D.7 Autoregulatory Gene Model

Zarif Ahmed

February 2f025

Abstract

In this study, we analyze an autoregulatory gene modeled using the Hill equation. We explore how different initial conditions impact the trajectory of the model. We find that gene needs to reach a certain environmental threshold of protein and RNA concentration before the gene can become self-regulated. Otherwise any amount below the threshold leads to the system collapsing and halting any RNA and protein production. These results highlight the importance of initial conditions in determining the stability and functionality of autoregulatory gene systems.

Introduction

An autoregulatory gene is one that encodes a protein whose function (or part of whose function) is to regulate its own transcription. For activating regulation modeled with the Hill equation, a system of differential equations is:

$$\frac{d[X_r]}{dt} = \frac{\mu[X_p]^h}{K_{1/2}^h + [X_p]^h} - \chi_r[X_r]$$

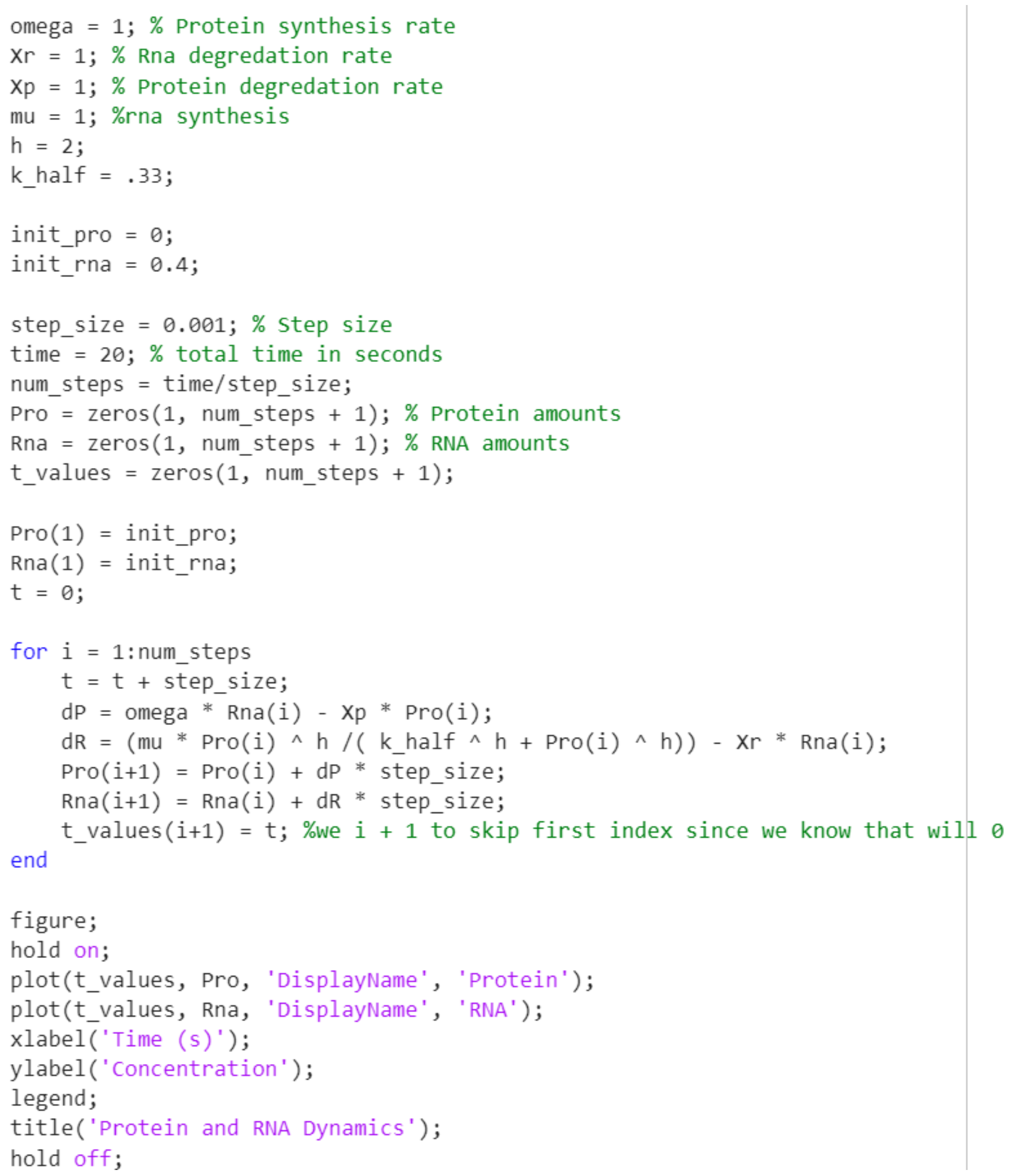
$$\frac{d[X_p]}{dt} = \omega[X_r] - \chi_p[X_p]$$

