Tutorial Examples – ICSB 2007

This tutorial will cover seven different network motifs which have dynamical properties of relevance to systems biology.

All models described in the following tutorials can be found that the tutorial web site:

http://www.sys-bio.org/sbwWiki/tutorials/icsb2007

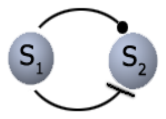
1. Homeostasis

A homeostatic system is one that resists internal change when external parameters are perturbed. We will illustrate two such networks: one that shows perfect adaptation and another that uses negative feedback.

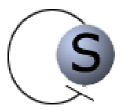
Perfect adaptation describes a system that recovers from a perturbation without any error (thus perfectly). There are a number of approaches to achieving perfect adaptation, one is via integral control and another, simpler approach, is via coordinate stimulation. In this tutorial we will illustrate perfect adaptation using coordinate stimulation.

A simpler and perhaps more common method for achieving homeostasis is to use negative feedback to resist external perturbations. Unlike systems which show perfect adaptation, systems which employ negative feedback cannot completely restore a disturbance.

Schematic Control Diagrams



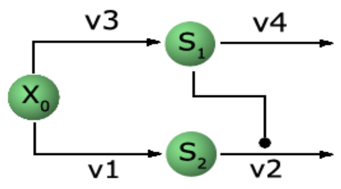
Perfect Adaptation



Negative Feedback

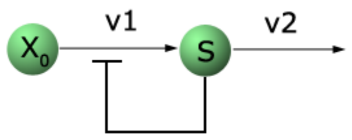
Equivalent Reaction Diagrams

S_2 remains homeostatic



$$v1 = k_1 X_0 \qquad v2 = k_2 S_1 S_2$$
$$v3 = k_3 X_0 \qquad v4 = k_4 S_1$$

S remains homeostatic



$$v1 = \frac{X_0}{K_m + S^4} \qquad v2 = k_1 S$$

Perfect Adaptation **Negative Feedback**

Exercise:

Perfect adaptation, the following Jaranc script describes a model that shows perfect adaptation:

Jarnac code:

```
// model
p = defn cell
      $Xo   -> S2;  k1*Xo;
       S2   -> $w;  k2*S1*S2;
       $Xo -> S1;  k3*Xo;
       S1   -> $w;  k4*S1;

end;

// initialize
p.k1 = 1;
p.k2 = 1;
p.k3 = 1;
p.k4 = 1;
p.Xo = 1.0;
```

To observe how the system recovers from a perturbation:

0. Enter the above model into Jarnac or download it from the tutorial web site (Use copy and paste to copy the model from the web site to the Jarnac editor window).
1. From the SBW menu in Jarnac, select "kgi.edu.roadRunner Simulation Service".
2. Select the "Time course (continuous)" tab. Click Start.
3. From the Options menu, select "Sliders". Under the boundary variables, you will find X_0

4. Change X_0 and observe the manner in which the two concentrations are disturbed, and then observe how S_2 recovers to the original state.
5. Alternatively, you can use Jarnac to generate perturbations. Suppose one needs to introduce a perturbation at time 30, then one would simulate the network from time 0 to time 30. Then one of the values would be altered (perturbed). Calling the simulate command once again, but this time starting from time 30, would be identical to introducing a sudden perturbation at time 30. Here is the code for such a perturbation (you can also see the code on the next page).

```
 p.Xo = 2.0;
m1 = p.sim.eval (0, 30, 100, [<p.Time>, <p.Xo>, <p.s1>, <p.s2>]);
p.Xo = 3.0;    //perturb Xo
m2 = p.sim.eval (30, 50, 100, [<p.Time>, <p.Xo>, <p.s1>, <p.s2>]);
```

6. Challenge: A molecular species S is at steady state when the total flow out of that species equals the total flow into it. Can you find the value of S_1 when it reaches steady state? Can you find the value of S_2 when it reaches steady state? Does this show why S_2 remains unchanged when X_0 changes?

Negative feedback system:

1. In order to see why a negative feedback produces homeostatic behavior, one needs to see how the flow into S_2 and the flow out of S_2 are connected to the concentration of S_2. Open the Simulation Service (SBW menu -> kgi.edu.roadRunner Simulation Service).
2. Select the "Rate vs. Species" tab. Select S_2 for the "Species", and plot. The two curves indicate the flow in and flow out of S_2.
3. Use the sliders (Options -> Sliders) to see how the curves are affected by the parameters.
4. (Challenge) Can you find the qualitative features of the two curves that are needed to maintain homeostasis?

