Bioreactor: Linearization at a Specified Operating Condition.

This script demonstrates simulation and analysis of a simple bioreactor model using Matlab/Simulink. The complete package consists of the following files:

- `Bioreactor.slx` The Simulink model.
- `Bioreactor_Script.m` This script.
- `Bioreactor_schematic.png` The icon use in the Simulink model.

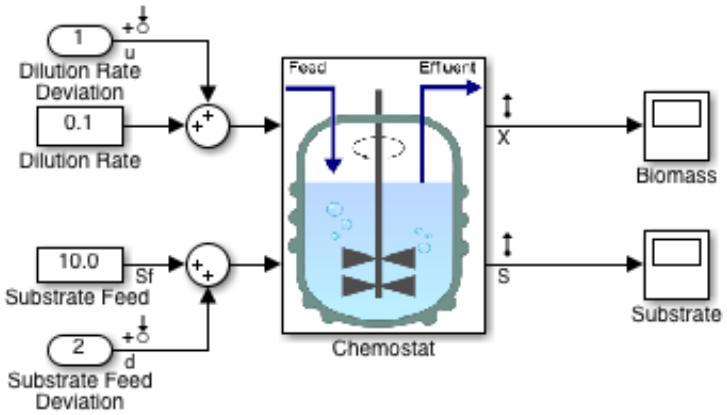
Contents

Simulink Simulation Model

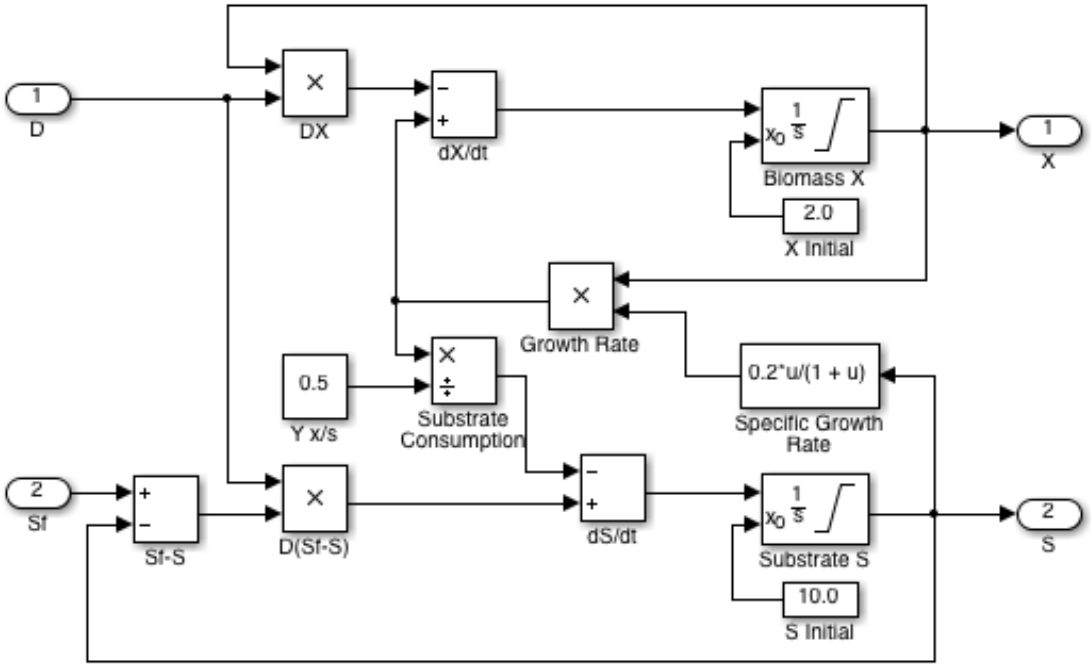
The simulation model was developed using standard simulink blocks and techniques. There are several considerations to keep in mind when developing a model that will be used for control design.

- It's normally convenient to develop models for major componnents as subsystems. At the subsystem level should go all of the detail regarding the subsytem dynamics. External inputs to the subsystems will consist of external manipulations and disturbances. Outputs should be those variables that will be measured or controlled.
- Root level input nodes should be included for each variable that can be adjusted, whether by the control system or as an external disturbance.
- Identify control and disturbance inputs and assign an input perturbance using the linear analysis points menu.
- Assign an output measurement point from the linear analysis menu to each process output.

```
open_system('Bioreactor')
```

