soustava algebraických rovnic (1.4) se řeší algoritmem LU, tj. modifikací Gaussovy eliminační metody založené na rozkladu matice soustavy na součin horní a dolní trojúhelníkové matice. Tento rozklad se však spolu s výpočtem jakobiánů v (1.4) v zájmu numerické účinnosti neprovádí v každém iteračním kroku, ale jen tehdy, zpomalí- li se konvergence iterací.

Velmi výrazné úspory strojového času i operační paměti počítače se v programu dosahuje **využitím řídkosti** Jacobiho **matic** vystupujících v (1.4). Do operační paměti jsou ukládány pouze nenulové prvky matic a při rozkladu matic jsou předem vyloučeny ty operace, které vedou k nulovým mezivýsledkům. Přitom řídkost matic rychle roste se složitostí analyzovaných soustav.

Po zkonvergování iterací lze linearizovaný vztah (1.4) prostřednictvím Laplaceovy transformace převést na tvar

$$(A_0 + pA_1)X(p) = B(p) \qquad (1.5)$$

kde p je Laplaceův operátor. Pomocí zobecněného Cramerova pravidla pak z (1.5) lze vyjádřit libovolnou racionální **přenosovou funkci** linearizované soustavy nebo její odezvu na počáteční podmínky **v semisymbolickém tvaru** [14].

K redukci (1.5) na matice, jejichž charakteristickými čísly jsou póly a nuly požadované racionální operátorové funkce jsou v programu implementovány podprogramy z [11], využívající řídkost matic v (1.5). K vlastnímu **výpočtu pólů a nul** je použit zdvojený algoritmus QR pro výpočet charakteristických čísel redukovaných matic. Koeficienty polynomů se v případě potřeby počítají z pólů a nul.

Výpočet časových charakteristik operátorových funkcí se provádí symbolickou zpětnou Laplaceovou transformací.

## 1. 5. Vývoj programu

Program DYNAST vychází z programu SADYS, který navazoval na několik verzí programu DAVID. Programy DAVID, určené pro analýzu elektronických obvodů a systémů [4], [5], [6], byly vytvořeny na katedře teorie obvodů FEL ČVUT pod vedením autora. Program DAVID4 v roce 1952 úspěšně obstál při národních a v roce 1983 i při mezinárodních zkouškách programů, na jejichž základě byl zařazen do katalogu programů RVHP.

Program SADYS, jehož určení bylo proti programům DAVID daleko širší, vznikal již po ukončení příslušného vědecko-výzkumného úkolu katedry na soukromých osmibitových výpočetních prostředcích s operační pamětí 64 kbyte pod operačním systémem CP/M v jazyku FORTRAN IV. Byly vypracovány verze programu určené pro minipočítače řady SM s operačním systémem DOS RV, i pro počítače řady EC s operačním systémem DOS.

Program SADYS se od programu DAVID4 odlišoval především možností

- zadávat přímo i rovnice, nikoliv jen schémata,
- používat makromodely,
- využívat symbolické derivování,
- počítat i události a intervaly.

Podstatnou část programů DAVID4 i SADYS naprogramoval Ing. Z. Oliva, CSc.

Cílem současného vývoje programu DYNAST je nejen jeho další zdokonalování, ale i jeho začleňování do stále komfortnějšího uživatelského prostředí. Současně je dále rozvíjena systematická metodika jeho efektivního využívání v různých oborech.

# 2. Struktura programu

## 2.1. Soubory programu

Program DYNAST je uložen v jediném souboru. Program spolupracuje s následujícími pracovními soubory:

- soubor vstupních údajů,
- soubory maker,
- soubor výstupních údajů,
- soubor výstupních grafických údajů,
- soubor počátečních podmínek,
- pomocný soubor DYN.TMP.

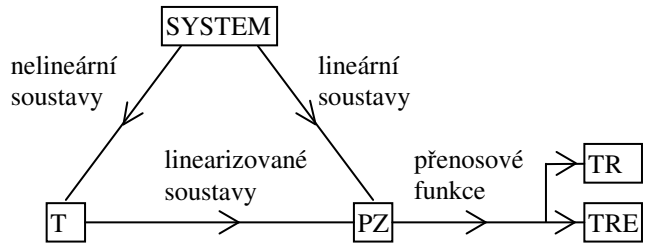
## 2.2. Sekce programu

Program je rozdělen do několika sekcí, které sdílejí společné soubory dat. Označení a účel jednotlivých sekcí programu uvádí tab. 2.1.

Tab.2.1: Sekce programu DYNAST

| Sekce | Účel |
|---|---|
| SYSTEM | Načtení zadání analyzované soustavy v podobě algebro-diferenciálních rovnic, blokového či branového schématu nebo jejich kombinace. Zadání výpočtu událostí a intervalů. |
| APP | Účel této sekce je shodný se sekcí SYSTEM s tím,že navíc umožňuje dodatečné načítání dalších rovnic nebo blokových či branových prvků. |
| TR | Výpočet statického řešení,výpočet časového řešení vycházející ze zadaných počátečních podmínek nebo ze statického řešení,výpočet závislostí veličin soustav na změnách jejich parametrů,výpočet kmitočtového spektra ustáleného periodického řešení,výpočet událostí v průbězích odezev a jejich intervalů, linearizace soustav v klidovém nebo zadaném pracovním bodě. |
| PZ | Semisymbolická analýza,tj.výpočet pólů,nul a koeficientů přenosových funkcí a obrazů odezev na počáteční stav linearizovaných soustav. |
| FRE | Výpočet různých složek kmitočtových charakteristik semisymbolicky vyjádřených přenosových funkcí. |
| TRA | Semisymbolický a numerický výpočet časových charakteristik a odezev semisymbolických přenosových funkcí. |

Přirozenou vzájemnou následnost volání sekcí ukazuje diagram na obr. 2.2.. Přípustné jsou však i návraty proti směru šipek v diagramu a při jednom běhu programu lze řešit i několik různých úloh, přičemž každá z nich může být řešena i několikrát s různě modifikovanými prvky a parametry.




Obr. 2.1: Následnost sekcí programu DYNAST

## 2.3. Věty vstupního jazyka

Vstupní údaje pro program DYNAST se zadávají formou vstupních vět, které se ukončují znakem ; . **Věta** může pokračovat i na několika řádcích. Na jednom řádku může být i několik vět. Přípustný počet znaků na jediném řádku je 80. Věty sestávají z příkazů, identifikátorů a čísel v podobě řetězců obsahujících výhradně abecedně-číselné znaky. Tyto řetězce se ve větách navzájem oddělují mezerami nebo ne-abecedně-číselnými znaky (např. / , - , = apod.) v souhlase s pravidly vstupního jazyka popsanými v následujících kapitolách.

**Identifikátory** používané ve vstupních údajích např. k označování parametrů, proměnných, prvků a uzlů analyzovaných soustav mohou sestávat až z osmi abecedně-číselných znaků. Je-li použito více znaků, počínaje devátým jsou programem ignorovány. Při volbě identifikátorů uživatel musí přihlédnout k tomu, že některé identifikátory jsou již v programu pevně vyhrazeny pro určitý účel a tudíž nesmí být užity k účelu jinému.

Pokud se na řádku vyskytne znak : , část řádku napravo od něho je považována za komentář, který program zobrazí ve výstupním souboru, jinak jej však ignoruje. (Této vlastnosti je možno využívat při odlaďování úloh k dočasnému blokování určitých vět.)

Začíná-li řádek znaky *, následující text na řádku se ve výstupním souboru zobrazuje jako záhlaví tabulek a grafů.

Program nerozlišuje mezi **znaky** malé a velké **abecedy**. V programu jsou všechny znaky interpretovány jako velké. Znaky národní abecedy (např. české či slovenské) mohou být použity jen v komentářích.

## 2.4. Popis vstupních vět

V následujícím popisu pravidel pro zadávání vstupních vět vstupního jazyka programu DYNAST budeme užívat tyto zásady:

- konstanty nebo proměnné uvedené v hranatých závorkách [] lze ve vstupních údajích vypustit,
- z konstant nebo z proměnných uvedených ve svorkách { } pod sebou se volí jedna jediná podle potřeby,
- pokud je v obecném popisu formátu určité věty, k oddělení některých jejích dvou složek použita alespoň jedna mezera, potom v zadávané větě tyto dvě složky mohou být odděleny libovolným počtem mezer.

V obecném popisu formátu různých vět vstupního jazyka budeme k typografickému vyznačení jednotlivých řetězců určovaných uživatelem používat *kurzívu*. K vyznačení řetězců v programu pevně vyhrazených budeme používat `strojové písmo`. Tento typ písma budeme používat i v příkladech.

## 2.5. Vstupní soubor

Celkové uspořádání údajů ve vstupním souboru programu DYNAST je následující

```
 : komentář
 *sekce
.
.          údaje pro sekci
.
*sekce
.
.          údaje pro sekci
.
*sekce
.
.
.
*END
```

Identifikátory sekcí programu DYNAST *sekce*, jež jsou uvedeny v tab. 2.1, se zadávají za znakem * bez mezery. Větou *END; se ukončí běh programu. Údaje uvedené za touto větou program ignoruje.

# 3. Formulace úloh

# 4. Proměnné rovnic a schémat

## 4.1. Třídění proměnných

V zadání řešených úloh mohou vystupovat jednak nezávisle a jednak závisle **proměnné veličiny** a **parametry**, v dalším stručně nazývané **proměnné**.

**Nezávisle proměnné** se v programu rozlišují na vnitřní a vnější, jak naznačuje tab. 4.1.

Tab. 4.1: Proměnné veličiny a parametry

| | |
|---|---|
| Nezávisle proměnné | vnitřní |
| | vnější |
| Závisle proměnné | primární |
| | sekundární |

**Vnitřní proměnné** jsou nezávisle proměnnými, jejichž identifikátory jsou pevně vyhrazeny v programu DYNAST. Uvádí je tab. 4.2.

Tab. 4.2: Vnitřní nezávisle proměnné

| Proměnná | Význam |
|---|---|
| TIME | obvykle čas v s |
| TEMP | parametr, např. teplota v kelvinech |
| FREQ | kmitočet v Hz |
| SIGMA | reálná část Laplaceova operátoru *p* |
| GAMA | integrační parametr |

**Vnější proměnné** jsou nezávisle proměnné, jejichž identifikátory si definuje sám uživatel programu. Tyto proměnné program vypočítá prostým dosazením do zadaných explicitních vztahů.

**Závisle proměnné** jsou v programu rozlišeny jako primární a sekundární (viz tab. 4.1.). Výpočet **primárních proměnných** vyžaduje řešení alespoň jedné implicitní rovnice nebo analýzu blokového či branového schématu.

**Sekundární proměnné** program vypočítá prostým dosazením do zadaných explicitních vztahů podobně jako v případě vnějších nezávislých proměnných. Rozdíl je však v tom, že ve vztazích pro výpočet sekundárních závisle proměnných vystupuje jako argument alespoň jedna primární závisle proměnná, zatímco v definičních vztazích mohou vystupovat pouze nezávisle proměnné. Sekundární proměnné lze získat i analýzou blokových a branových schémat.

V symbolicko-číselných výrazech mohou vystupovat dvouoperandové **operátory**, uvedené v tab. 4.4.

**Funkce**, které mohou přímo vystupovat v symbolicko-číselných výrazech, jsou uvedeny v tab. 4.5. Funkce jsou zde rozděleny na

- logické,
- základní,
- další,
- standardní.

## 4.1.1. Číselné konstanty

**Číselné konstanty** se v programu DYNAST zadávají s formátem

$$\left\{ \begin{array}{l} mantisa \quad \left[ E \left\{ \begin{array}{c} [+] \\ - \end{array} \right\} charakteristika \right] \\ mantisa \quad \left\{ \begin{array}{c} [ná\,sobná\ \ konstanta][text] \\ \_\,text \end{array} \right\} \end{array} \right\}$$

*Mantisa* může mít desetinnou tečku umístěnou na kterémkoliv místě. Je-li číslem celým, může se zadávat bez tečky. Pokud je číslem v absolutní hodnotě menším než jedna, nula před desetinnou tečkou se může vynechat. *Charakteristika* musí být celé číslo, ležící v intervalu přípustném pro použitý počítač.