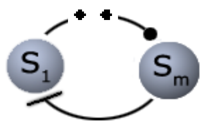
```
// Jarnac Code: Perfect adaptation

p = defn cell
     v1: $Xo -> S2;  k1*Xo;
     v2:  S2 -> $w;  k2*S1*S2;
         $Xo -> S1;  k3*Xo;
          S1 -> $w;  k4*S1;
end;

// Initialize
p.k1 = 1;
p.k2 = 1;
p.k3 = 20;
p.k4 = 1;

// Perform 5 different simulations for 5 different values of Xo
p.Xo = 1.0;
m1 = p.sim.eval (0, 30, 100, [<p.Time>, <p.Xo>, <p.S1>, <p.S2>]);

p.Xo = 2.0;
m2 = p.sim.eval (30, 50, 100, [<p.Time>, <p.Xo>, <p.S1>, <p.S2>]);

p.Xo = 4.0;
m3 = p.sim.eval (50, 60, 100, [<p.Time>, <p.Xo>, <p.S1>, <p.S2>]);

p.Xo = 6.0;
m4 = p.sim.eval (60, 70, 100, [<p.Time>, <p.Xo>, <p.S1>, <p.S2>]);

p.Xo = 8.0;
m5 = p.sim.eval (70, 80, 100, [<p.Time>, <p.Xo>, <p.S1>, <p.S2>]);

// Augment the results of the simulation
m = augr (m1, m2);
m = augr (m, m3);
m = augr (m, m4);
m = augr (m, m5);

// plot all the results together
graph (m);
```

```
// Jarnac Code: Homeostatic network, Negative Feedback

p = defn NegativeFeedback
      $Xo -> S;   Xo/(km + S^h);
        S -> $w;  k1*S;
end;

// initialize
p.h = 4;   // Hill coefficient
p.k1 = 1;
p.km = 1;
p.S = 1.5;

// Perform different simulations for different values of Xo
p.Xo = 5.0;
m1 = p.sim.eval (0, 20, 100, [<p.Time>, <p.Xo>, <p.S>]);

p.Xo = 6.0;
m2 = p.sim.eval (20, 40, 100, [<p.Time>, <p.Xo>, <p.S>]);

p.Xo = 7.0;
m3 = p.sim.eval (40, 60, 100, [<p.Time>, <p.Xo>, <p.S>]);

p.Xo = 8.0;
m4 = p.sim.eval (60, 80, 100, [<p.Time>, <p.Xo>, <p.S>]);

// Augment the results of the simulation
m = augr (m1, m2);
m = augr (m, m3);
m = augr (m, m4);
// Plot all the results together
graph (m);
```

2. Oscillation I: Negative Feedback

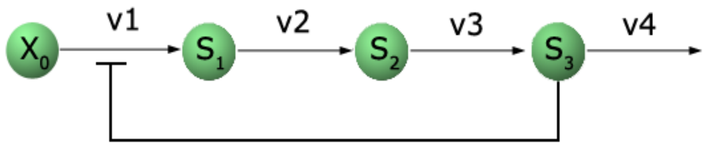
The correct amount of feedback can maintain homeostasis. However, when the feedback is too strong, it can cause oscillations in the system.

Schematic Control Diagrams (note: there may be multiple steps in-between each arrow)



m=3 for the reaction below.

Reaction Diagram



$$v1 = \frac{k_0 X_0}{k_m + S_3^{\ h}} \qquad v2 = k_1 S_1 \qquad v3 = k_2 S_2 \qquad v4 = k_3 S_3 \qquad v5 = k_4 S_4$$

Trial parameter values: n=5, $k_0=k_1=k_2=k_3=k_m=1$, $X_0=10$.

Exercise:

1. Can you get sustained oscillations in s_1, s_2, s_3? If not, increase the strength of feedback by altering the different parameters. The **Oscill8** program can be used to find the critical values of parameters that indicate when oscillations can occur. The Oscill8 program can be executed from the SBW menu in Jarnac.

2. Using **Oscill8** to find the critical points:

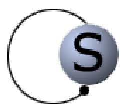
    a. Select “Oscill8 GUI” from the SBW menu in Jarnac. Go to the “Run” menu and select “Parameter”. Choose the parameter you would like to vary (try the exponent h), and click “Run”. The output shows you the critical point(s) beyond which the system begins to oscillate.

```
// Jarnac Code
// Negative Feedback - oscillator

// model
p = defn cell
      $Xo -> S1;  k0*Xo/(km + S3^h);
       S1 -> S2;  k1*S1;
       S2 -> S3;  k2*S2;
       S3 -> $w;  k3*S3;
end;

// Initialize
p.h = 6;   // Hill coefficient
p.k0 = 1;
p.k1 = 1;
p.k2 = 1;
p.k3 = 1;
p.km = 1;
p.Xo = 10;

// Simulate
m = p.sim.eval (0, 100, 1000, [<p.Time>, <p.S1>, <p.S2>, <p.S3>]);

// Plot
graph(m);
```

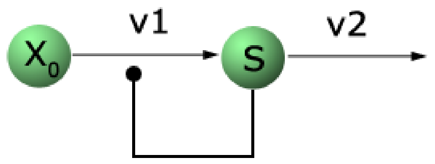
3. Positive Feed-back System

Positive feedback is a common approach to generate bi-stable (a system with two stable states) systems.