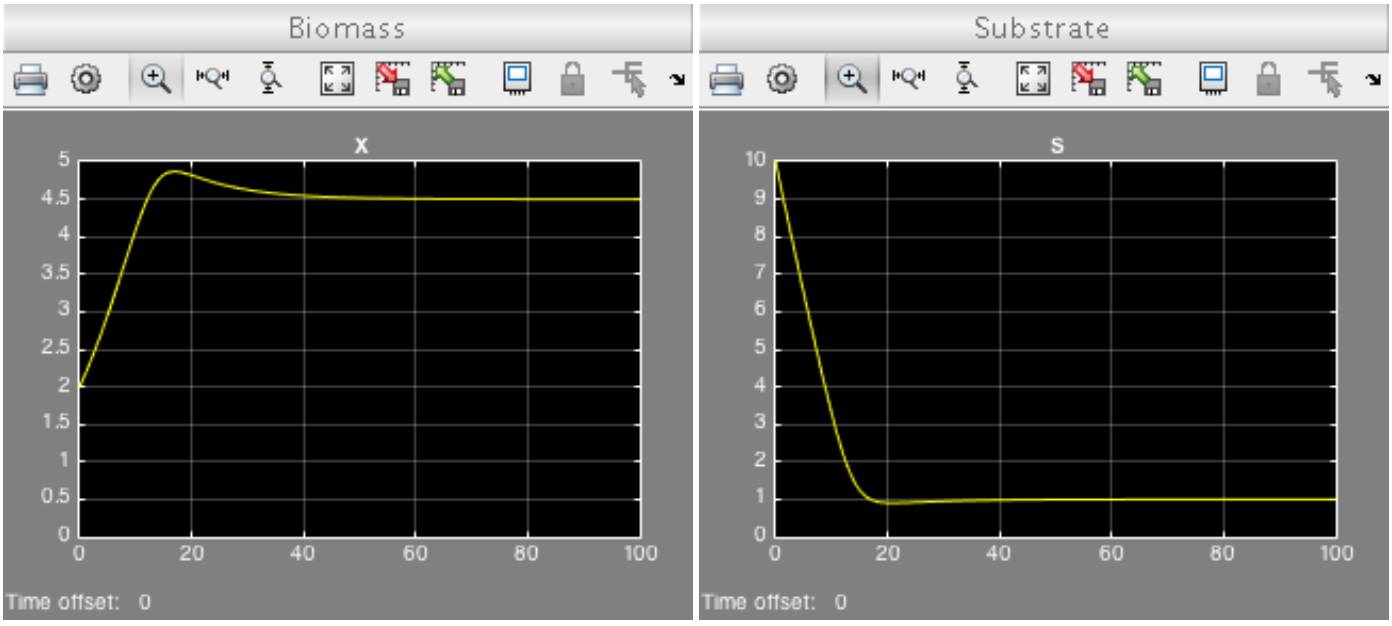


```
open_system('Bioreactor/Chemostat')
```



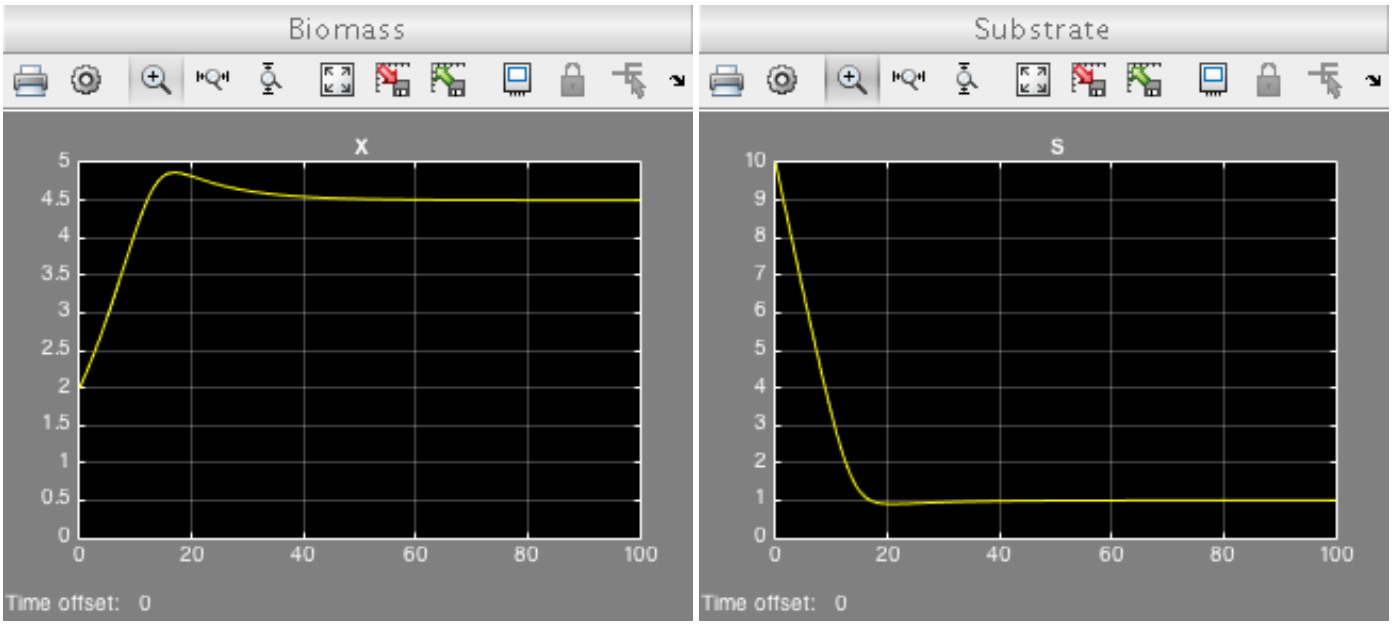
Simulation

```
% open scope windows
open_system('Bioreactor/Biomass');
open_system('Bioreactor/Substrate');

sim('Bioreactor');
```