Druhý způsob zadávání číselných konstant využívá **násobných konstant** z tab. 4.3. Mezi mantisou a násobnou konstantou nesmí být mezera.

Tab. 4.3: Násobné konstanty

| Symbol | Význam | Hodnota |
|---|---|---|
| T | tera | $10^{12}$ |
| G | giga | $10^{9}$ |
| ME | mega | $10^{6}$ |
| K | kilo | $10^{3}$ |
| M | mili | $10^{-3}$ |
| U | mikro | $10^{-6}$ |
| N | nano | $10^{-9}$ |
| P | piko | $10^{-12}$ |
| F | femto | $10^{-15}$ |
| PI | Ludolfovo číslo | $\pi$ |

Bezprostředně za násobnou konstantou může ještě následovat textový řetězec *text*, představující i s násobnou konstantou nejvýše 8 abecedně-číselných znaků. Tento *text* slouží pouze k lepší orientaci uživatele, program jej ignoruje. Pokud přitom žádnou násobnou konstantu neužijeme, *text* je nutno od *mantisy* oddělit znakem _ (podtržítkem).

**Příklad:**

Číselné konstanty můžeme zadat např. jako

-3.4 .3 67.08E-10 -2PI 6.3K 5NANO 1KVOLT 100_OHMU

Např. 1FARAD program interpretuje jako $10^{-15}$.

Tab.4.4: Operátory

| Operátor | Význam | Operátor | Význam |
|---|---|---|---|
| + | součet | <= | menší nebo rovno |
| - | rozdíl nebo unární minus | > | větší než |
| * | součin | >= | větší nebo rovno |
| / | podíl | & | logické AND |
| ** | mocnina | ! | logické OR |
| = | rovná se | ' | logické NOT |
| <> | nerovná se | % | operátor derivace |
| < | menší než | | (jen jako poslední ve výrazu) |

## 4.1.2. Symbolicko-číselné výrazy

V programu DYNAST lze hodnoty proměnných i jejich vzájemné funkční závislosti zadávat nejen číselnými konstantami, ale i **symbolicko-číselnými výrazy**. V těchto výrazech přitom mohou vystupovat:

- číselné konstanty,
- identifikátory proměnných,
- operátory,
- funkce.

S využitím logických funkcí uvedené výrazy mohou být zadávány i jako výrazy podmíněné.

## 4.1.3. Logické funkce

**Logické funkce** mohou nabývat jedné ze dvou hodnot, bud 1 nebo 0. V symbolicko-číselných výrazech se logické funkce s příslušným logickým operátorem uvádějí v závorkách ().

**Příklad:**

Logickou funkci X = (A OR B) AND (C OR D) lze zadat jako

```
X = (A ! B)&(C ! D);
```

nebo jako

```
X = (A ! B) * (C ! D) ;
```

## 4.1.4. Základní a další funkce

**Základní funkce a další funkce** se od standardních odlišují tím, že jejich *typ* představuje současně i jejich identifikátor. U standardních funkcí si jejich identifikátor určuje uživatel.

**Příklady:**

Tak matematický vztah

$X = V. \sin (I^2 + 3)$

lze v programu DYNAST zadat symbolicko-číselným výrazem

```
X = V * SIN(I**2 + 3);
```

kde X, V a I jsou identifikátory proměnných definované uživatelem.

Např.vztah
$I = I_0(e^{0.V} - 1)$

linearizovaný pro V vně intervalu <0,1>, jemuž odpovídá podmíněný výraz

$$I = \begin{cases} I_0(e^{\theta.V} - 1) & \theta < V < 1 \\ I_0(e^{\theta} - 1) \quad pro & V \geq 1 \\ I_0\theta & V \leq \theta \end{cases}$$

lze zadat větou

```
I = (IO*EXP(THETA*V) -1)*(V > 0)*(V < 1)
+ (IO*EXP(THETA) -1)*(V >=1) + IO*THETA*(V < 0);
```

Tab. 4.5: Funkce použitelné v symbolicko-číselných výrazech

| Funkce | Typ | Význam |
|---|---|---|
| logické | | dvojková hodnota 0,1 |
| základní | ABS | Absolutní hodnota |
| | EXP | exponenciální funkce |
| | SIN | sinus |
| | COS | cosinus |
| | TAN | tangens |
| | ATAN | arcustangens |
| | SINH | hyperbolický sinus |
| | COSH | hyperbolický kosinus |
| | TANH | hyporbolický tangens |
| | LOG | přirozený logaritmus |
| | LOG 10 | dekadický logaritmus |
| | E10 | dekadická exponenciální funkce |
| další | SQRT | druhá odmocnina |
| | INT | celá část |
| | SGN | znaménko |
| | CTN | cotangens |
| | ASIN | arcussinus |
| | ACOS | arkuskosinus |
| | ACTN | arkuskotangens |
| | COTGH | hyperbolický kotangens |
| standardní | viz nahoře | základní funkce |
| | LIN | lineární funkce |
| | POLY | standardní polynom zadaný koeficienty |
| | ROOT | polynom zadaný kořeny |
| | TAB | tabelovaná funkce |
| | PULSE | impulsní funkce |

Výraz

$$f = \frac{d^2}{dxdt} x^3 \sin(t)$$

lze zadat jako

```
F = X**3*SIN(TIME)%X%TIME;
```

## 4.1.5. Standardní funkce

**Standardní funkce** mají tvar $f(x) = A + B.g(C.x + D),\ L < x < U$

kde g(.) je jedna ze základních funkcí uvedených v tab. 4.5. Činitelé A, B, C, D, E, L a U jsou parametry standardní funkce. Přitom vně intervalu (L, U) platí

$$\frac{df}{dx} = \left(\frac{df}{dx}\right)_{x=L} \quad pro \quad x \le L$$
$$\frac{df}{dx} = \left(\frac{df}{dx}\right)_{x=U} \quad x \ge U$$

Znamená to, že implicitně je derivace $\frac{df}{dx}$ v bodech $x$ = L i $x$ = U spojitá, přičemž funkční průběh je pro $x$ vně intervalu (L, U) lineární.

Strmost lineární extrapolace funkčního průběhu pro $x$ vně intervalu <L,U> však uživatel může v případě potřeby zadat dalšími parametry $S_L$ a $S_U$. Potom platí

$$\frac{df}{dx} = S_L \quad pro \quad x \le L$$
$$\frac{df}{dx} = S_U \quad x \ge U$$

Je-li standardní **funkce** f(x) **periodická**, její periodu lze zadat jako parametr P a tudíž platí

$$f(x + k.P) = f(x)$$

kde *k* je celé číslo.

Vstupní věta pro zadání standardní funkce má formát:

*funkce* / *typ* / [A = *hodnota*,] [B = *hodnota*,] [E = *hodnota*,] [P = *hodnota*,]
[L = *hodnota*,] [U = *hodnota*,] [SL = *hodnota*,] [SU = *hodnota*]

*Funkce* je identifikátor zadávané funkce. Musí odpovídat některému z typů funkcí uvedených v tab. 4.5. *Hodnota* parametru funkce může být zadána jako číselná konstanta nebo i jako symbolicko-číselný výraz. Parametry Ize zadávat v libovolném pořadí.

Implicitní hodnoty parametrů standardních funkcí jsou A = D = 0. , B = C = E = 1. , L = $\infty$. Pokud některý z parametrů má mít implicitní hodnotu, není nutné jej zadávat.

V dalším textu budeme parametry standardních funkcí A, B, . . . , SU označovat jako **standardní** funkční **parametry**.

**Příklady:**

Vztah

$$I = I_0(e^{\theta.V} - 1)$$

linearizovaný pro V vně intervalu <0,1> lze zadat rovněž následovně

```
DIODE /EXP/ A = -I0, B = I0, C = A, L = 0, U = 1;
I = DIODE (V);
```

Vstupními větami

```
Y = 3/SQRT(I.L**2 + 10*cos(5*(I.L + V.C)**2));
Z = 10*cos (5*I. L) % I.L;
```

lze zadat výrazy

$$Y = \frac{3}{\sqrt{I_L^2 + 10.\cos 5(I_L + V_C)^2}}$$

$$Z = \frac{d}{dI_L} 10.\cos(5I_L)$$

kde $I_L$ je průtok prvku L a $V_C$ je spád prvku C určitého branového schématu.

## 4.1.6. Polynomiální funkce

**Polynomiální funkci**

$$f(x) = a_0 + a_1x + a_2x^2 + ... = k(x - x_1)(x - x_2)...$$

lze zadat dvěma způsoby. Buď prostřednictvím jejích koeficientů s formátem:

*funkce* / POLY / [*standardní parametry*,] $a_0, a_1, a_2, ...$

nebo prostřednictvím jejích kořenů s formátem:

*funkce* / ROOT / [*standardní parametry*,] $k, x_1, x_2, ...$

Koeficienty $a_0, a_1, a_2, ...$ mohou být reálná čísla, zadaná číselnými konstantami nebo symbolicko-číselnými výrazy. Je-li kořen $x_k$ komplexní, tj. je-li

$$x_k = \mathrm{Re}\, x_k + j\,\mathrm{Im}\, x_k ,$$

kořenem reálného zadávaného polynomu f(x) musí být i komplexně sdružený kořen

$$\overline{x_k} = \mathrm{Re}\, x_k - j\,\mathrm{Im}\, x_k$$

Při zadávání polynomiálni funkce typu ROOT se proto namísto dvou komplexně sdružených kořenů $x_k$ a $\overline{x}_k$ zadává pouze jediný, a to s formátem

$$(\mathrm{Re}x_k\,,\ \left\{\begin{matrix}[+]\\ -\end{matrix}\right\}\ \mathrm{Im}\ x_k)$$

kde znaménko imaginární části může být libovolné. Reálné kořeny i reálné a imaginární části komplexních kořenů je nutno zadat číselnými konstantami.

**Příklad:**

Polynomiální funkci

$$y = x^3 + x = x(x - j)(x + j)$$

lze zadat bud' prostřednictvím jejích koeficientů jako

F /POLY/ 0, 1, 0, 1; Y=F(x);

nebo prostřednictvím jejích kořenů jako

G /ROOT/ 1, 0, (0, 1); Y = G(x);

Vedle koeficientů nebo kořenů při specifikaci polynomiální funkce lze zadat i některé ze standardních parametrů.

**Příklad:**

Tak polynomiální funkci

$$g(t) = z(t) + 5z^3(t) - 8z^5(t)$$

kde

$$z(t) = 3t - 1,$$

přičemž *t* je nezávisle proměnná veličina, můžeme zadat takto:

FG /POLY/ C = 3, D = -1, 0, 1, 0, 5, 0, - 8; G = FG(TIME);

## 4.1. 7. Impulsní funkce

**Impulsní funkce** s lichoběžníkovým průběhem se zadává s formátem

funkce / PULSE / [*standardní parametry*] [L1= *hodnota*,] [L2 = *hodnota*,]
[TD = *hodnota*,] [TR = *hodnota*,] [TT = *hodnota*,] [TF = *hodnota*]

Parametry impulsní funkce mají následující význam:

| | |
|---|---|
| L1,L2 | ...úrovně impulsu |
| TD | ...doba zpoždění náběhu čela prvního impulsu |
| TR | ...doba trvání čela impulsu |
| TT | ...doba trvání temene impulsu |
| TF | ...doba trvání týlu impulsu |

Parametry lze opět zadávat číselnými konstantami nebo symbolicko-číselnými výrazy v libovolném pořadí. Implicitně je zadána hodnota parametru L2 = l, implicitní hodnoty ostatních parametrů jsou nulové.

Při zadávání impulsních funkcí lze rovněž využívat i standardní parametry.