where:

- $[X_r]$: mRNA concentration.

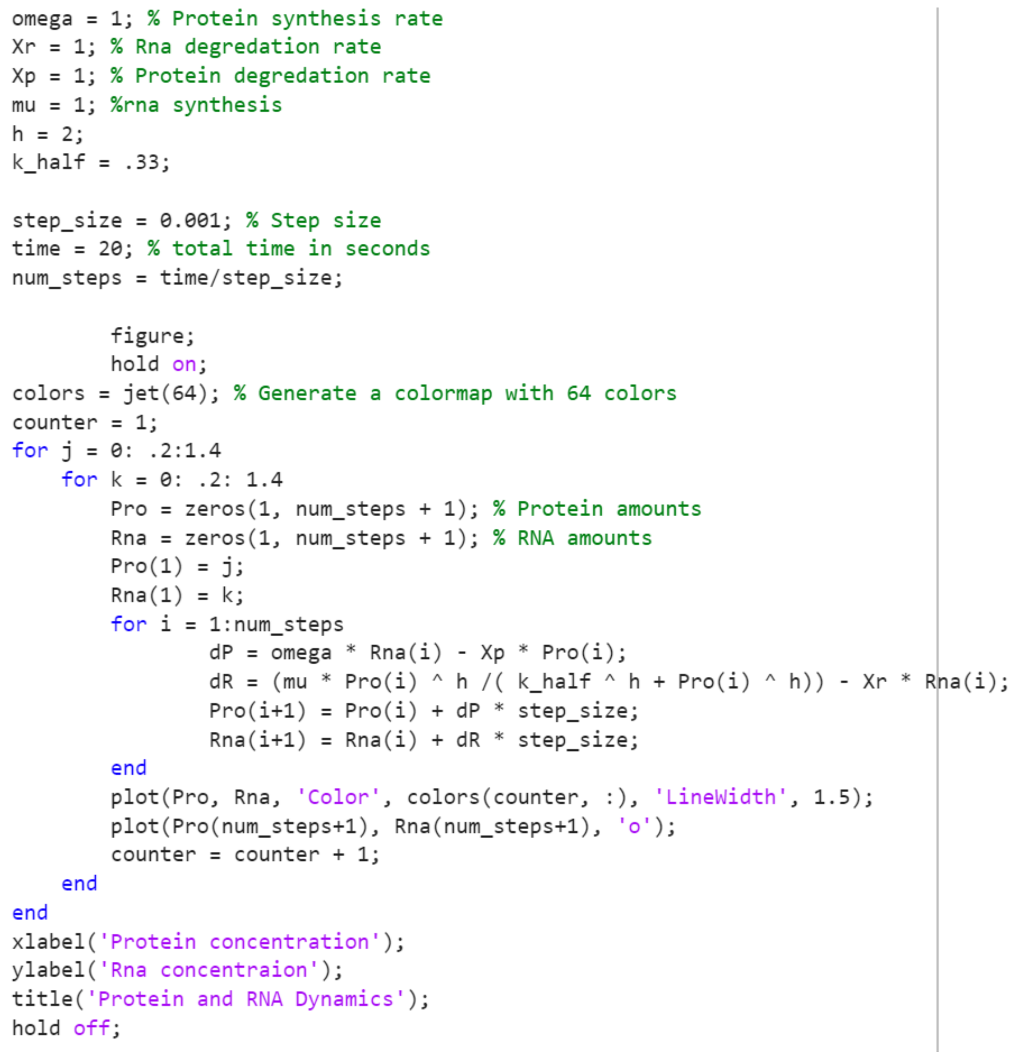
- $[X_p]$: Protein concentration.
- μ: maximal transcription rate.
- $K_{1/2}$: half-maximal activation concentration.
- h: Hill coefficient, determines the degree of cooperativity.
- χ_r: mRNA degradation rate.
- ω: Protein translation rate.
- χ_p: Protein degradation rate.