Simulation with different initial conditions

```
% Set initial conditions
set_param('Bioreactor/Chemostat/X Initial', 'Value','2.0');
set_param('Bioreactor/Chemostat/S Initial', 'Value','10.0');

sim('Bioreactor');
```



Operating Point 1: Known Inputs

The standard procedure for finding a steady state is to establish operating specifications, then solve for the corresponding operating point. For this first case we'll assume the input deviations are zero and attempt to solve the steady state equations.

```
% close unnecessary scope windows
bdclose('Bioreactor/Biomass');
bdclose('Bioreactor/Substrate');
```

The operating specification is a Matlab data structure used to specify a desired operating point. It provides considerable flexibility which we will attempt to demonstrate below.

We start by using the `operspec` function to obtain an initial template for the operating specification. The initial default is to specify the operating point as a steady state solution of the process model.

```
opspec1 = operspec('Bioreactor')
```

```
 Operating Specification for the Model Bioreactor.
 (Time-Varying Components Evaluated at time t=0)

States:
----------
(1.) biomass
        spec:  dx = 0,  initial guess:            2
(2.) substrate
```

```
          spec:  dx = 0,  initial guess:            10

Inputs:
----------
(1.) Bioreactor/Dilution Rate Deviation
          initial guess: 0
(2.) Bioreactor/Substrate Feed Deviation
          initial guess: 0

Outputs: None
----------
```

The operating specification includes all state and inputs as decision variables. For this application there are two inputs and two states, so there are four variables that can be adjusted to solve to steady-state equations.

Here we specify that the input deviation variables are fixed to zero value.

```
opspec1.Inputs(1).Known = 1;
opspec1.Inputs(1).u = 0;
opspec1.Inputs(2).Known = 1;
opsepc1.Inputs(2).u = 0;
opspec1
```

```
 Operating Specification for the Model Bioreactor.
 (Time-Varying Components Evaluated at time t=0)

States:
----------
(1.) biomass
          spec:  dx = 0,  initial guess:             2
(2.) substrate
          spec:  dx = 0,  initial guess:            10

Inputs:
----------
(1.) Bioreactor/Dilution Rate Deviation
          u = 0
(2.) Bioreactor/Substrate Feed Deviation
          u = 0

Outputs: None
----------
```

The function `findop` attempts to find an operating point consistent with the operating specification. Given a model and an operating specification, `findop` returns a data structure describing the operating point. Let's see how it works for this example.

```
op = findop('Bioreactor',opspec)
```

```
 Operating Point Search Report:
---------------------------------

 Operating Report for the Model Bioreactor.
 (Time-Varying Components Evaluated at time t=200)

Operating point specifications were successfully met.
States:
----------
(1.) biomass
      x:           4.5      dx:      -5.3e-10 (0)
(2.) substrate
      x:             1      dx:      2.36e-10 (0)

Inputs:
----------
(1.) Bioreactor/Dilution Rate Deviation
      u:             0
(2.) Bioreactor/Substrate Feed Deviation
      u:             0

Outputs: None
----------


 Operating Point for the Model Bioreactor.
 (Time-Varying Components Evaluated at time t=200)

States:
----------
(1.) biomass
      x: 4.5
(2.) substrate
      x: 1

Inputs:
----------
(1.) Bioreactor/Dilution Rate Deviation
      u: 0
(2.) Bioreactor/Substrate Feed Deviation
      u: 0
```

Examining the solution, we find the operating specifications were satisfied. But unfortunately, `findup` found the 'wash-out' steady state, not the steady state we observed in the simulation.