**Příklad :**

Periodickou impulsní funkci $y = f(t)$ lze zadat např. takto:

```
TRAPER /PULSE/ L2 = 10, TT = 20U, TF = 10 U, P = 40U;
Y = TRAPER(TIME);
```

## 4.1.8. Tabelované funkce

**Tabelované funkce** $y = f(x)$ se zadávají navzájem si odpovídajícími dvojicemi hodnot argumentu x a funkce y s formátem:

*funkce* / TAB / [*standardní parametry*,] $x_1$, $y_1$, $x_2$, $y_2$, ...

Hodnoty $x_i$ a $y_i$ lze opět zadávat jako číselné konstanty nebo jako symbolicko-číselné výrazy. Musí však přitom platit $x_i \leq x_{i+1}$ pro všechna zadávaná $x_i$.

Při zadávání tabelovaných funkcí lze rovněž uplatnit i standardní parametry.

**Příklad:**

Impulsní periodickou funkci z předchozího příkladu za předpokladu, že první impuls začíná již v t = 0 (tj. že by v zadání odpovídající impulsní funkce bylo TD = 0), lze zadat jako tabelovanou funkci tímto způsobem:

```
TRAPER0 /TAB/ P = 40U, 0,10, 20U,10, 30U,0, 40U,0; Y = TRAPER0(TIME);
```

## 4.1.9. Události

**Událost** představuje určitou změnu stavu jedné nebo více proměnných analyzované soustavy. V programu DYNAST událostí současně rozumíme tzv. **proměnnou událost**, jejíž průběh udává, zda sledovaná událost již nastala.

**Popis události** se zadává formátem:

EVENT *událost* [ ( *pořadí* ) ] = *výraz* [, *událost* [ ( *pořadí* ) ] = *výraz* ... ]

*Událost* je identifikátor zadávané události i příslušné proměnné. *Pořadí* události je celé číslo, udávající kolikátý výskyt sledovaného stavu považujeme za zadávanou událost. Implicitně má *pořadí* hodnotu l. *Výraz* se zadává v symbolicko-číselném tvaru a je výrazem logickým. Událost je splněna právě v tom okamžiku, kdy se výraz stal pravdivým.

Dokud událost nenastane, příslušná **proměnná události** má hodnotu $10^{34}$. Při uskutečnění události příslušná proměnná nabývá hodnotu nezávisle proměnné, při níž událost nastala. Proměnné událostí patří mezi sekundární závisle proměnné.

**Příklady:**

Událost U, za kterou považujeme časový okamžik, při němž spád na branovém prvku C poklesl pod hodnotu 0,5 se zadá jako

EVENT U = V.C <= 0.5;

Zadání události B spočívající v tom, že poprvé po splnění události A průtok prvku RI překročil úroveň 0,5 směrem shora dolů je následující:

EVENT B = (A < l E34)*(I.RI <= 0.5)*(ID.RI < 0);

## 4.1.10. Intervaly

**Intervalem** mezi dvěma událostmi rozumíme rozdíl hodnot nezávisle proměnné odpovídajících těmto událostem. Interval zadáváme s formátem:

INTRV *interval* = *výraz* [, *interval* = výraz ...]

*Interval* je identifikátor intervalu. *Výraz* se zadává v symbolicko-číselném tvaru a specifikuje se jím rozdíl proměnných uvažovaných událostí.

**Příklady:**

Výpočet časového intervalu I, který uplyne mezi okamžiky, kdy spád na branovém prvku C poklesne z hodnoty 0,5 na hodnotu 0,3 se zadá následovně:

```
EVENT U1 = V.C <= 0.5, U2 = V.C <= 0.3 ;
INTRV  I = U1 - U2;
```

Šířka TPULS kladného impulsu uzlového spádu
V. OUT na úrovni 1,5 se určí takto:

```
EVENT    T1 = (V.OUT >= 1.5)*(VD.OUT > 0),
         T2 = (Tl < l E34)*(V.OUT <= 1.5)*(VD.OUT < 0);
INTRV    TPULS = T2 - Tl;
```

# 5. Soustavy rovnic

Program DYNAST může být použit k řešení lineárních i nelineárních algebraických, obyčejných diferenciálních i algebro-diferenciálních rovnic 1. řádu či jejich soustav. Zadání rovnic se do programu načítá v sekci SYSTEM.

Soustava řešených rovnic může sestávat ze vztahů tří typů, a to ze **vztahů definičních**

$$\begin{aligned} k_1 &= e_1(t) \\ k_2 &= e_2(k_1, t) \\ k_3 &= e_3(k_1, k_2, t) \end{aligned} \qquad (4.1a)$$

ze vztahů primárních

$$\begin{aligned} f_1(x_1, x_2, ..., x_n, x_1', x_2', ..., x_n', k_1, k_2, k_3, ..., t) &= 0 \\ f_2(x_1, x_2, ..., x_n, x_1', x_2', ..., x_n', k_1, k_2, k_3, ..., t) &= 0 \\ &... \\ f_3(x_1, x_2, ..., x_n, x_1', x_2', ..., x_n', k_1, k_2, k_3, ..., t) &= 0 \end{aligned} \qquad (4.1b)$$

a ze vztahů sekundárních

$$\begin{aligned} y_1 &= f_1(x_1, x_2, ..., x_n, x_1', x_2', ..., x_n', k_1, k_2, k_3, ..., t) \\ y_2 &= f_2(x_1, x_2, ..., x_n, x_1', x_2', ..., x_n', k_1, k_2, k_3, ..., t, y_1) \\ y3 &= f_3(x_1, x_2, ..., x_n, x_1', x_2', ..., x_n', k_1, k_2, k_3, ..., t, y_1, y_2) \\ &... \end{aligned} \qquad (4.1c)$$

Každý z definičních vztahů typu (4.1a) explicitně přiřazuje hodnotu vnější závisle proměnné $k_i$ prostřednictvím funkčního výrazu $e_i(.)$. Argumentem tohoto výrazu může být jednak určitá vnitřní nezávisle proměnná $t$ a jednak to mohou být vnější proměnné nadefinované již předchozími definičními vztahy.

K vyhodnocení vnější nezávisle proměnné $k_i$ na základě vztahů (4.1a) stačí postupně dosadit příslušné hodnoty proměnných $t$, $k_1$, $k_2$, ..., $k_{i-1}$ do výrazů $e_1(.)$, $e_2(.)$, $e_3(.)$. Vztahy (4.la) tedy nepředstavují rovnice, které by bylo zapotřebí v pravém slova smyslu řešit.

Definiční vztahy se zadávají s formátem:

*proměnná* = *hodnota*

kde *proměnná* je identifikátor vnější nezávisle proměnné a *hodnota* je číselná konstanta nebo symbolicko-číselný výraz. V tomto výrazu mohou vystupovat identifikátory vnitřních proměnných a identifikátory těch vnějších proměnných, které již byly zadány předchozími větami (procedurálnost jazyka).

Primarní vztahy typu (4.lb) představují vlastní implicitní rovnice, které je potřeba řešit. Přitom $x_i$ je i-tá primární závisle proměnná a $\dot{x}_i$ je její derivace podle nezávisle proměnné $x$. Počet primárních vstahů musí být roven právě počtu proměnných $x_i$, které v těch to vstazích vystupují.

Primární vstahy se zadávají v implicitním tvaru s nulovou levou stranou, a to s formátem:

0 = *hodnota*

kde *hodnota* je abecedně-číselný výraz. V tomto výrazu mohou vystupovat identifikátory primárních proměnných a identifikátory vnitřních i vnějších nezávislých proměnných, které již byly definovány.

**Derivace primárních proměnných** podle vnitřní nezávisle Proměnné *x* se zadávají s formátem:

VD.*proměnná*

kde *proměnná* je identifikátor příslušné primární závisle proměnné.

Primární *proměnné* musí být před zadáním primárních vztahů deklarovány větou s formátem:

SYSVAR *proměnná* [, *proměnná* ] [, *proměnná* ...]

Jako *proměnná* se zde uvádí identifikátor každé primární proměnné vystupující, v primárních vztazích. Uvedená věta může být ve vstupním souboru použita i vícekrát s různými (nebo i shodnými) identifikátory primárních proměnných. Každá primární proměnná však musí být deklarována před jejím prvním použitím v zadání některé rovnice.

Sekundární vztahy typu (4.1c) přiřazují hodnoty sekundárním proměnným $y_i$ pomocí explicitních výrazů, v nichž vystupují primární závisle proměnné $x_i$ a jejich derivace $\dot{x}_i$. Kromě toho zde mohou vystupovat i nezávisle proměnné $k_i$ a ty sekundární proměnné, které již byly specifikovány předchozími sekundárními vztahy.

**Pořadí vztahů** (4.l) v zadání může být libovolné do té míry, že definiční, primární a sekundární vztahy mohou být navzájem prostřídány. Pořadí primárních vztahů, nemá vliv na řešitelnost nebo na výsledek řešení zadané úlohy z tohoto důvodu, že jsou v programu řešeny přímými numerickými metodami všechny současně.

V definičních a sekundárních vztazích však musí být respektována **procedurálnost vstupního jazyka**, tj. všechny jejich argumenty musí být zadány vztahy předcházejícími, kromě vnitřních proměnných.

**Příklad :**

Soustavu dvou nelineárních diferenciálních rovnic lze zadal např. jako

```
*SYSTEM;   A = SIN(100*TIME);      B = 1.5*A  .5;
SYSVAR X,Y;        0 = VD.X + Y - A;        0 = X*VD.Y + B;
                   Z = X Y;                 W = VD.X + Z/B;
```

kde vztahy v prvním řádku jsou definiční, ve druhém řádku primární a ve třetím sekundarní. Tutéž soustavu bychom však mohli zadat třeba i následovně

```
*SYSTEM;           SYSVAR X,Y;          Z = X - Y;
A = SIN(100*TIME); 0 = VD.X + Y - A;    B = 1.5*A -    .5;
W = VD.X + Z/B;    0 = X*VD.Y + B;
```

Definiční primární a sekundární vztah lze kombinovat i s blokovými a branovými schématy (viz násl. odstavce). Jako primární proměnné se pak uplatňují i závisle proměnné příslušející blokovým a branovým prvkům. Ty však obvykle nemusí být deklarovány příkazem SYSVAR.

# 6. Bloková schémata

## 6.0.11. Bloky

**Sortiment bloků** z nichž může sestávat blokové schéma analyzované programem DYNAST, najdeme v tab. 6.1 . Bloky mají jediný výstup, mohou však mít i více vstupu (kromě bloku BT). Výstupní proměnné, označené jako $y$ jsou pro jednotlivé typy bloků charakterizovány v závislosti na jejich vstupních i jiných proměnných $z_i$ charakteristickými vztahy uvedenými ve třetím sloupci tabulky.

Tab.6.1: Sortiment bloků

| Typ | Význam | Charakteristický vstah |
|---|---|---|
| BS | dynamický blok | $y = f(z_1,z_2,...)$ |
| BI | integrační blok | $y = \int f(z_1,z_2,...)dt + y_0$ |
| BD | derivační blok | $y = \frac{d}{dt} f(z_1,z_2,...)$ |
| BO | operační blok | $y = f(z1,z2,...), y = 0$ |
| BT | přenosový blok | $Y(p) = K\frac{M(p)}{N(p)} Z(p)$ |

**Dynamický blok** je charakterizován lineárním nebo nelineárním charakteristickým vztahem, který může být i časově nebo parametricky závislý. Pokud mezi argumenty tohoto vztahu není žádná závisle proměnná blok se chová jako autonomní **zdroj nezávisle proměnné** $y$.

Blok typu BS se chová skutečně jako blok dynamický jen tehdy, je-li mezi argumenty jeho charakteristického vztahu časová derivace alespoň jedné vstupní proměnné, jinak se de facto chová jako **blok statický**.