In this study we will simulate this system using the Forward Euler method and explore how different initial conditions influence the system's trajectory and whether it reaches a steady state. In addition, we analyze null clines and stationary points to gain insights into the system's stability and how it relates biologically. Figures 1 and 2 give the codes used to model simulate and generated graphs for the report.



```
omega = 1; % Protein synthesis rate
Xr = 1; % Rna degredation rate
Xp = 1; % Protein degredation rate
mu = 1; %rna synthesis
h = 2;
k_half = .33;

init_pro = 0;
init_rna = 0.4;

step_size = 0.001; % Step size
time = 20; % total time in seconds
num_steps = time/step_size;
Pro = zeros(1, num_steps + 1); % Protein amounts
Rna = zeros(1, num_steps + 1); % RNA amounts
t_values = zeros(1, num_steps + 1);

Pro(1) = init_pro;
Rna(1) = init_rna;
t = 0;

for i = 1:num_steps
    t = t + step_size;
    dP = omega * Rna(i) - Xp * Pro(i);
    dR = (mu * Pro(i) ^ h /( k_half ^ h + Pro(i) ^ h)) - Xr * Rna(i);
    Pro(i+1) = Pro(i) + dP * step_size;
    Rna(i+1) = Rna(i) + dR * step_size;
    t_values(i+1) = t; %we i + 1 to skip first index since we know that will 0
end

figure;
hold on;
plot(t_values, Pro, 'DisplayName', 'Protein');
plot(t_values, Rna, 'DisplayName', 'RNA');
xlabel('Time (s)');
ylabel('Concentration');
legend;
title('Protein and RNA Dynamics');
hold off;
```

Figure 1: Code to generate Protein Concentration vs time plot



```
omega = 1; % Protein synthesis rate
Xr = 1; % Rna degredation rate
Xp = 1; % Protein degredation rate
mu = 1; %rna synthesis
h = 2;
k_half = .33;

step_size = 0.001; % Step size
time = 20; % total time in seconds
num_steps = time/step_size;

        figure;
        hold on;
colors = jet(64); % Generate a colormap with 64 colors
counter = 1;
for j = 0: .2:1.4
    for k = 0: .2: 1.4
        Pro = zeros(1, num_steps + 1); % Protein amounts
        Rna = zeros(1, num_steps + 1); % RNA amounts
        Pro(1) = j;
        Rna(1) = k;
        for i = 1:num_steps
                dP = omega * Rna(i) - Xp * Pro(i);
                dR = (mu * Pro(i) ^ h /( k_half ^ h + Pro(i) ^ h)) - Xr * Rna(i);
                Pro(i+1) = Pro(i) + dP * step_size;
                Rna(i+1) = Rna(i) + dR * step_size;
        end
        plot(Pro, Rna, 'Color', colors(counter, :), 'LineWidth', 1.5);
        plot(Pro(num_steps+1), Rna(num_steps+1), 'o');
        counter = counter + 1;
    end
end
xlabel('Protein concentration');
ylabel('Rna concentraion');
title('Protein and RNA Dynamics');
hold off;
```

Figure 2: Code to generate RNA Concentration vs protein Concentration plot

Results and Discussion

To start of we will explore how different starting concentrations impact effect the trajectory of our model of an autoregulatory gene.In Figure one we show the implementation of a Forward Euler model to determine the concentration of both RNA and protein over a time of 20 seconds. We set a time step of .01 seconds and model parameters of h = 2, $K_{half} = .33$ mM and $\mu = 1$ mMs/s, $omega = 1$ mM/s and $\chi_r = \chi_p = 1$ mM/s. Figure 3, 4, 5 and 6 show the results of running our model with different initial conditions.

