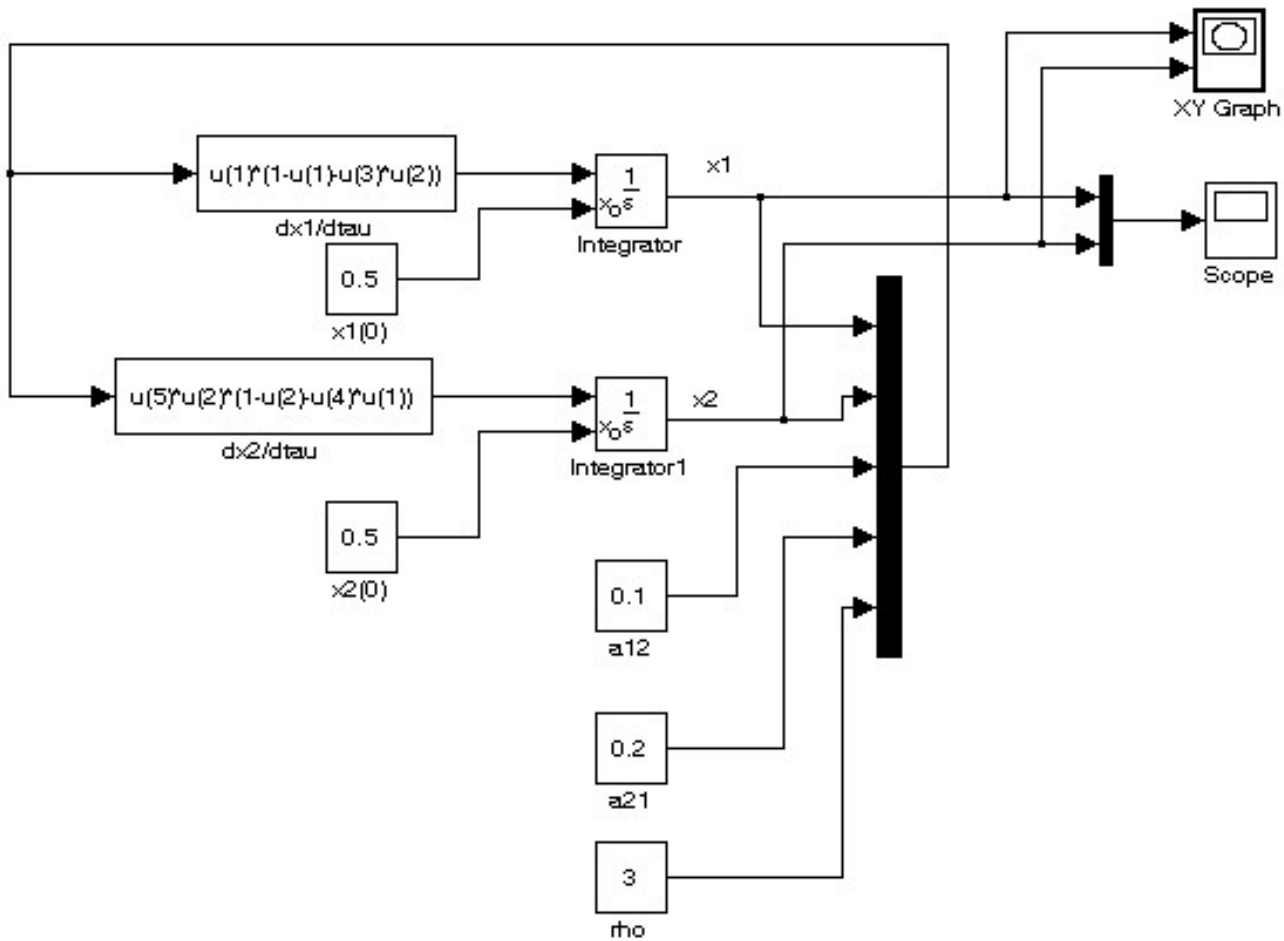
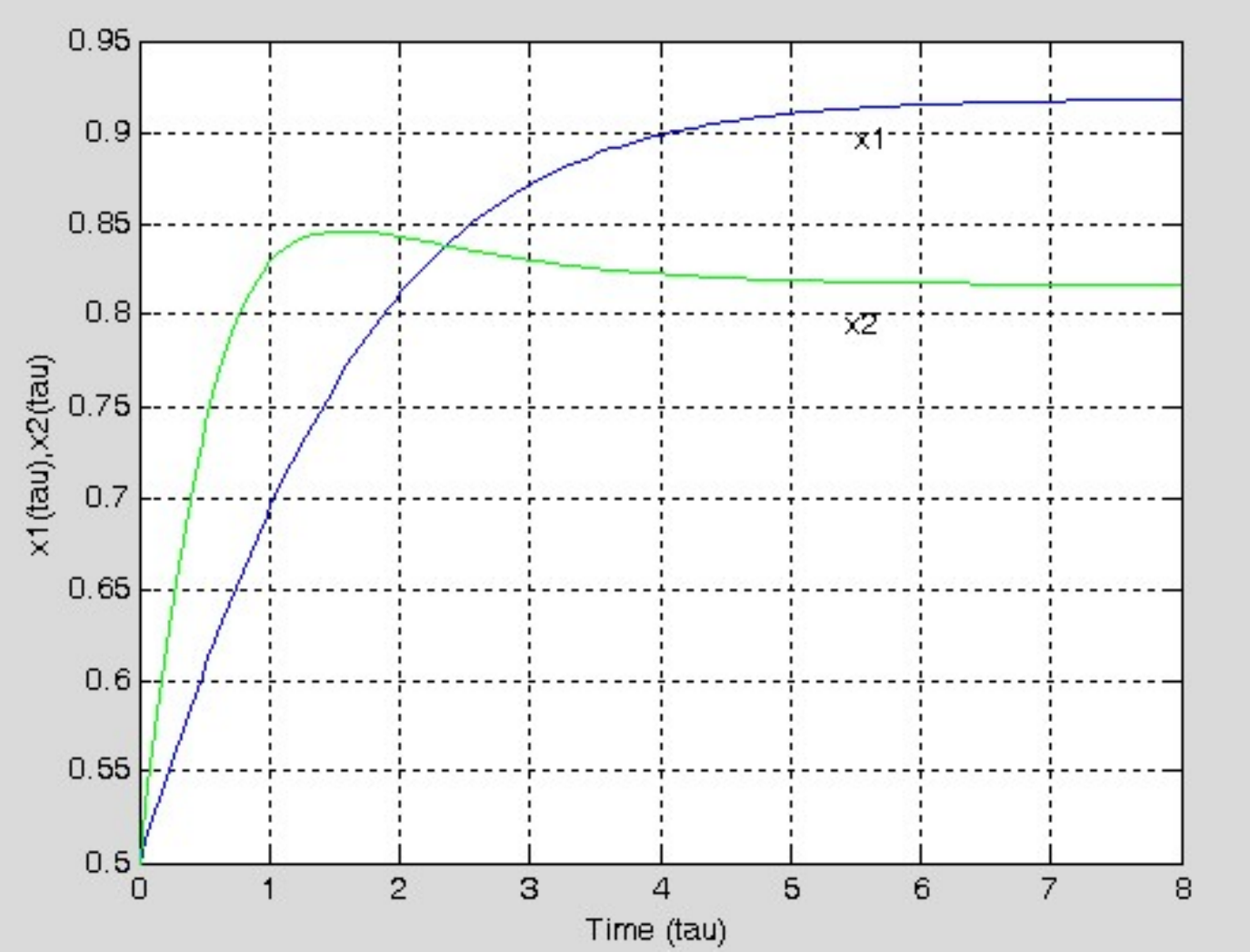
BME 5742 Bio-Systems Modeling and Control