A Schematic Control Diagram



A Reaction Diagram



$$v1 = v_{\max} \frac{X_0 S^h}{k_m + S^h} + k_0 \qquad v2 = k_1 S$$

Trial parameter values: h=4, X_0=1, k_m=0.1, k_0=0.1, k=1.0

Exercise:

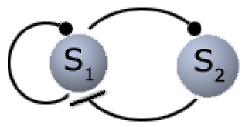
1. Use the Simulation Service to plot the two rates (v1 and v2) against S. S is at steady state when v1 is equal to v2 (i.e. when the two curves intersect). Can you use this plot to predict the two stable states of S? Can you predict which initial values of S will reach the two stable states?
2. (Challenge) Use the plot of v1 and v2 to find a condition where the system does not exhibit bi-stability. Use Jarnac to do multiple simulations with different initial conditions of S to confirm that there is only one steady state value.

```
// Jarnac Code
// Positive Feedback - bistability
// model
p = defn cell
      v1: $Xo -> S;  vmax*S^h/(km + S^h) + k0;
       v2: S -> $w;  k1*S;
end;

// Initialize
p.h = 4;   // Hill coefficient
p.vmax = 1;
p.km = 0.1;
p.k0 = 0.1;
p.k1 = 1;

// Perform different Simulations for different values of S
p.S = 0.3;
m1 = p.sim.eval (0, 10, 100, [<p.Time>, <p.S>]);

p.S = 0.4;
m2 = p.sim.eval (0, 10, 100, [<p.S>]);

p.S = 0.5;
m3 = p.sim.eval (0, 10, 100, [<p.S>]);

p.S = 0.6;
m4 = p.sim.eval (0, 10, 100, [<p.S>]);

p.S = 0.7;
m5 = p.sim.eval (0, 10, 100, [<p.S>]);

// Augment the results of the simulation
m = aug (m1, m2);
m = aug (m, m3);
m = aug (m, m4);
m = aug (m, m5);

// Plot all the results together
graph(m);
```

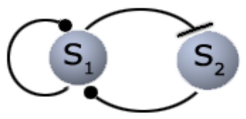
4. Relaxation Oscillators (Positive Feedback + Negative Feedback)

A relaxation oscillator utilizes a positive feedback coupled to a negative feedback loop. The reaction diagram can be interpreted as an extension to the positive feedback bi-stable system.

Schematic Control Diagrams

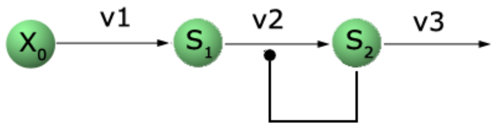


A.



B.

Reaction Diagram for control diagram A.



$$v1 = k_1 X_0 \qquad v2 = \frac{k_2 S_1 S_2{}^h}{k_m + S_2{}^h} + k_3 S_1 \qquad v3 = k_4 S_2$$

Trail parameter values: p.h=1, p.Xo=1, p.k2=2, p.k3=0.02, p.k4=1.0

Exercise:

1. Simulate the reaction network. If you do not get sustained oscillation, try altering some of the parameters to obtain an oscillation. Which parameter do you think has the most significant effect?

2. It is often useful to plot one species against another to see how they affect one another. The p.sim.eval function can be used to generate such a plot:

```
m = p.sim.eval (0, 100, 1000, [<p.S1>, <p.S2>]);
graph(m)
```

```
// Jarnac Code
// Positive Feedback - relaxation oscillator

// model
p = defn cell
      v1: $Xo -> S1;  k1*Xo;
      v2:  S1 -> S2;  k2*S1*S2^h/(10 + S2^h) + k3*S1;
      v3:  S2 -> $w;  k4*S2;
end;

// Initialize
p.h = 2;   // Hill coefficient
p.k1 = 1;
p.k2 = 2;
p.k3 = 0.02;
p.k4 = 1;

// Simulate
m = p.sim.eval (0, 100, 1000, [<p.Time>, <p.S1>, <p.S2>]);

// Plot
graph(m);
```

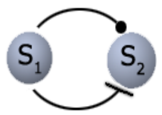
5. Incoherent Feed-forward: Concentration Detector

An incoherent feed-forward network is where the input positively and negatively controls one of the species. The rate of positive and negative control allows the system to produce a behavior that is called a “concentration detector”.