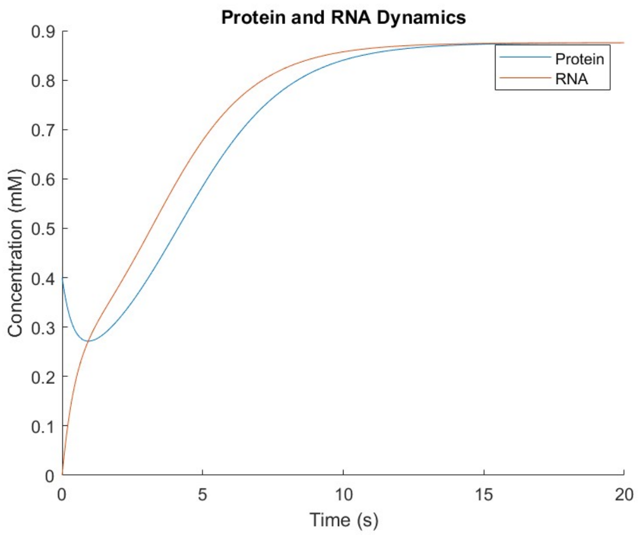


Figure 3: Starting $X_r = 0$, $X_p = 0.4$

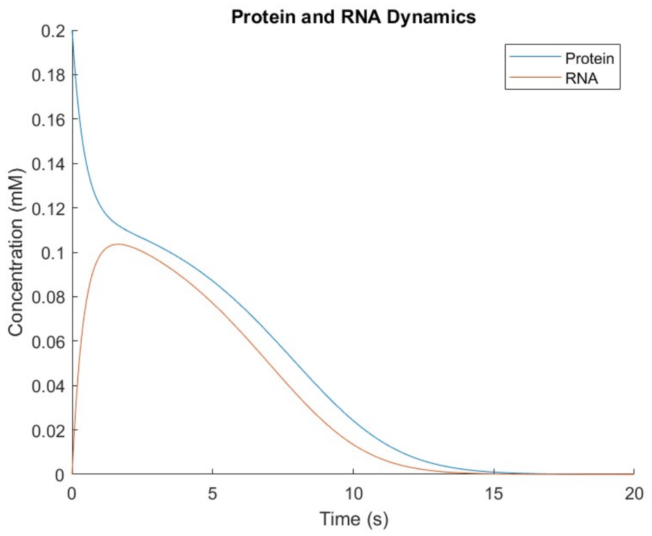


Figure 4: Starting $X_r = 0$, $X_p = 0.2$

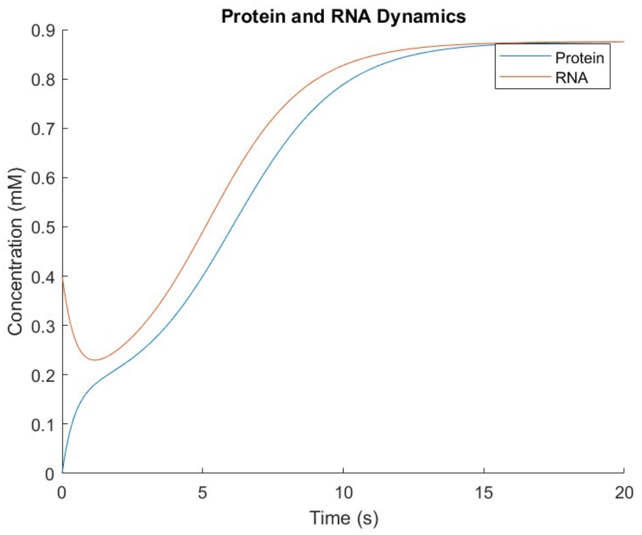


Figure 5: Starting $X_r = 0.4$, $X_p = 0$

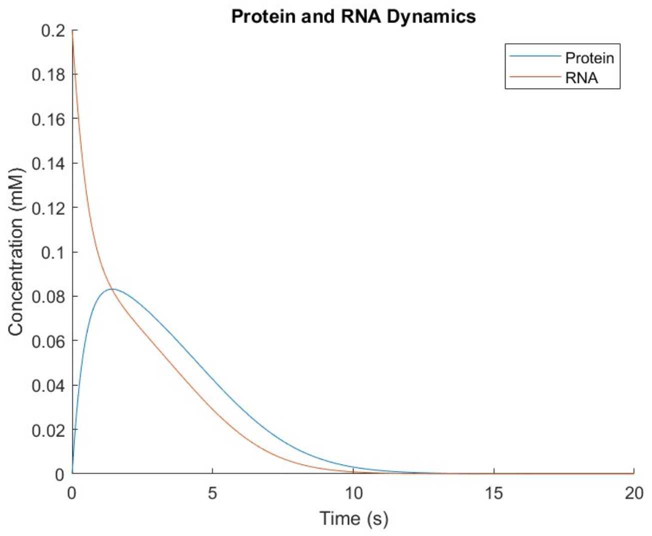


Figure 6: Starting $X_r = 0.2$, $X_p = 0$

From Figures 3, 4, 5, and 6, we observe a few key trends. When either RNA or protein starts at a concentration of zero while the other molecule has a very low initial concentration, the system eventually collapses on itself. This occurs because the insufficient starting amounts fail to sustain the transcription and translation processes, leading to the concentration of both molecules over time reaching zero. Another observed trend is that when the system has a sufficiently high initial concentration, then system consistently reaches the same equilibrium state. In all cases, both protein and RNA concentrations stabilize at 0.9 mM, indicating a steady-state balance between transcription and translation processes.

To better understand how initial condition impacts the trajectory of our model, we modified the code shown in Figure 1. By embedding the original code in an outer double loop such that the simulation may be repeated with many combinations of concentrations of both protein and RNA, with the results plotted on a single plot of χ_r vs χ_p. We vary the starting concentrations of both protein and mRNA from 0.0 to 1.4 mM in intervals of 0.2 mM and consider all combinations of these. The resulting code is shown in Figure 2 and the graph generated from running the code is shown in Figure 7.