Lecture 14
Matlab & Simulink Tutorial 4
Linearization using Matlab and Simulink

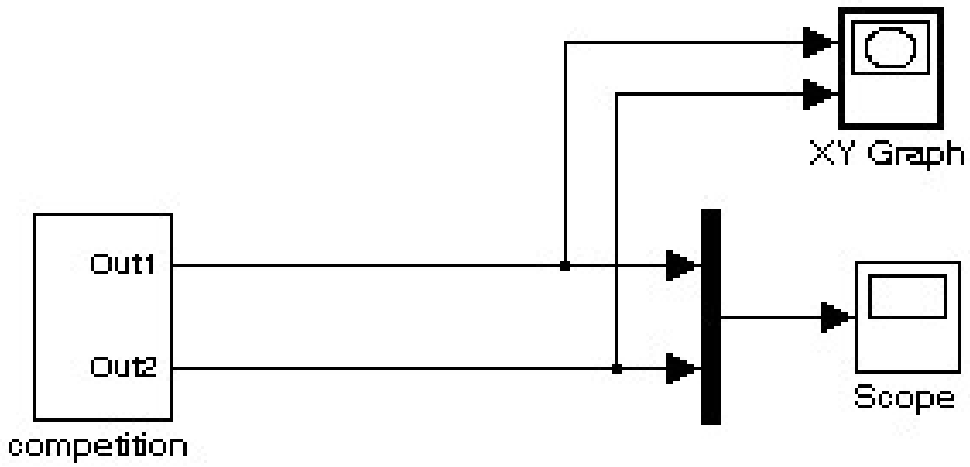
Two-species competition model



Non-aggressive competition example



Enclosing the whole system as a subsystem entitled “competition”



Matlab’s equilibrium finding and linearization tools require that such subsystem creation is done.

How do we create a Simulink model's sub-system?

- Look at the model and decide which part of it you wish to convert into a subsystem
- Start at any point near the model (typically somewhere on the upper left corner of your intended subsystem-surrounding rectangle)
- Draw a rectangle that surrounds the intended subsystem. It basically click-selects this part of the model.
- In Edit select "Create Subsystem".

Comments about Subsystems

- Once you create a subsystem it is an irreversible action.
- Therefore, before creating a subsystem, it is recommended to keep a copy of the original model, under a different name.
- The creation of subsystems essentially creates a hierarchy in the model.
- By double-clicking on the subsystem block you can see the model included inside.

```
>> [sizes,x0,states]=Bio_competition2_012304([],[],[],0)

sizes =

     2
     0
     0
     0
     0
     0
     2


x0 =

    0.5000
    0.5000


states =

    'Bio_competition2_012304/competition/Integrator'
    'Bio_competition2_012304/competition/Integrator1'
```

A Model command

Number of continuous states

Program's internal names for initial conditions and states

The Trim Command

- Basic syntax:

  trim(‘model-name’,[a;b])
- [a;b] (where a and b are given numerical values provided by the user) is an arbitrary point in the state plane.
- Matlab starts the search at an initial guess [a;b] (provided by the user) and stops when <u>one of the equilibria</u> is found.
- Need to start search at various initial guesses, if we hope to find <u>all</u> system equilibrium points.

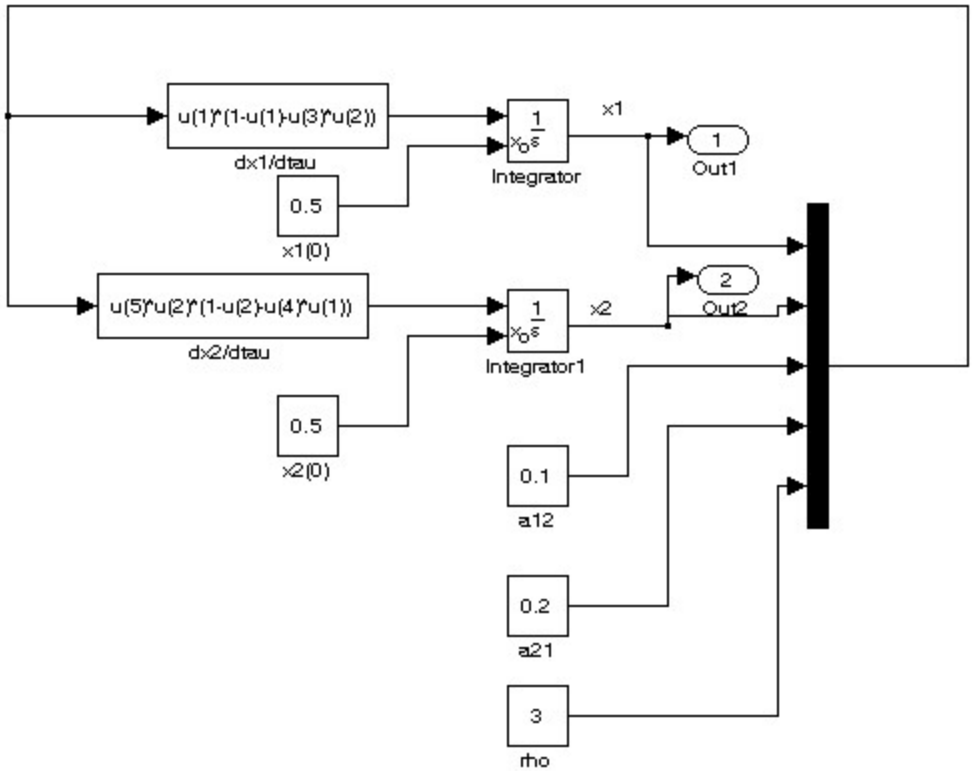
For $a_{12}=a_{21}=0.1$, here is the result of a trim command:

```
xeq=trim('Bio_competition2_012304',[0.9;0.9])
```

```
>>
xeq =

    0.0000
    1.0000
```

```
for x1guess=0:0.5:1.5
    for x2guess=0:0.5:1.5
        xeq=trim('Bio_competition2_012304',[x1guess;x2guess])
    end
end
```