Another another way to use `findop` is to take a snapshot of the simulation at a specified time, then use `initopspec` to place the resulting operating point into the opspec.

```
% Find operating point at t = 200
op1 = findop('Bioreactor',200)
```

```
 Operating Point for the Model Bioreactor.
 (Time-Varying Components Evaluated at time t=200)
```

```
States:
----------
(1.) biomass
      x: 4.5
(2.) substrate
      x: 1

Inputs:
----------
(1.) Bioreactor/Dilution Rate Deviation
      u: 0
(2.) Bioreactor/Substrate Feed Deviation
      u: 0
```

Insert the operating conditions as the initial guess for opspec

```
opspec1 = initopspec(opspec1,op)
```

```
 Operating Specification for the Model Bioreactor.
 (Time-Varying Components Evaluated at time t=200)

States:
----------
(1.) biomass
         spec:  dx = 0,  initial guess:           4.5
(2.) substrate
         spec:  dx = 0,  initial guess:             1

Inputs:
----------
(1.) Bioreactor/Dilution Rate Deviation
         u = 0
(2.) Bioreactor/Substrate Feed Deviation
         u = 0

Outputs: None
----------
```

Use `findop` to find the operating point

```
o1p = findop('Bioreactor',opspec1)
```

```
 Operating Point Search Report:
---------------------------------

 Operating Report for the Model Bioreactor.
 (Time-Varying Components Evaluated at time t=200)
```

```
Operating point specifications were successfully met.
States:
----------
(1.) biomass
      x:           4.5      dx:      -5.3e-10 (0)
(2.) substrate
      x:             1      dx:      2.36e-10 (0)

Inputs:
----------
(1.) Bioreactor/Dilution Rate Deviation
      u:             0
(2.) Bioreactor/Substrate Feed Deviation
      u:             0

Outputs: None
----------


 Operating Point for the Model Bioreactor.
 (Time-Varying Components Evaluated at time t=200)

States:
----------
(1.) biomass
      x: 4.5
(2.) substrate
      x: 1

Inputs:
----------
(1.) Bioreactor/Dilution Rate Deviation
      u: 0
(2.) Bioreactor/Substrate Feed Deviation
      u: 0
```

Operating Point 2: A Specified Output

In this case we seek an operating point where the outlet substrate concentration is a desired value S = 0.5. The starting point is to obtain an operating specification. We'll do this in a single step where we initialize an operating spec using the results of a simulation run.

```
opspec2 = initopspec( ...
    operspec('Bioreactor'), ...
    findop('Bioreactor',100))
```

```
 Operating Specification for the Model Bioreactor.
 (Time-Varying Components Evaluated at time t=100)

States:
----------
(1.) biomass
         spec:  dx = 0,  initial guess:           4.5
(2.) substrate
```

```
          spec:  dx = 0,  initial guess:             1

Inputs:
----------
(1.) Bioreactor/Dilution Rate Deviation
          initial guess: 0
(2.) Bioreactor/Substrate Feed Deviation
          initial guess: 0

Outputs: None
----------
```

There are four variables (two states, and two inputs) and two steady-state equations. Therefore we need to specify two degrees of freedom before attempting to solve the steady-state equations.

We'll fix the outlet substrate concentration to 0.5, and fix the substrate feed concentration.

```
opspec2.States(2).x = 0.5;
opspec2.States(2).Known = 1;

opspec2.Inputs(2).u = 0;
opspec2.Inputs(2).Known = 1;
```

For this case we show how to impose additional constraints on the desired operating point.

```
% The state variables must be greater than zero.
opspec2.States(1).Min = 0;
opspec2.States(2).Min = 0;

% The dilution rate must be greater than zero which establishes a lower
% limit on the value of the deviation variable for dilution rate.
opspec2.Inputs(1).Min = -str2num( ...
    get_param('Bioreactor/Dilution Rate','Value'));

op2 = findop('Bioreactor',opspec2)
```

```
 Operating Point Search Report:
---------------------------------

 Operating Report for the Model Bioreactor.
 (Time-Varying Components Evaluated at time t=100)

Operating point specifications were successfully met.
States:
----------
(1.) biomass
      x:          4.75      dx:     -3.72e-08 (0)
(2.) substrate
      x:           0.5      dx:      -1.5e-12 (0)

Inputs:
```

```
----------
(1.) Bioreactor/Dilution Rate Deviation
      u:       -0.0333    [-0.1 Inf]
(2.) Bioreactor/Substrate Feed Deviation
      u:             0

Outputs: None
----------


 Operating Point for the Model Bioreactor.
 (Time-Varying Components Evaluated at time t=100)

States:
----------
(1.) biomass
      x: 4.75
(2.) substrate
      x: 0.5

Inputs:
----------
(1.) Bioreactor/Dilution Rate Deviation
      u: -0.0333
(2.) Bioreactor/Substrate Feed Deviation
      u: 0
```

Linearization: Operating Point 1

```
io = getlinio('Bioreactor');
sys1 = linearize('Bioreactor',op1,io);
```

Linearization: Operating Point 2

```
sys2 = linearize('Bioreactor',op2,io);
```

Close windows

```
bdclose('Bioreactor');
```

Published with MATLAB® R2014b