V případě **integračního bloku** i **derivačního bloku** charakteristický vztah, musí být lineární, tj. musí být, ve tvaru

$$f(z_1, z_2, ...) = k_1 z_1 + k_2 z_2 + ...$$

Nenulové **počáteční hodnoty** $y_0$ výstupních proměnných integračních bloků lze specifikovat při zadávání časové analýzy.

**Operační blok** má obdobné vlastnosti jako dynamický blok s tím, že jeho chování je charakterizováno vztahem v implicitním tvaru. Je tedy ekvivalentní primárnímu vztahu zadanému implicitní rovnicí.

**Přenosový blok** je charakterizován reálnou racionální Přenosovou funkcí. Je-li

$$M(p) = p^m + a_{m-1}p^{m-1} + ... + a_0$$

polynom čitatele a

$$N(p) = p^n + b_{n-1}p^{n-1} + ... + b_0$$

polynom jmenovatele této přenosové funkce, příslušný přenosový blok lze rovněž charakterizovat obyčejnou diferenciální rovnicí n-tého řádu

$$y^{(n)} + b_{n-1} y^{(n-1)} + ... + b_0 y = K( z^{(m)} + a_{m-1} z^{(m-1)} + ... + a_0 z$$

kde K je konstantní činitel přenosové funkce.

## 6.0.12. Struktura blokových schémat

Abychom ve vstupním souboru mohli zadat **strukturu** analyzovaného **blokového schématu** tj: vzájemnou interakci vstupů a výstupů jednotlivých bloků ve schématu, musíme označit uzly schématu vhodnými identifikátory.

Z principu blokového modelování vyplývá, že ke každému uzlu schématu může být připojen libovolný počet vstupů bloků, ale nejvýše jeden jediný výstup některého bloku. Vyjímku představují
tzv. virtuální sumátory

Při použití programu DYNAST se nesetkáme s problémy s tzv. algebraickými nebo rychlými smyčkami, běžnými u jiných simulačních programů. Je to dáno tím, že všechny prirmární vztahy generované pro blokový diagram jsou v programu řešeny všechny současně přímými numerickými metodami.

## 6.0.13. Popis bloků

**Popis blokových prvků** se v programu DYNAST zadává s formátem:

*blok* [> *typ*] *uzel* [= *hodnota*]

*Blok* zde značí identifikátor zadávaného bloku, *typ* je vyhrazený identifikátor jeho typu ve shodě s prvním sloupcem tab. 6.1. Pokud identifikátor *blok* zvolíme tak, že se jeho první znaky s tímto sloupcem shodují, potom znak > a *typ* můžeme vypustit.

Jako *uzel* se zadává identifikátor uzlu schématu, k němuž je připojen výstup dalšího bloku.

*Hodnota* představuje charakteristickou funkci f(.) zadávaného bloku v souhlase s tab. 6.1. Zadává se číselnou konstantou nebo symbolicko číselným výrazem. Pokud znak = a *hodnota* v zadání bloku chybí, program implicitně považuje jeho výstup za jednotkový.

Jako argument v symbolicko-číselném výrazu vyjadřujícím *hodnotu* mohou vystupovat identi- fikátory nezávislých proměnných a primárních proměnných. V případě lineárního dynamického bloku typu BT se na místě *hodnoty* zadává jeho přenosová funkce s formátem:

[*K* * ] [*čitatel* ] [/ *jmenovatel* ] (*vstup*)

kde *K* je číselně vyjádřený konstantní činitel přenosu, *čitatel* je identifikátor polynomu čitatele a jmenovatel identifikátor polynomu jmenovatele přenosové funkce. Tyto polynomy je nutno zadat předem. Pokud je konstanta jednotková nebo pokud je jednotkový některý z polynomů, lze je vypustit. Jako *vstup* se zadává v závorkách ( ) identifikátor uzlu schématu, k němuž je připojen vstup bloku.

**Příklady:**

Integrátor se vstupem připojeným k uzlu Z a s výstupem připojeným k uzlu Y se zadá např. jako
BINTEG Y = Z; nebo třeba jako INTEG > BI Y = Z;

Tutéž funkci však splní např. i následovně zadaný přenosový blok BT

M /POLY/ 0, 1; BT Y = 1/M (Z);

nebo operační blok BO zadaný jako

BO Y = VD.Y - Z;

## 6.0.14. Proměnné blokových schémat

Za primární závisle proměnné veličiny blokového schématu jsou v programu považovány všechny **uzlové proměnné** schématu, které se ve vstupním jazyku programu označují s formátem:

V.*uzel*

prvním znakem tohoto identifikátoru není číslice ale abecední znak. Znaky V. lze vypustit, pokud prvním znakem identifikátoru *uzel* není číslice.

**Derivace uzlových proměnných** schématu podle vnitřní nezávisle proměnné TIME se ve vstupním jazyku programu označují s formátem:

VD. *uzel*

kde *uzel* je identifikátor příslušného uzlu.

**Deklarace uzlové proměnné** jako proměnné primární se v programu DYNAST provádí implicitně jakmile se zadá blok, pro který je příslušný uzel uzlem výstupním. Pokud však je některá z uzlových proměnných použita v některém výrazu ještě dříve, než byla takto implicitně deklarována, je nutné ji deklarovat explicitně větou s příkazem SYSVAR.

**Příklad:**

Uvažujme např. zpětnovazební soustavu zadanou jako

BS1 z = sin(100*time); SYSVAR z;
BS2 y = 5*(x - z); BS3 z = y**2;

buzenou blokem BS1 s blokem BS2 v přímé větvi a s blokem BS3 ve větvi zpětné. Jelikož v zadání *hodnoty* bloku B2 vystupuje uzlová proměnná z, která přísluší uzlu výstupnímu vzhledem k bloku BS3 zadanému až dále, je nutné tuto proměnnou explicitně deklarovat.

Jako **sekundární proměnné** jsou v programu dostupné hodnoty charakteristických funkcí jednotlivých bloků schématu označené identifikátory bloků. Jsou však shodné s hodnotami příslušných uzlových proměnných.

## 6.0.15. Virtuální sumátory

Přesto, že principielně ke každému uzlu blokového schématu může být připojen výstup pouze jediného bloku, program DYNAST dovoluje uvádět shodný identifikátor výstupního uzlu v zadání více bloků současně (kromě bloků typu BT). Program pak v takovémto případě automaticky předpokládá, že mezi výstupy uvedených bloků a daný uzel je zařazen **virtuální sumátor** takže uzlová proměnná dotyčného uzlu je rovna součtu výstupních veličin připojených bloků.

Tento způsob zadání sumátoru je výpočetně výhodný, neboť vede k menšímu počtu formulovaných a řešených primárních vztahů. Výstupní veličiny jednotlivých bloků se společným výstupním uzlem však nejsou samostatně dostupné.

**Přiklad:**

Blokový diagram zadaný jako

BS1 1 = a; BS2 2 = b; BS3 4 = c*(V.1 + V.2);

vede na formulaci tří primárních vztahů, zatímco ekvivalentní blokový diagram zadaný s virtuálním sumátorem sčítajícím výstupní proměnné bloků BS1 a BS2 jako

BS1 3 = n; BS2 3 = b; BS3 4 = c*V.3;

vede na formulaci pouze dvou vztahů, které program musí při analýze schématu řešit. Výstupy bloků jsou však potom nedostupné, dostupný je pouze jejich součet v podobě uzlově proměnné V.3.

## 6.0.16. Rovnice blokových schémat

Pro zadané blokové schéma program DYNAST automaticky formuluje příslušné primární vztahy. Pro každý další uzel a tedy i pro každý další blok (kromě bloků s virtuálním sumátorem) připojený k blokovému schématu program DYNAST vygeneruje jeden primární vztah. Jenom pro blok typu BT těchto vztahů může být více, a to právě n + 1, je-li n stupeň polynomu ve jmenovateli příslušné přenosové funkce.

### 6.0.17. Popis přenosových funkcí

**Analýzu** racionální **přenosové funkce** v časové i v kmitočtové oblasti můžeme převést na analýzu bloku BT s přenosem odpovídajícím této funkci buzeným autonomním blokem BS. Při semisymbolické analýze v sekci PZ na hodnotě vstupu bloku BT přitom vůbec nezáleží.

**Příklad:**

Uvažujme přenosovou funkci

$$F(p) = K\frac{M(p)}{N(p)} = 5\frac{p^2}{(p+1-j)(p+1+j)}$$

Její analýzu můžeme převést např. na analýzu blokového diagramu zadaného jako

```
M /poly/ 1,0,1; N /root/ (1,1);
BS in; BT out = 5*M/N (in);
```

### 6.0.18. Bloková schémata s rovnicemi

Programem DYNAST lze analyzovat současně **bloková schémata** s definičními, primárními i sekundárními **rovnicemi**. Proměnné $z_i$ uplatněné v charakteristických vztazích bloků musí být potom buď nezávislými nebo primárními proměnnými či jejich derivacemi, nesmí být proměnnými sekundárními.

**Příklad:**

Např. při řešení Besselovy rovnice můžeme následujícím způsobem využít derivační bloky:

```
N = 1; SYSVAR X; BD XD = X ; BD XDD = XD;
0 = TIME**2*XDD + TIME*XD + (TIME**2 - N**2)*X;
```

# 7. Branová schémata

### 7.0.19. Branové prvky

**Sortiment branových prvků,** z nichž může sestávat branové schéma analyzované programem DYNAST, je uveden v tab. 7.2. Ve třetím sloupci tabulky nalezneme vztahy charakterizující parametry branových prvků jednotlivých typů. Přitom *i* zde značí průtok prvku a *v* jeho spád.

Přehled fyzikálního významu a fyzikálních jednotek **branových veličin**, tj. průtoků a spádů různé fyzikální podstaty uvádí tab. 7.2. Jsou zde uvažovány tzv. izomorfní analogie, které vedou na branová schémata o struktuře shodné se strukturou modelovaných reálných soustav.

Jako argument $z_i$ mohou v charakteristickém vztahu parametru kteréhokoliv prvku z tab. 7.2 vystupovat nezávisle proměnné i některé závisle proměnné příslušející témuž nebo jiným prvkům kromě té proměnné, kterou vztah explicitně vyjadřuje. Prvky tedy mohou být lineární i nelineární neřízené i řízené, stacionární i nestacionární.

Z hlediska obecné teorie mnohobranů na každý branový prvek můžeme pohlížet jako na zvláštní případ realizace jedné brány branového schématu. Vztah mezi průtokem a spádem této brány je dán charakteristickým vztahem parametru příslušného prvku. Branovými prvky můžeme modelovat jak jednobrany, tak i mnohobrany. K vytvoření modelu n-branu potřebujeme právě n branových prvků.

Branové prvky jsou v tabulce rozděleny do dvou tříd, do **třídy prvků Y** (admitančních) a do **třídy prvků Z** (impedančních). Prvky obou tříd se od sebe navzájem liší ve způsobu automatické formulace příslušných rovnic i tím, jak jsou v programu interpretovány jejich průtoky a spády.

Fyzikální význam a fyzikální jednotky parametrů branových prvků uvádí tab. 7.0.19. Vychází se zde opět z izomorfních analogií. Na skutečnosti, že mechanický odpor je interpretován jako odpor proti spádu; zatímco v ostatních fyzikálních soustavách je odpor interpretován jako odpor proti průtoku, nese vinu tradice. Poddajnost je převrácenou hodnotou tuhosti.