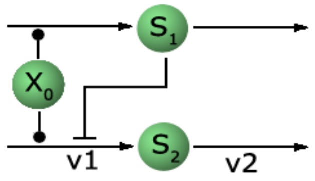
A concentration detector is a network such that the reponding molecule has a low concentration (“off”) when the input signal is too low or when it is too high. The output (S_2) molecule has a high concentration (“on”) only when the input signal is within a certain interval.

This same network can also act like a band-pass filter. In other words, a signal with very high or very low frequencies will not pass well through the network, but a frequency in the middle range will.

A schematic control diagram:



Reaction diagrams:



$$v1 = \frac{k_3 X_0}{1 + k_3 X_0 + k_4 X_0 S_1}$$

$$v2 = k_5 S_2$$

Exercise:

1. Use Jarnac to confirm that this model is a concentration detector. Use a loop to test different values of X_0. The function p.ss.eval can be used to bring the system to steady state. Make a matrix of different values of X_0 and the corresponding steady state values of S. Plot this matrix to see whether or not this network is a concentration detector.

2. Use Frequency Analyzer (from the SBW menu) to see whether this network can act like a band-pass filter (a filter that responds only in a middle range of frequencies). Note: it is important to set the value of S at the end of your Jarnac code. This value of S is used in the Frequency Analyzer.

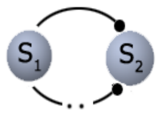
3. (Challenge) The steady state values of $\mathbf{S_2}$ occur when **v1 = v2**. Find the steady state value of $\mathbf{S_2}$ as a function of $\mathbf{X_0}$. Sketch the function; does it illustrate why this network acts like a concentration detector?

```
// Jarnac Code
// Concentration detector, band-pass filter

// model
p = defn cell

     $Xo -> S1; k1*Xo;
      S1 -> $w;  k2*S1;

    v1: $Xo -> S2; k3*Xo/(1 + k3*Xo + k4*Xo*S1);
    v2:  S2 -> $w; k5*S2;

end;

// Parameters
p.k1 = 1; p.k2 = 1; p.k3 = 1; p.k4 = 1; p.k5 = 1;

// Initialize
p.Xo = 0;
p.S1 = 0;
p.S2 = 0;

m = matrix(100,2);    //make matrix
for i = 1 to 100 do   //fill matrix
begin
   p.Xo = p.Xo + 0.1;    // Different values of Xo
   p.S1 = 0;
   p.S2 = 0;
   p.ss.eval;         //obtain steady state values
   m[i,1] = p.Xo;    //store concentrations in the matrix
   m[i,2] = p.S2;
end

graph(m);    // Plot S2 vs Xo

p.Xo = 1;    // Get the "active" region of Xo for the frequency
                analyzer
```

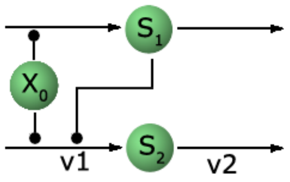
6. Coherent Feed-forward: A Low-pass Filter

A coherent feed-forward network is one where the input signal positively regulates the output signal through two different routes. The input signal directly regulates the output as well as regulates it through an intermediate step. This double activation has a unique effect – it is capable of eliminating signals that are of very high frequency (such as noise). Thus, a coherent feed-forward network motif acts as a low-pass filter.

A schematic control diagram



A reaction diagram



$$v1 = k_3 X_0 + k_4 X_0 S_1 \qquad v2 = k_5 S_2$$

Exercise:

1. Look at the frequency response of the network (use Frequency Analyzer).
2. (Challenge) Conceptually, imagine a signal starting at $\mathbf{X_0}$ and traveling to $\mathbf{S_1}$ and $\mathbf{S_2}$, and from $\mathbf{S_1}$ to $\mathbf{S_2}$. Consider a low frequency oscillating signal and a high frequency signal (such as noise). Can you give a qualitative reason why such a network might not respond well to a high frequency signal?

```
// Jarnac Code
// Feed forward, low-pass filter

// model
p = defn cell

    $Xo -> S1;  k1*Xo;
     S1 -> $w;  k2*S1;

   v1: $Xo -> S2; k3*Xo + k4*Xo*S1;
   v2:  S2 -> $w; k5*S2;

end;

// parameters
p.k1 = 1; p.k2 = 1; p.k3 = 1; p.k4 = 1; p.k5 = 1;

// initialize
p.S1 = 0;
p.S2 = 0;
p.Xo = 5;
```