1. Laborgyakorlat

A MATLAB-SIMULINK KÖRNYEZET S-FUNCTION LEHETŐSÉGEI

1.1. A dolgozat célja:

A dolgozatban röviden ismertetjük a Matlab-Simulink környezetben használt „S-function" lehetőségeket. A dolgozat nem kimondottan az adaptív szabályozási témakörhöz tartozik, viszont az S-függvényeknek az ismerete segít a különböző algoritmusoknak a tesztelésében, azaz a Simulink környezetben való megvalósításoknál.

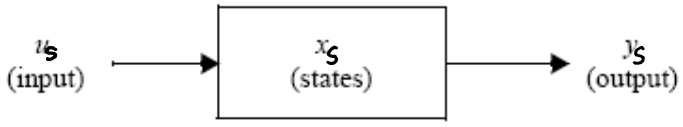
1.2. Elméleti bevezető:

Az S-function használata lehetővé teszi a saját algoritmusunknak a hozzáadását egy Simulink ábrához. Az algoritmust megírhatjuk Matlab vagy C környezetben is. Jelen dolgozatban csak az első változatot (azaz a Matlabos változatot) tárgyaljuk.

Az S függvénynek van egy jól meghatározott módozata, mely megengedi a különböző Matlab által rendelkezésünkre bocsájtott megoldási módszerek elérését. Az S függvényekkel megoldhatók az ú.n. folytonos differenciálegyenlet rendszerek, a diszkrét differencia egyenletrendszerek illetve más bonyolult algebrai kifejezések meghatározása is lehetséges. Mint ismeretes ezeknek a differencia és differenciál egyenletek lineáris változatainak a megoldásai megtalálhatók a különböző Simulink könyvtárakban megjelenő objektumok által, viszont a nemlineáris megoldásokhoz már nincs egyértelműen kidolgozott lehetőség erre. Főleg a rekurzivitás kérdése oldható meg könnyen ezekkel az S-függvényekkel.

Az elkövetkezőkben csak az S függvényekkel kapcsolatosan alapismeretek kerülnek bemutatásra, bővebb információkat a „Matlab help" vagy az erre a célra kiadott szakkönyvekben lehet találni.

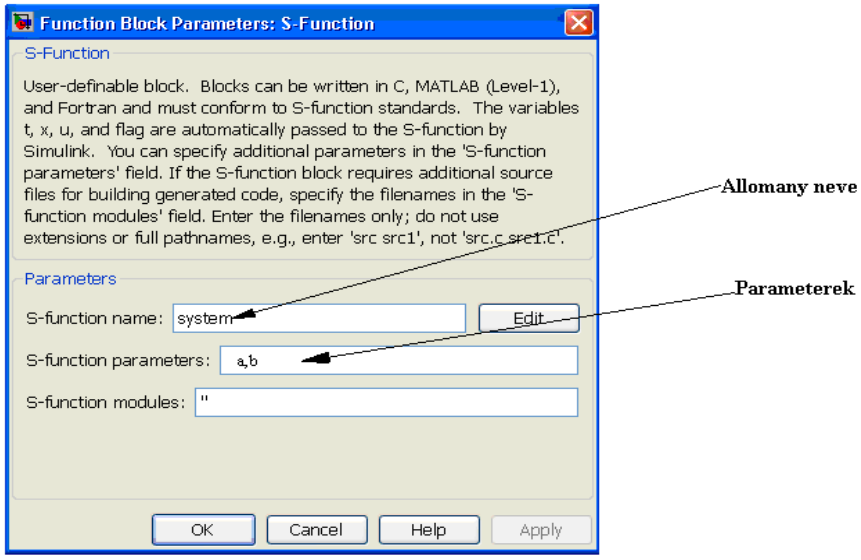
Egy S-function általános tömbdiagramja a rendszerelméletből ismert forma szerint a 1.1 ábrán van megjelenítve, ahol u_s a bemenetek vektora, y_s a kimenetek vektora és x_s az állapotvektor (folytonos vagy diszkrét változat).



1.1.ábra S-function általános tömbdiagram

A Simulink modellben az S-függvény beillesztése a Simulink könyvtárban (Simulink Library Browser) található User-Defined Functions alkönyvtárban előre elkészített S-function modulon keresztül történik. Az így bemásolt modul paramétere

az 1.2. ábrán megjelenített ablakot jeleníti meg. Kötelező módon be kell írnunk az állomány nevét, ami tartalmazza majd a megoldani kívánt algoritmusunkat. A másik két mező már opcionális, csak akkor használjuk, ha az algoritmusunk szükségelteti bizonyos többlet paraméterek illetve modulok használatát.



1.2 ábra S-function modul paraméter ablaka

Egy Matlab S-function általános szintaxisa a következő:

function [sys, x0 , str , ts] = system(t, xs, us, flag)

Az itt megjelenített ki és bemenő argumentumokat sem sorrendben sem elnevezésben nem ajánlott módosítani. Az állomány neve (pl. system) változtatható viszont fontos, hogy a Simulink modell és az itt használt állomány ugyanabban a mappában kell legyen elmentve és a Matlab környezetnek meg kell adni majd ezt az útvonalat.

A ***t, u***s (idő és bemenő jel) bemenő argumentumokkal a Simulink automatikusan változtatja, az ***xs***-et (állapotváltozó) pedig a ***flag*** változó értelmében mindig aktualizálja az általunk megadott algoritmus szerint.