Tab. 7.1: Fyzikální význam branových veličin

| Soustava | Veličiny | | | |
|---|---|---|---|---|
| | průtok *i* | pád *v* | $\int i\,dt$ | $\int v\,dt$ |
| Elektrická | elektrický proud<br>A | elektrické napětí<br>V | elektrický náboj<br>C | magnetický tok<br>Wb |
| Mechanická translační | síla<br>N | rychlost<br>m/s | hybnost<br>Ns | dráha<br>m |
| Mechanická rotační | moment síly<br>Nm | úhlová rychlost<br>rad/s | rotační impuls<br>Nms | úhel<br>rad |
| Hydraulická<br>Pneumatická<br>Akustická | objemový tok<br><br>m/s | prostorový tlak<br><br>Pa | objem<br><br>m | |
| Tepelná | tepelný tok<br>W | teplotní rozdíl<br>K | tepelné množství<br>J | |
| Magnetická | magnetický tok<br>Wb | magnetické napětí<br>Az | | |

Tab.7.2: Sortiment branových prvků

| Typ | Prvek | Parametr |
|---|---|---|
| Prvky Y | | |
| J | zdroj průtoku | $i = f(z_1, z_2, ...)$ |
| R | odpor proti průtoku | $\frac{v}{i} = f(z_1, z_2, ...)$ |
| G | odpor proti spádu | $\frac{i}{v} = f(z_1, z_2, ...)$ |
| C | odpor proti změně spádu | $\frac{i}{dv/dt} = f(z_1, z_2, ...)$ |
| Prvky Z | | |
| E | zdroj spádu | $v = f(z_1, z_2, ...)$ |
| RI | odpor proti průtoku | $\frac{v}{i} = f(z_1, z_2, ...)$ |
| L | odpor proti změně průtoku | $\frac{v}{di/dt} = f(z_1, z_2, ...)$ |
| OA | operační prvek | $0 = f(z_1, z_2, ...)$ |

## 7.0.20. Struktura branových schémat

Abychom ve vstupních datech mohli zadat **strukturu** analyzovaného **branového schématu** tj. vzájemnou interakci průtoků a spádů jeho jednotlivých prvků, ve schématu musíme označit jeho uzly vhodnými **identifikátory uzlů**.

**Referenční uzel** přitom vždy označujeme identifikátorem 0 (nula). Celé schéma musí mít právě jeden referenční uzel. To znamená, že každé dvě **nesouvislé části schématu** musí být před zadáním do programu navzájem sjednoceny jedním uzlem (obvykle referenčním) (tato podmínka

Tab. 7.1: Fyzikální význam parametrů branových prvků

| Soustava | Parametry prvků | | | |
|---|---|---|---|---|
| | odpor proti průtoku | odpor proti spádu | odpor proti změně spádu | odpor proti změně průtoku |
| | $\frac{v}{i}$ | $\frac{i}{v}$ | $\frac{i}{dv/dt}$ | $\frac{v}{di/dt}$ |
| Elektrická | odpor<br>Ω | vodivost<br>S | kapacita<br>F | indukčnost<br>H |
| Mechanická translační | | odpor<br>kg/s | hmotnost<br>kg | poddajnost<br>$s^2$/kg |
| Mechanická rotační | | odpor<br>Nms/rad | moment setrvačnosti<br>$m^2$kg/rad | pddajnost<br>rad/Nm |
| Hydraulická<br>Pneumatická<br>Akustická | odpor<br><br>Pa .s/$m^3$ | | kapacita<br><br>$m^3$/Pa | poddajnost<br><br>Pa .$s^2$/$m^3$ |
| Tepelná | odpor<br>Ks/J | | kapacita<br>J/K | |
| Magnetická | odpor<br>A/V.s | | | |

ovšem nijak neomezuje aplikovatelnost programu). Některé typy prvků je možno zadávat i v **sériové kombinaci**, která pak představuje jedinou bránu schématu bez vnitřních uzlů.

## 7.0.21. Popis branových prvků

**Popis branových prvků** se zadává větami s formátem:

*prvek* [> *typ* ] 1.*uzel* [ $\left\{ \begin{matrix} - \\ , \end{matrix} \right\}$ 2.*uzel* ] [= *hodnota* ]

*Prvek* zde značí identifikátor zadávaného branového prvku, *typ* je vyhrazený identifikátor jeho typu ve shodě s prvním sloupcem tab. 7.2. Pokud identifikátor *prvek* zvolíme tak, že se jeho první znaky shodují s identifikátorem *typ*, potom znak > a identifikátor *typ* můžeme vypustit.

1.*uzel* a 2.*uzel* jsou identifikátory uzlů schématu, k niž je zadávaný prvek připojen. Pokud identifikátorem 2.*uzel* je 0 (tj. jedná-li se o referenční uzel), znaky,0 nebo -0 lze vypustit.

*Hodnota* specifikuje parametr zadávaného prvku charakterizovaný funkcí f(.) v souhlase s tab. 7.2. Zadává se číselnou konstantou nebo symbolicko-číselným výrazem. Pokud znak = a *hodnota* v zadání prvku chybí, program považuje jeho parametr za jednotkový.

## 7.0.22. Sériové kombinace prvků

Program DYNAST umožňuje zadávání sériových kombinací několika branových prvků třídy Z tak, že představují jedinou bránu. Zmenší se tím počet rovnic, které program musí řešit. Postupuje se přitom tak, že se v zadávané **sériové kombinaci prvků** jeden prvek zvolí jako referenční. Tento **referenční prvek** se zadá tak, jako kdyby představoval celou uvažovanou bránu, tzn. zadá se s identifikátory uzlů, mezi něž je brána připojena.

Ostatní prvky sériové kombinace se pak k referenčnímu prvku přiřadí zadáním s formátem:

*prvek_PRVEK* [= *hodnota*]

kde *prvek* značí identifikátor zadávaného prvku. *PRVEK* zde značí identifikátor referenčního prvku zadávané sériové kombinace.

Pokud je v sériové kombinaci zařazen i operační prvek, musí být považován za referenční prvek celé kombinace. U referenčního prvku lze zadat i shodné uzly, takže příslušná brána pak představuje uzavřenou smyčku.

**Příklad:**

Bránu mezi uzly 13 a 0 představovanou sériovou kombinací čtyř prvků se zdrojem spádu jako referenčním prvkem lze zadat např. jako

E6 13 = l0; R - E6 = .1K; R3 - E6 = 1K; L8 - E6 = 2M;

nebo třeba jako

BR > E 13 = 10 ; R - BR = .1K ; R3 - BR = 1K ; L8 - BR = 2M;

## 7.0.23. Induktivní vazba

Vzájemná **induktivní vazba** mezi dvěma prvky typu L se zadává s formátem:

*vazba induktor - induktor* [= *hodnota*]

kde *vazba* je identifikátor zadávané induktivní vazby, *induktor* je identifikátor každého z vázaných prvků L. Prvním znakem identifikátoru *vazba* je znak **M** nebo **K**. **M** značí, že *hodnota* udává přímo velikost **vzájemné indukčnosti** prvků **L**, kdežto **K** symbolizuje, že je zadáván jejich tzv. **činitel induktivní vazby**. Hodnota činitele induktivní vazby **K** souvisí s hodnotou vzájemné indukčnosti **M** dvou prvků typu L podle vztahu,

$$K = \frac{\pm M}{\sqrt{L_a L_b}}$$

kde $L_a$ a $L_b$ značí hodnotu každého z obou induktorů.

Znaménko u M i u K závisí na vzájemné orientaci induktorů (tzv. začátků vinutí) a jejich průtoků. Program DYNAST dovoluje uvedeným způsobem zadat vzájemno vazbu mezi libovolným počtem induktorů.

*Hodnota* specifikuje charakteristickou funkci zadávané vzájemné indukčnosti (může být i nestacionární). Zadává se číselnou konstantou nebo symbolicko-číselným výrazem. Pokud znak = a *hodnotu* v zadání prvku chybí, program považuje jeho charakteristickou funkci za jednotkovou.

**Příklad:**

Tři brány představující uzavřené smyčky s induktivní vazbou uvnitř 2. smyčky i mezi 2. a 3. smyčkou lze zadat např. jako

```
IN > E 0 ; RI-IN = 10. ;                    :1.smycka
E 0 = 50*I.RI; R2-E = 1K; L1-E = 8M;
L2 > L-E = 3M; K L1-L2 = - 0.95M;           :2.smycka
L 0 = 20M; OUT > R-L = 1.5K; M L-L2 = 1M;   :3.smycka
```

Analýza takto zadaného schématu si vyžádá řešení soustavy pouze tří primárních rovnic.

## 7.0.24. Proměnné branových schémat

Tab. 7.4 uvádí přehled závisle **proměnných** dostupných v programu DYNAST při analýze **branových schémat.**

Tab.7.4: Závisle proměnné branových schémat

| Formát | Význam | Proměnná |
|---|---|---|
| Y.uzel<br>I.prvek Z | spád uzlu<br>průtok prvku Z | primární |
| VD.vzel<br>ID.pruck 2 | derivace spádu uzlu<br>derivace průtoku prvku Z | primární derivovaná |
| *prvek*<br>V.*prvek*<br>I. *prvek* Y | parametr prvku<br>spád prvku<br>průtok prvku Y | sekundární |
| VD.*prvek* Y | derivace spádu | sekundární derivovaná |

**Spádem uzlu** rozumíme jeho absolutní spád, tj. spád uzlu vzhledem k referenčnímu uzlu schématu. **Spádem prvku** rozumíme spád brány příslušející prvku, tj. rozdíl spádů jejích uzlů. Přesněji,

$$V_1=V_1 - V_2$$

kde $V_p$ je spád prvku, $V_1$ resp. $V_2$ je spád uzlu, jehož identifikátor byl v zadání prvku použit jako první resp. jako druhý. Shodné zásady platí pro orientaci derivací spádů uzlů a prvků.

**Průtokem prvku** rozumíme branový průtok prvkem ve směru od uzlu, jehož identifikátor byl v zadání prvku použit jako první, k jeho druhému uzlu. Byl-li prvek zadán v sériové kombinaci orientace průtoku je dána pořadím identifikátorů uzlů příslušného referenčního prvku. Shodně je orientována i derivace průtoku prvku.

Tab. 7.4 uvádí, které z proměnných veličin branového schématu jsou v programu považovány za primární proměnné a které proměnné jsou v programu dostupné jako sekundární, příp. jako derivace primárních či sekundárních proměnných. U veličin prvků přitom může záležet na tom, jsou-li prvky třídy *Z* nebo *Y*, což souvisí s použitou metodou automatické formulace rovnic charakterizujících branové schéma.

Z tab. 7.4 je dále patrno, že předpona V. se užívá k označení spádu, předpona I. k označení průtoku a předpona VD. resp. ID. k označení jejich derivace vzhledem k vnitřní nezávislé proměnné. *Uzel* resp. *prvek* značí identifikátor příslušného uzlu resp. prvku.

Uzlový spád lze označovat přímo identifikátorem příslušného uzlu bez předpony V. , pokud tento identifikátor začíná abecedním znakem.

U prvků zadaných v sériové kombinaci jako spád referenčního prvku V.*PRVEK* program prezentuje spád příslušné brány, tj. celé sériové kombinace (a nikoliv jen skutečný spád na referenčním prvku). Obdobně je tomu s derivací spádu referenčního prvku VD. *PRVEK*. Spády jednotlivých prvků sériové kombinace nejsou dostupnou sekundární proměnnou.

Primární proměnné v podobě spádů uzlů jsou deklarovány implicitně zadáním příslušných uzlů. Podobě primární proměnné v podobě průtoků prvků Z jsou deklarovány implicitně zadáním příslušných prvků. Potřebujeme-li se však na některé primární proměnné v zadání *hodnot* některého prvku odkázat dříve, než byly takto implicitně deklarovány, musíme je deklarovat explicitně větou seznamem s příkazem SYSVAR.

## 7.0.25. Rovnice branových schémat

Příslušné **primární vztahy pro** zadané **branové schéma** program DYNAST automaticky formuluje modifikovanou metodou uzlových spádů. Při formulaci těchto primárních vztahů se bere v úvahu, že součet průtoků v každém uzlu schématu je roven nule a stejné tak, že součet spádů podél kterýchkoliv bran společně představujících uzavřenou smyčku je roven nule.

Rozšířením branového schématu o každý další uzel nebo prvek Z (kromě prvků Z zapojených v sériové kombinaci) **počet primárních vztahů** vzroste o jeden. Připojováním prvků Y a indukčních vazeb se počet primárních vztahů nezvyšuje: Připojením nulového prvku Y se primární vztahy naprosto neovlivní.