In this case there are four equilibria. Above program lists many identical results, four of which are different

```
xeq =

    0.0000
    1.0000


xeq =

  1.0e-010 *

   -0.2574
    0.0000


xeq =

    1.0000
   -0.0000


xeq =

    0.9184
    0.8163


xeq =

    0.9184
    0.8163


xeq =

     1
     0
```

Sample results

This is, for all practical purposes, the (0,0) equilibrium

This is part of a long list. All 4 equilibrium points happen to be shown in this results segment.

Comments about Trim

- The solver program initiated by Trim converges to one of the equilibrium points. Which one – depends on the initial guess.
- A more elaborate m file can be programmed, so that only different solutions will be listed.
- How do we know if an equilibrium point is stable or not? → Need the Linmod command.

The Linmod Command

- At the most basic level, for autonomous (i.e. systems with no external input signal) nth-order nonlinear model with an equilibrium state vector x_e, a <u>linearized</u> model $d(\Delta x)/dt=A(\Delta x)$, with A being a nxn matrix, is obtained using the command:
- A=linmod('model-name',xeq)
- Prior to using linmod use xeq=trim('model-name') to find an equilibrium state.
- Let us first show the mechanics of doing it, and then we'll discuss the meaning of Linearization.

Linmodv5 applied from the command line

Overall view of the entire interactive dialog.

We'll zoom in on segments of the code.

Linmod (of Matlab 6) may have a bug. It looks as if Linmodv5 (of Matlab 5) works right. Matlab 7 ???

```
>> xeq=trim('Bio_competition2_012304',[1;1])

xeq =

    0.9184
    0.8163

>> A=linmodv5('Bio_competition2_012304',xeq);
>> A

A =

  1.0e+004 *

   -2.2480   -2.2482
   -5.9945   -5.9954

>> eig(A)

ans =

  1.0e+004 *

   -0.0001
   -8.2433
```

First identify a point to linearize about, typically an equilibrium

```
>> xeq=trim('Bio_competition2_012304',[1;1])

xeq =

    0.9184
    0.8163
```

- The above is an equilibrium found near [1;1]
- Linmod can linearize the model about any vector value (not necessarily an equilibrium).

Linearize model about equilibrium point found earlier:

```
>> A=linmodv5('Bio_competition2_012304',xeq);
>> A

A =

   1.0e+004 *

    -2.2480   -2.2482
    -5.9945   -5.9954
```

Meaning of A: d(Δx)/dt=A(Δx), where Δx=x-xeq, where x is the vector [x1;x2]. Presumably, Δx is sufficiently small.

Is xeq=[0.9184;0.8163] stable?

```
>> eig(A)

ans =

   1.0e+004 *

    -0.0001
    -8.2433
```

- We see that one of eigenvalues is ≈zero.
- Answer: Here, linearization is not valid. [We know by simulations that point is stable, but we cannot prove it by linearization].