A flag változóval állítjuk be azt, hogy milyen feladat megoldását várjuk el az S függvénytől. Az 1.3. ábrán megjelenített táblázatban láthatók ennek a lehetséges értékei:

Elnevezés	**Függvény**	**Flag**
Inicializálás	mdlInitializeSizes	0
Deriváltak számítása	mdlDerivatives	1
Diszkrét állapotok aktualizálása	mdlUpdate	2
Kimenetek számítása	mdlOutputs	3
A következő mintavétel taszk (sample hit) számítása (opcionális)	mdlGetTimeOfNextVarHit	4
A szimulációs taszk megállítása	mdlTerminate	9

1.3.ábra. Az S-function flag változóinak lehetséges értékei

Az itt szereplő függvényeknek is megvannak a sajátos alakja. A függvények elnevezése megváltoztatható, de az argumentumok (sorrendje) nem.

Az inicializálásnál a következő paranccsal

sys = simsizes(sizes)

elvileg létrehozunk egy „dimenzió” struktúrát. Az 1.4. ábrán megjelenő táblázatban feltüntetjük, hogy milyen információ mezőket adhatunk meg.

Mező neve	**Leírás**
sizes.NumContStates	Folytonos állapotok száma
sizes.NumDiscStates	Diszkrét állapotok száma
sizes.NumOutputs	Kimenetek száma
sizes.NumInputs	Bemenetek száma
sizes.NumSampleTimes	Mintavételek száma

1.4.ábra. Az S-function size szerkezete

Ha pl.. egy nagyon egyszerű S-függvényt írunk, ami kiszámolja a bemenő jel abszolút értékének a logaritmusát, akkor elég, ha csak az inicializálás és kimenet függvényeket használjuk. Az ennek megfelelő függvény az 1.5 ábrán lévő kis példa programmal valósítható meg.

```
function [sys, x0, str, ts] = szamol(t,xs,us,flag)
switch flag,
        case 0
           [sys,x0,str,ts] = mdlInitializeSizes;  % Initialization
        case 3
            sys = mdlOutputs(t,xs,us);              % Calculate outputs
        case { 1, 2, 4, 9 }
             sys = []; % Unused flags
        otherwise
              error(['Unhandled flag = ',num2str(flag)]);  % Error handling
        end;

function [sys,x0,str,ts] = mdlInitializeSizes
        sizes = simsizes;
        sizes.NumContStates= 0;
        sizes.NumDiscStates= 0;
        sizes.NumOutputs= 1;
        sizes.NumInputs= 1;
        sizes.DirFeedthrough=1;
        sizes.NumSampleTimes=1;
        sys = simsizes(sizes);
        x0 = []; % No continuous states
        str = []; % No state ordering
        ts = [-1 0]; % Inherited sample time

function sys = mdlOutputs(t,xs,us)
        sys = log(abs(us));
```

1.5. ábra Példaprogram egy egyszerű S function megvalósításhoz

Ha a tömbhöz paramétereket is szeretnénk rendelni (lásd 1.2 ábra), akkor ezeket a paramétereket az S-függvénynél a flag bemenő argumentuma után helyezzük el. Ezeknek a felhasználását pedig a felhasználó feladata feldolgozni az adott függvényekben.

function [sys,x0,str,ts] = szamol (t, xs, us, flag, a, b)

Ahhoz, hogy könnyebben tudjunk S-függvényeket írni a laborgyakorlathoz csatolva van egy elvileg egyszerű állomány, melynek a tanulmányozásával könnyen megérthetjük ezeknek a működését.

1.3. A munka menete:

Használva a csatolt példa programot, végezzék el a következő feladatokat:

- Készítsünk egy Simulink modellt, amely meghívja az adott állományt és derítsük ki, hogy milyen műveleteket végez el a megadott állomány.
- Paraméterezzük le ezeket az S-függvényeket és maszkolással adjunk a paramétereknek értékeket, majd teszteljük ezeket a változatokat is.
- Oldjuk meg a következő feladatokat S-functiont használva és ábrázoljuk a modulok kimeneteit használva tetszőleges bemeneteket.

Algebrai egyenlet:

1. $\begin{cases} y_1(t) = 3 \cdot u_1(t) - 4 \cdot u_2(t); \\ y_2(t) = 3 \cdot u_1(t) \cdot u_2(t) - 0.3 \cdot u_3(t)^2; \end{cases}$

Differenciál egyenlet

2. $\begin{cases} \dot{x}(t) = \begin{pmatrix} -1 & 0.02 \\ 0.01 & -3 \end{pmatrix} \cdot x(t) + \begin{pmatrix} 3 \\ 2 \end{pmatrix} \cdot u(t); \\ y(t) = \begin{pmatrix} 1 & 1 \end{pmatrix} \cdot x(t) + 2 \cdot u(t); \end{cases}$

Differencia egyenlet:

3. $\begin{cases} z_1[k+1] = -0.6 \cdot z_1[k] - c \cdot u[k]; \\ z_2[k+1] = 0.02 \cdot z_1[k] - 0.8 \cdot z_2[k] + d \cdot u[k] \cdot z_1[k]; \\ y[k] = z_1[k] + 2 \cdot z_2[k] \end{cases}$ megj. c, d – paraméterek

1.4. Kérdések:

1. Hogyan módosítanánk meg az S modulok kezdeti állapotait?
2. Mikor nem használunk ú.n. „mdlDerivatives” függvényeket?
3. Mi történik, ha nem használunk „mdlOutputs” függvényt?
4. Mire használjuk a „sizes.DirFeedthrough” flag-et?