## 7.0.26. Branová schémata s bloky a s rovnicemi

Program DYNAST lze použít i k **analýze smíšených** branových a blokových **schémat** kombinovaných případně i s **rovnicemi** v podobě definičních, primárních a sekundárních vztahů (3.1).

Proměnné $z_i$ uplatněné v charakteristických vztazích branových prvků a bloků musí být přitom buď nezávislými nebo primárními proměnnými či jejich derivacemi, nesmí být proměnnými sekundárními.

**Bloky** jsou v programu z **branového hlediska** interpretovány jako řízené nebo neřízené zdroje spádu, jejichž druhým uzlem je referenční uzel schématu. Uzlové proměnné jsou pak interpretovány jako spády příslušných uzlů. Tím ovšem není nijak omezena použitelnost programu, at' již jsou ve skutečnosti uvažované vstupní a výstupní veličiny bloků jakékoliv fyzikální povahy obdobně, jako je tomu při modelování na analogových počítačích.

# 8. Makromodely

## 8.0.27. Vytváření makromodelů

Program DYNAST umožňuje vytvářet **makromodely** různých modulů analyzovaných soustav, ukládat je do periferní paměti počítače a opakovaně je odtud vyvolávat.

**Popis** ukládanébo **makromodelu** se zadává s následujícím formátem:

$$\text{model}\left\{\begin{matrix} uzel \\ větev \end{matrix}\right\}\left[\left\{\begin{matrix} - \\ , \end{matrix}\right\}\left\{\begin{matrix} uzel \\ větev \end{matrix}\right\}...\right]\left[/\,parametr\left[=hodnota\right]\left[,parametr\left[=hodnota\right]...\right]\right]$$

.
. *vnitřní popis makromodelu*
.
EO@;

*Model* je identifikátor typu makromodelu. Dále pak následuje seznam identifikátorů uzlů a/nebo bran **rozhraní makromodelu**, které zprostředkovávají interakci modelovaného modulu se zbývající částí analyzované soustavy. Každý identifikátor *uzel* uvedený v tomto seznamu musí souhlasit s identifikátorem některého z uzlů vnitřní struktury makromodelu.

Každý identifikátor *brána* se musí shodovat s referenčním identifikátorem některé brány představované určitou sériovou kombinací prvků vnitřní struktury makromodelu. Znamená to, že prvky této sériové kombinace se v popisu makromodelu zadávají s formátem:

*prvek* - BRÁNA [= hodnota]

Za seznamem identifikátorů uzlů a bran rozhraní může následovat ještě seznam identifikátorů **parametrů rozhraní** makromodelu oddělený znakem /. Jsou to identifikátory parametrů použitých ve vnitřním popisu vlastního makromodelu, jejichž hodnotu uživatel má možnost při použití makromodelu z jeho vnějšku měnit. Ke každému identifikátoru *parametr* lze za znakem = jako *hodnotu* přiřadit implicitní hodnotu příslušného parametru v podobě číselné konstanty. Pokud tato hodnota není u některého parametru zadána, program mu přiřadí nulovou implicitní hodnotu.

**Vnitřní popis makromodelu** může být zadán v podobě smíšené struktury blokových a branových prvků, případně i v podobě rovnic. Pro jejich zadávání platí stejné zásady, jaké již byly uvedeny v předchozích odstavcích. Zadání vnitřního popisu makromodelu se zakončuje příkazem

EO@.

Vytvořený makromodel s identifikátorem *model* se uloží do souboru

*model*.MOD

**Příklad :**

Dvojitý derivační článek RC s parametry $R_1 = a$, $R_2 = 2a$, $C_1 = b$, $C_2 = b/2$ jako makromodel s vnějšími uzly IN, REF a OUT a s implicitními hodnotami vnějších parametrů a $=10^3$ a b =10 lze zadat v souboru CRCR.MOD třeba takto :

```
CRCR IN, REF, OUT / a=1k,b=10;
R1 INNER-REF = a; R2 OUT-REF = b/2;
C1 IN-INNER = b; C2 INNER - OUT = b/2;
EO@;
```

Strukturu makromodelu představujícího ideální transformátor nebo ideální kinematickou vazbu s rozhraním v podobě uzlů A , B a brány BR můžeme zadat v souboru TRAFOID. MOD např. následovně:

```
TRAFOID A-B,BR / N=1;
E-BR =(V.A-V.B)*N; J A-B = -I.BR*N; EO@;
```

## 8.0.28. Užívání makromodelů

Jak jsme již uvedli, vyvolání makromodelu z periferní paměti počítače a jeho přiřazení k určité soustavě, kterou právě analyzujeme, se provádí přiřazením uzlů a bran rozhraní makromodelu k uzlům a branám soustavy.

**Popis přiřazení makromodelu** k soustavě se zadává s následujícím formátem:

$$[modul >]@\, model\left\{\begin{matrix} uzel \\ brána \end{matrix}\right\}\left[\left\{\begin{matrix} - \\ , \end{matrix}\right\}\left\{\begin{matrix} uzel \\ brána \end{matrix}\right\}...\right]$$

/ *parametr* [, = *hodnota* ] [, *parametr*[ = *hodnota* ] . . .]

Identifikátorem *modul* se v analyzované soustavě navzájem odlišují makromodely modelující její jednotlivé moduly. *Model* je identifikátor typu makromodelu použitého k modelování uvažovaného modulu. Tomuto identifikátoru předchází (bez mezery) znak @.

Za identifikátorem *model* následuje seznam identifikátorů uzlů a bran analyzované soustavy představujících rozhraní vyvolávaného makromodelu. Je-li v tomto seznamu identifikátor referenčního uzlu 0, nelze jej vypustit.

Identifikátor brány má tvar

I . *PRVEK*

kde *PRVEK* je identifikátor referenčního prvku uvažované brány v analyzované soustavě.

Identifikátory uzlů a bran soustavy představující rozhraní makromodelu se mohou lišit od identifikátorů odpovídajících uzlů bran rozhraní v makromodelu.

Počet identifikátorů *uzel* a *brána* rozhraní v soustavě a v makromodelu se však musí navzájem shodovat. Pokud ve skutečnosti některý uzel maktomodelu není v soustvě spojen s žádným jejím uzlem, nebo pokud některá brána makromodelu není spojena s žádnou z bran soustavy, soustavu musíme rozšířit o **fiktivní uzel** nebo o **fiktivní bránu** tak, aby k nim nepoužitý uzel nebo nepoužitá brána makromodelu mohly být přiřazeny. Nesmí se tím ale ovlivnit chování soustavy nebo regularita jejího matematického popisu. To, že nejsou přípustné uzly, k nimiž není připojen žádný prvek lze obejít připojením **fiktivních prvků** typu G o hodnotě 0.

Při přiřazování makromodelu k soustavě lze zadat **hodnoty** některých nebo všech **parametrů rozhraní** makromodelu. Chceme-li zadat pouze některé parametry rozhraní, jako *parametr* uvedeme identifikátor každého z nich (shodný s identifikátorem použitým pro daný parametr při vytváření makromodelu) a přiřadíme mu potřebnou hodnotu. Při zadávání hodnoty všech parametrů rozhraní můžeme postupovat tak, že v seznamu uvedeme pouze hodnoty parametrů bez identifikátorů ve stejném pořadí, v němž jsou parametry uvedeny v seznamu uvnitř rnakromodelu.

Jako *hodnotu* parametru rozhraní makromodelu můžeme uvést číselnou konstantu nebo symbolicko-číselný výraz. Nejsou-li hodnoty některých vnějších parametrů makromodelu při jeho přiřazování k soustavě zadány, parametry nabývají implicitních hodnot zadaných při vytváření makromodelu. Nebyla-li přitom implicitní hodnota některého parametru zadána, je považována za nulovou.

V zadání vnitřní struktury makromodelu mohou být jako stavební prvky používány i jiné makromodely.

**Příklad:**

Soustavu představovanou kaskádním spojením dvou dvojitých derivačních článků RC lze s využitím makromodelu CRCR zadat následovně:

- názvy
- parametry

```
*SYSTEM; E 1; RLOAD 3 = 100;
CIR1 > @CRCR 1, 2, 3 ;
CIR2 > @CRCR 2,0;2 / l0k, .1u;
```

První makromodel CIR1 je vyvoláván bez parametrů. Znamená to, že jeho parametry rozhraní v soustavě nabývají svých implicitních hodnot $a$ = $l0^3$ a $b$ =10. Druhý makromodel s názvem CIR2 má parametry $a$ =$10^3$ a $b$=$10^{-7}$.

Soustavu s makromodely dvou ideálních vazeb či transformátorů typu TRAFOID představujících serio-paralelní kombinaci je možno zadat takto:

```
*SYSTEM; E 1; RILOAD 0 = 100;
TR1 > @TRAFOID 1-O,I.RILOAD / N = 50;
TR2 > @TRAFOID 1-O,I.RILOAD / N = -20;
```

# 9. Analýza úloh

## 9.1. Statická a časová analýza

Ke statické časové a parametrické analýze nelineárních dynamických soustav i k jejich linearizaci v zadaném nebo vypočítaném pracovním bodě slouží sekce TR programu DYNAST.

Při **statické analýze** program

- v algebro-diferenciálních rovnicích položí všechny derivace podle nezávisle proměnné TIME rovny nule,
- v blokových sehématech ignoruje integrátory a výstupní uzly derivátorů "zkratuje" s referenčním uzlem,
- v branových schématech ignoruje prvky typu C a prvky typu L nahradí "zkratem".

Program pak vypočítá ustálený klidový bod řešení zadané úlohy.

Má-li úloha více různých řešení, to které řešení bude programem nalezeno lze ovlivnit volbou počátečních podmínek řešení. Při výpočtu ustáleného klidového řešení složitějších nelineárních úloh je někdy výhodné statickou analýzu nahradit analýzou parametrickou nebo i časovou.

**Parametrickou analýzou** rozumíme statickou analýzu v závislosti na určitém proměnném parametru. Jejím řešením je trajektorie odpovídajících ustálených klidových bodů.

Při **časové analýze** hledáme řešení v závislosti na nezávisle proměnné TIME pro počáteční podmínky zadané uživatelem nebo získané z předchozích výpočtů.

Analýza v sekci TR se zadává s formátem:

*analýza* [ *proměnná* ] [ *min max* ]

Identifikátor typu analýzy *analýza* se zadává znaky DC, TR nebo DCTR s následujícím významem:

- DC. . . statická nebo parametrická analýza
- TR. . . časová analýza (případně se zadanými počátečními podmínkami),
- DCTR. . . časová analýza, při které si počáteční podmínky určí sám program na základě statické analýzy.

Jako *proměnná* zde může vystupovat identifikátor některé proměnné, která se tak stane proměnnou nezávislou. Pokud tato *proměnná* není zadána, program implicitně předpokládá, že se jedná o nezávisle proměnnou TIME.

*Min* a *max* jsou číselné konstanty, udávající dolní a horní mez intervalu proměnné zadané jako *proměnná*, ve kterém se analýza má provádět. U statické analýzy typu DC lze tyto údaje v zadání vypustit a program pak nalezne klidový pracovní bod analyzované soustavy. Pokud zadáme údaje pro meze *min* a *max*, program provede parametrickou analýzu v závislosti na proměnné zadané jako *proměnná*.

Počáteční podmínky se pro statickou i časovou analýzu zadávají s formátem:

$$INIT\begin{Bmatrix} proměnná \\ V.\,prvek\ C \\ !\,XALL \end{Bmatrix} = hodnota\left[\begin{Bmatrix} proměnná \\ V.\,prvek\ C \end{Bmatrix} = hodnota\ldots\right]$$

*Proměnná* zde značí identifikátor primární proměnné a *prvek* C je identifikátor prvku typu C. !XALL nahrazuje seznam identifikátorů všech primárních proměnných a umožňuje zadání jejich společné hodnoty. *Hodnota* se zadává jako číselná konstanta.

Implicitní hodnota nezadaných počátečních podmínek primárních proměnných je nulová.

**Příklad :**

Počáteční podmínky lze zadat např. jako

INIT X1 = 3, V.3 = 10k, UZEL3 = .01, I . LOAD = 1_A;

Větou

INIT !XALL = 0, Y = 1, Z = 1;

"vynulujeme" všechny hodnoty počátečních podmínek z předchozího výpočtu v současném běhu programu a zvolíme jednotkové počáteční hodnoty proměnných Y a Z.

Analýzu určité soustavy lze v sekci TR libovolněkrát opakovat s různě modifikovanými parametry. Tyto **modifikace parametrů** lze zadat s formátem:

MODIFY *parametr* = *hodnota* [, *parametr* = *hodnota*... ]

kde *parametr* je identifikátor některé nezávisle proměnné nebo branového prvku, *hodnota* se zadává jako číselná konstanta.

**Příklad :**

Modifikaci hodnot parametrů proti jejich dosavadním hodnotám (zadaným např. v sekci SYSTEM) lze zadat třeba jako

MODIFI alpha = 0, R1 = 20.K, TIME = 10U, TEMP = 310_K;

**Zrušit všechny příkazy**, které již byly při posledním vyvolání sekce TR v současném běhu programu zadány a **inicializovat všechna pole** programu pro ukládání výsledků této sekce lze příkazem RESET.

**Tisk semigrafů** výsledných průběhů získaných analýzou v sekci TR se zadává s formátem:

*graf* ( *řádků* [ *nezávislá* ] *min max*) [ / *násobek* ] )
( [ ) *závislá* ( *min max*] (, *závislá* ( *min max*]... [ ) ]...]

Jako identifikátor *graf* označující typ grafu lze zadat:

- PLOT. . . tisk semigrafu o šíři 115 znaků,
- NPLOT. . . tisk semigrafů o implicitní šíři 80 znaků. Požadovanou šíři lze změnit příkazem WPLOT (viz tab. 9.1),
- PPLOT. . . výstup grafu společně s tabulkou (nahrazuje současné užití příkazu PLOT a PRINT),
- NPPLOT. . . výstup úzkého grafu společně s tabulkou (nahrazuje současné použití příkazů NPLOT a PRINT).

*Řádků* je celočíselný údaj určující na kolika řádcích se graf zobrazí.

Jako nezávislou proměnnou grafu můžeme zadat identifikátor *nezávislé* kterékoliv nezávisle či závisle proměnné veličiny analyzované soustavy. Pokud *nezávislou* nezadáme, program implicitně předpokládá, že se jedná o tutéž proměnnou, která byla zadána společně s typem analýzy.

Celočíselnými údaji *min* a *max* můžeme zadat dílčí interval *nezávislé*, pokud nechceme vynášet pro celý interval této proměnné.

Za znakem / lze zadal *násobek*, tj. celočíselnou konstantu, určující **násobky integračního kroku**, v nichž se vynáší body grafu. Je-li touto konstantou 0, body grafu se tisknou ekvidistantně. Je-li tato konstanta větší než 0, rozložení bodů není ekvidistantní, ale je řízeno integračním algoritmem. *Hodnota* vynášených bodů, je pak úměrná rychlosti změn vynášených průběhů. Pokud *násobek* zadáme rovný nule, program na každém řádku vynese pro každou závisle proměnnou grafu právě jeden bod (získaný interpolací výsledných bodů integrace). Implicitně je zadáno / 1, to zn. že program zobrazuje závisle proměnné grafu v každém integračním kroku.

Za každou závisle proměnnou grafu s identifikátorem *závislá* lze zadat její minimální a maximální hodnotu údaji *min* a *max* určujícími rozsah závislé proměnné grafu. Jsou-li tyto údaje vynechány, program si pro danou proměnnou určí rozsah automaticky tak, aby jej nepřekročila.

Pokud je několik identifikátorů *závislá* zadáno v závorkách ( ), tyto veličiny se vynesou se společným rozsahem. Tento rozsah je určen buď údaji *min max* první veličiny, u níž jsou zadány. Nejsou-li zadány, program si **společný rozsah** volí automaticky tak, aby jej žádná z veličin nepřekročila. Závorkami ( ) může být v zadání grafu označeno i několik skupin identifikátorů *závislá*. Každá skupina se pak vynese s vlastním společným měřítkem.

V jednom grafu lze současně vynášet průběhy až pro 15 závisle proměnných grafu. V jednom běhu programu je možno zadat nejvýše 4 grafy.

**Příklad:**

Tisk úzkého grafu průběhu průtoku a spádu prvku L ve 100 bodech s automatickou volbou měřítka lze zadat jako

NPLOT (100) I .L, V.L;,

tisk *širokého grafu* závislosti spádu prvku L na jeho průtoku s automatickou volbou měřítek v 50 bodech jako

PLOT (50 I . L) V . L ;,

tisk grafu v každém integračním kroku a tabulky na 100 řádcích pro hodnotu prvku RTERMIST s automatickou volbou měřítka a současně pro průtoky prvků RI 1 a RI 2 se společným měřítkem v intervalu od -1 mA do 1 mA v závislosti na teplotě lze zadat jako

PPLOT (100 TEMP/1) RTERMIST, (I .RI 1, I .RI2 -1M 1M);

**Tisk tabulek** výsledných hodnot analýzy se zadává s formátem:

PRINT [( *řádků* ) ] *závislá* ,[ *závislá* . . . ]

Význam údajů je zde shodný jako při zadávání grafů s tím, že nezávisle proměnnou tabulky je vždy proměnná zadaná současně s typem analýzy. Volba měřítek je zde ovšem zbytečná.

Tab.9.1: Parametry pro řízení výpočtu

| Parametr | Význam | Implicitně |
|---|---|---|
| EPS | přípustná relativní chyba řešení | $10^{-3}$ |
| DCEPS | přípustná relativní chyba statické analýzy | $10^{-6}$ |
| MAXIT | přípustný počet iterací v jednom integračním kroku | 50 |
| DAMP | činitel tlumení iterací | 0 |
| FLUF | konstanta řízení rozkladu LU | 1 |
| MIN | relativní délka přípustného nejkratšího integračního kroku | $10^{5}$ |
| MAX | relativní délka přípustného nejdelšího integračního kroku | 10 |
| FEPS | konstanta řízení predikce | 5 |
| KMAX | přípustný nejvyšší řád predikce | 6 |
| C1 až C6 | činitelé tlumení predikce řádu 1 až 6 | 1 |
| WPLOT | počet znaků na řádce při stisku semigrafu příkazem NPPLOT | 80 |

Pokud uživatel při zadávání tisku tabulky nezadá údaj *řádků*, tabulka se tiskne v každém integračním kroku.

**Příklady:**

Tisk tabulky spádu uzlu 3 a jeho derivace v každém integračním kroku se zadá jako

PRINT V.3, VD.3;

Spuštění výpočtu se provádí příkazem RUN s formátem:

RUN [ HOLD ] [ *parametr* = *hodnota* [, *parametr* = *hodnota* . . . ] ]

kde *parametr* představuje identifikátor některého z **parametrů pro řízení výpočtu** uvedeného v tab. 9.1. *Hodnota* parametru se zadává jako číselná konstanta. Pro parametry, které nejsou zadány, program bere v úvahu jejich implicitní hodnoty.

Užijeme-li za příkazem RUN příkaz HOLD, k tisku grafu nedojde (vytiskne se pouze tabulka, je-li požadována), a1e výsledky se uloží do vnější paměti počítače. To nám dává možnost současného vynášení průběhů první veličiny zadané jako *závislá* pro různě modifikované parametry analyzované soustavy, a to do jednoho grafu. Graf se vynese až poté, kdy uživatel zadá příkaz RUN bez příkazu HOLD.

## 9.2. Výpočet operátorových funkcí

K výpočtu **semisymbolických operátorových funkcí** lineárních blokových nebo branových schémat, příp. schémat linearizovaných v určitém pracovním bodě sekcí TR, slouží sekce PZ. Schémata přitom mohou být kombinována s rovnicemi.

Uvedené operátorové funkce mohou představovat **přenosové funkce**, **obrazy odezev** na počáteční podmínky nebo obrazy odezev na budící signály s racionálním Laplaceovým obrazem.

**Semisymbolickou operátorovou funkcí** zde rozumíme racionální funkci ve tvaru

$$F(p) = K\frac{(p-z_1)(p-z_2)...}{(p-p_1)(p-p_2)...} = \frac{a_0 + a_1 p + a_2 p^2 + ...}{b_0 + b_1 p + b_2 p^2 + ...}$$

se symbolicky vyjádřeným Laplaceovým operátorem *p* a s číselně vyjádřenými kořeny jejích polynomů $z_1$, $z_2$, . . . (tj. nulami), $p_1$, $p_2$, . . . (tj. póly) a násobnou konstantou K nebo koeficienty jejích polynomů $a_0$, $a_1$, $a_2$ . . . , $b_0$, $b_1$, $b_2$, . . ..

Jak známo, představuje-li operátorová funkce F(p) přenosovou funkci blokového nebo branového schématu, platí

$$F(p) = \frac{Y(p)}{Z(p)}$$

kde Y(p) je Laplaceův obraz odezvy *y(t)* určité proměnné veličiny schématu v závislosti na jiné jeho proměnné veličině *z(t)* s obrazem *Z(p)*, a to při nulových počátečních podmínkách. Představuje-li funkce F(p) obraz odezvy určité veličiny *y(t)*, pak prostě

$$F(p) = Y(p)$$

Požadované operátorové funkce se zadávají s formátem:

$$\text{TRAN}\ \mathit{funkce}\left[ = \mathit{výstup} / \left\{ \begin{matrix} \mathit{zdroj} \\ \text{INIT} \end{matrix} \right\} \right][\text{COEF}]\left[ , \mathit{funkce}\left[ = \mathit{výstup} / \left\{ \begin{matrix} \mathit{zdroj} \\ \text{INIT} \end{matrix} \right\} \right][\text{COEF}]... \right]$$

*Funkce* je identifikátor požadované operátorové funkce. Jako *výstup* zde lze zadat

- identifikátor některé primární proměnné,
- V.*prvek Y,* tj. spád prvku třídy Y,
- l.*uzel* - 2.*uzel*, tj. rozdíl spádu mezi dvěma uzly.

Za znakem / se v případě přenosové funkce jako *zdroj* zadává

- identifikátor některého prvku typu BS, E nebo J, který již byl zadán v sekci SYSTEM,
- BS.*uzel* tj. blok typu BS jako zdroj signálu dodatečně připojený k určitému uzlu,
- J.*prvek* Y, tj. zdroj průtoku dodatečně zařazený k určitému prvku třídy Y paralelně,
- E.*prvek* Z, tj. zdroj spádu dodatečně zařazený k určitému prvku třídy Z do serie.

Při výpočtu každé z požadovaných přenosových funkcí program vždy všechny ostatní budící zdroje i všechny počáteční podmínky považuje za nulové.

V případě výpočtu obrazu odezvy na počáteční stav se místo identifikátoru *zdroj* zadává příkaz INIT.

**Počáteční podmínky** se zde zadávají s formátem:

INIT *proměnná* = *hodnota* [, *proměnná* = *hodnota*. . . ]

Jako proměnná se zde zadává:

- V.*blok* BI, tj. počáteční stav integrátoru,
- V.*prvek* C, tj. počáteční spád prvku typu C,
- I.*prvek* L, tj. počáteční průtok prvku typu L.

*Hodnota* se zadává jako číselná konstanta.

Při výpočtu obrazu určité odezvy schématu na zadané počáteční podmínky program považuje všechny budící zdroje za nulové. Implicitně za nulový považuje i počáteční stav všech prvků typu BI, C a L, pokud nebyl zadán.

Požaduje-li se pro zadávanou operátorovou funkci vedle výpočtu kořenů jejích polynomů a násobné konstanty i **výpočet koeficientů polynomů**, zadá se za její specifikaci příkaz COEF (pozor - u některých úloh to může způsobit "přetečení" registru počítače).

Pokud uvažovaná přenosová funkce již byla v sekci SYSTEM zadána jako přenos některého bloku typu BT, pro její specifikaci stačí uvést pouze identifikátor tohoto bloku.

**Příklady:**

Tak požadované operátorové funkce lze zadat třeba jako

```
TRAN T = I .RIOUT/EIN COEF, INIRESP = V.LOAD/INIT, BT13=BT13;
INIT V.C4 =  .5; I .LO = 1.2M, V.BI = 1;
```

Uvedené nenulové počáteční podmínky budou uvažovány pouze při výpočtu operátorové funkce INIRESP.

## 9.3. Časová analýza operátorových funkcí

K numerickému i semisymbolickému výpočtu časových průběhů impulsních a přechodových charakteristik i odezev na budicí signály s racionálním Laplaceovým obrazem a na počáteční stav či jejich kombinace slouží sekce TRA. Tato sekce k tomu využívá operátorové funkce soustav vypočítané prostřednictvím sekce PZ.

Čas se zadává bud s formátem:

TIME *min*, *max* [, *bodů*]

nebo s formátem:

TIME =$t_1$, $t_2$, . . .

V prvním případě se jako číselný údaj *min* a *max* zadává dolní a horní mez časového intervalu. Celočíselným údajem *bodů* se zadává počet diskrétních časových bodů. Ve druhém případě se údaji $t_1$, $t_2$, . . . zadávají přímo diskrétní časové body. Tyto údaje jsou číselné konstanty. Jejich požadovaný počet se zadává celočíselným údajem *bodů*.

Není-li identifikátor TIME zadán, program si horní mez časového intervalu zvolí automaticky jako násobek nejdelší časové konstanty zadaných operátorových funkcí, dolní mez zvolí nulovou.

**Zobrazování semigrafů** výsledných časových průběhů se zadává s formátem:

*graf* ( *řádků*) [ STEP.] *funkce*[[& *funkce*] . . . [+ *funkce* ]]  [,[STEP.] *funkce* [& *funkce* . . .]]

Typ grafu graf lze zadat shodně jako v sekci TR. Příkazem STEP se volí výpočet **přechodové charakteristiky**. Není-li tento příkaz zadán, program počítá **charakteristiku impulsní**.

*Funkce* značí identifikátor operátorové funkce získané semisymbolickou analýzou v sekci PZ. Znakem & se označuje součin operátorových funkcí, znakem + součet jejich časových odezev.

Měřítka vynášených časových průběhů lze zadávat obdobně jako v sekci TR. Ve výše uvedeném popisu formátu se pro přehlednost neuvažují.

**Příklady:**

Zobrazení širokého semigrafu o 100 řádcích pro přechodové charakteristiky přenosových funkcí TF1 a TF2 se společným měřítkem zvoleným automaticky se zadá jako

```
PLOT (100) (STEP.TF1,  STEP.TF2);
```

zobrazení úzkého semigrafu o 70 řádcích impulsní charakteristiky kaskády tří bloků s průnosem BTF1, BTF2 a BTF3 s rozsahem 0 až 10 se zadá jako

```
NPLOT (70) BTFl & BTF2 & BTF3 0 10;
```

Výsledkem dalšího zadání je široký semigraf a tabulka o 100 řádcích pro součet odezvy soustavy  s přenosem PRENOS na signál s racionálním obrazem SIGNAL a odezvy této soustavy na její počáteční stav, jejíž obraz je označen identifikátorem POCSTAV. Volba měřítek se provádí automaticky.

```
PPLOT (100) SIGNAL & PRENOS + POCSTAV;
```

**Zobrazení tabulek** časových odezev nebo jejich kombinací se zadává s formátem:

PRINT [[( *řádků*)] [ STEP.] *funkce* [& *funkce*& . . . +
[*funkce*] [ ,[STEP.]*funkce*[& *funkce* . . .]]]

Počet řádek se zadává obdobně jako při tisku tabulek v sekci TR. *Operátorové funkce* a jejich kombinace se zadávají obdobně jako při vynášení grafu. Zadáme-li pouze příkaz PRINT, program vytiskne tabulku všech odezev požadovaných v předcházejícím zadání grafu:

**Příklad:**

Výsledkem zadání

PRINT (10) TF, STEP.TF + TF INIT;

je tabulka o 10 řádcích pro impulsní charakteristiku přenosu TF a pro součet přechodové charakteristiky tohoto přenosu s příslušnou odezvou na počáteční stav, jejíž obraz je označen jako TFINIT.

Vedle grafů a tabulek lze v sekci TRA tisknout požadované **časové odezvy** nebo jejich kombinace v **semisymbolickém tvaru**, tj. s časem označeným jako T a s exponenciálními i goniometrickými funkcemi vyjádřenými symbolicky a s koeficienty numerickými. Tisk semisymbolických odezev lze zadat s formátem:

SYMB [ STEP.] *funkce* [& *funkce* . . . [+ funkce]] [, [STEP.] *funkce*[& *funkce* . . . ]]

Specifikace *funkcí* je obdobná jako při zadávání grafů.

**Výpočet se spouští** příkazem:

RUN

## 9.4. Kmitočtová analýza přenosových funkcí

K výpočtu různých složek kmitočtových charakteristik semisymbolicky vyjádřených racionálních přenosových funkcí s póly, nulami a násobnými konstantami vypočítanými v sekci PZ slouží sekce FRE.

Kmitočet se zadává buď s formátem:

FREQ [/ [LIN]] *min*, *max* [, *bodů*]

nebo s formátem:

FREQ = $f_1$, $f_2$, . . .

V prvém případě se údaji *min* a *max* zadává dolní a horní mez kmitočtového rozsahu, údajem *bodů* se zadává počet diskrétních kmitočtových bodů. Ve druhém případě, se údaji $f_1$, $f_2$, . . . zadávají přímo diskrétní kmitočtové body. Všechny tyto údaje jsou číselné konstanty.

Při zadání kmitočtového rozsahu lze příkazem / LIN, případně jen znakem /, zvolit lineární **měřítko kmitočtové osy**. Implicitně program volí měřítko logaritmické.

Není-li příkaz FREQ zadán, program si jeho rozsah zvolí automaticky jako násobek nejnižšího a nejvyššího kritického kmitočtu zadaných přenosových funkcí.

**Vynášení grafů** výsledných průběhů složek kmitočtových charakteristik se zadává s formátem:

*graf* ( *řádků* [ *nezávislá* ] ) [( ] *závislá* [ *min max* ] [, *závislá* [ *min max* ]. . . [ )] . . . ]

Typ grafu *graf* lze zadat shodně jako v sekci TR. Není-1i *nezávislá*, tj. identifikátor nezávisle proměnné zadán, program implicitně předpokládá, že se jedná o proměnnou FREQ (kmitočet). Jinak identifikátory *nezávislá* a *závislá* představují některou **složku kmitočtové charakteristiky** příslušné přenosové funkce. Tyto proměnné se zadávají s formátem:

*složka.funkce*

Typ složky kmitočtové charakteristiky se zadává identifikátorem složka v souladu s tab. 9.2. Identifikátor přenosové funkce se musí shodovat s identifikátorem přiřazeným uvažované funkci již při jejím výpočtu v sekci PZ.

Při jednom spuštění sekce FRE je možno zadat pouze jeden graf. V případě, že kmitočet FREQ byl zadán pouze v diskrétních bodech, příkaz pro tisk grafu je ignorován.

**Příklad:**

Zadáním

PPLOT (40) (DB.TF1 -3 6, DB.TF2), DEG.BTRA;

dosáhneme tisk širokého grafu a tabulky o 40 řádcích pro modul přenosů TF1 a TF2 v dB se společným měřítkem v intervalu -3 až +6 dB a současně tisk fázové charakteristiky bloku BTRA ve stupních s automatickou volbou měřítka, zadáním

PLOT (20 RE .TF2) IM.TF2;

tisk širokého grafu o 20 řadách pro kmitočtovou charakteristiku přenosu TF2 v komplexní rovině

Tab. 9.2: Složky kmitočtové charakteristiky

| Typ | Význam | Typ | Význam |
|---|---|---|---|
| MOD | modul | DEL | skupinová zpoždění |
| DB | modul v dB | SLO | strmost modulu |
| RAD | fáze v radiánech | RE | reálná část |
| DEG | fáze ve stupních | IM | imaginární část |

**Tisk tabulek** numerických hodnot složek kmitočtových charakteristik přenosových funkcí se zadává s formátem:

PRINT [ ( *řádků* ) ] ) *závislá* [, *závislá* . . . ]

Závisle proměnné se zadávají obdobně jako při zadávání grafu. Pro zadávání počtu řádek tabulky platí obdobný předpis jako v sekci TR.

**Příklady:**

Důsledkem zadání

PRINT (60) MOD.ADMIT, MOD.IMPED

je tisk tabulky modulů přenosů ADMIT a IMPED o 60 řádcích.

**Výpočet se spouští** příkazem:

RUN

# Literatura


[1] Mann,H.: *Využití počítačů při elektrotechnických návrzích*. SNTL, Praha 1984

[2] Čajka,J. Mann,H. Oliva,Z.: *Počítačové řešení elektronických obvodů (DAVID4)*. Doplňkové skriptum FEL ČVUT, Praha 1983

[3]DAVID-3 *program pro komplexní analýzu nelineárních a lineárních analogových spínaných a impulsních soustav*. Kap. 4 ve skriptu Čajka,J. Mann,H.: *Počítačové řešení elektronických obvodů*. skriptum FE VUT, Brno 1983.

[4]Mann,H. Boreš,O. Oliva,Z.: *DAVID2 uživatelský návod*. Příloha ke skriptu Mikulec,M.:*Teorie obvodů*. Skriptum ČVUT, Praha 1981

[5]Oliva,Z.:*Počítačové metody jmenovité analýzy nelineárních dynamických soustav*. Práce k odborné kandidátské zkoušce. FEL ČVUT, Praha 1982

[6]Mann,H. Oliva,Z.: *DAVID4 program pro elektrickou a funkční simulaci analogových, číslicových a spínaných obvodů*. Sborník FMEP-TESLA VÚST Návrh obvodů počítačem, Praha 1982, str. 9-19

[7] Mann,H.: *NAP2 program pro stejnosměrnou, časovou a kmitočtovou analýzu a optimalizaci nelineárních elektronických obvodů*. Sborník ZP ČSVTS, TESLA VÚST, Praha 1975

[8] Rübner-Peterson,T.: *ALGDIF a FORTRAN IV subroutine for solution and perturbated solutions of algebraic-diferential equations*. Zpráva, Dánská technická universita, Lyngby 1979

[9] Rübner-Peterson,T.: *SFORM1 and SFORM2 two FORTRAN IV subroutines for sparse matrix transformation of the general eigenproblem to standard form*. Zpráva IT-41, Dánská technická univessita, Lyngby 1979

[10] Mann,H.: *Modifikovaná metoda uzlových napětí*. Slaboproudý obzor 41 (1980), str.177-181

[11] Mann,H.: *Blokové diagramy a elektrické obvody*. Slaboproudý obzor 41 (1983), str.581-587

[12] Mann,H.: *Postup pro semisymbolickou analýzu lineárních dynamických soustav*. Elektrotechnický obzor 71 (1982), str. 634-639

[13] Sommer,T.: *Program pro výpočet jakobiánů a citlivostí nelineárních dynamických soustav symbolickým derivováním*. Práce SVOČ, FEL, ČVUT, Praha 1985

[14] Oliva Z.: *Některé algoritmy analýzy elektronických obvodů*. Kandidátská disertační práce, FEL, ČVUT, Praha 1985

[15] Mann.H. a kol.: *Automatizace projektování dynamických soustav*. Sborník, Dům techniky ČSVTS, Praha 1986