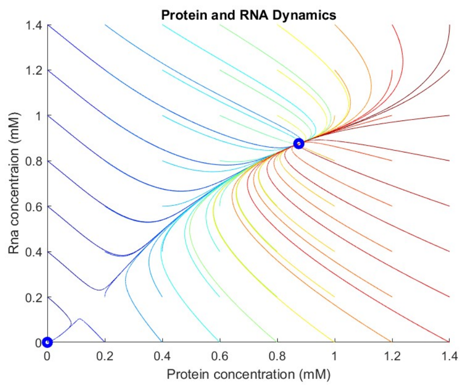


Figure 7: Graph generated by running code from Figure 2

We added a piece of code that draws a blue circle at the last point of each line to help highlight where each line ends. Two distinct points emerge as a result indicating that all plotted lines end up at one or the other. These two points are the equilibrium points in our system. Again, if there is not enough starting RNA or protein then the system collapses as indicated by the equilibrium point at (0,0) in Figure 7. However, if the initial conditions are in sufficient concentration then the system approaches the second equilibrium point roughly at/around (.9, .9). This is a stable steady state where both RNA and protein reach a sustained concentration. Even if the starting conditions are higher than the equilibrium point, the system always approaches that point.

To better illustrate the direction of the system's dynamics on the graph, we can plot the null clines and stationary points. The null clines are curves where one of the derivatives is zero while stationary points are where all derivatives in the system are at 0.

- RNA nullcline: Set $\frac{d[X_r]}{dt} = 0$, which gives:

$$\frac{\mu[X_p]^h}{K_{1/2}^h + [X_p]^h} - \chi_r[X_r] = 0$$

Solving for $[X_r]$:

$$X_r = \frac{\mu[X_p]^h}{\chi_r(K_{1/2}^h + [X_p]^h)}$$

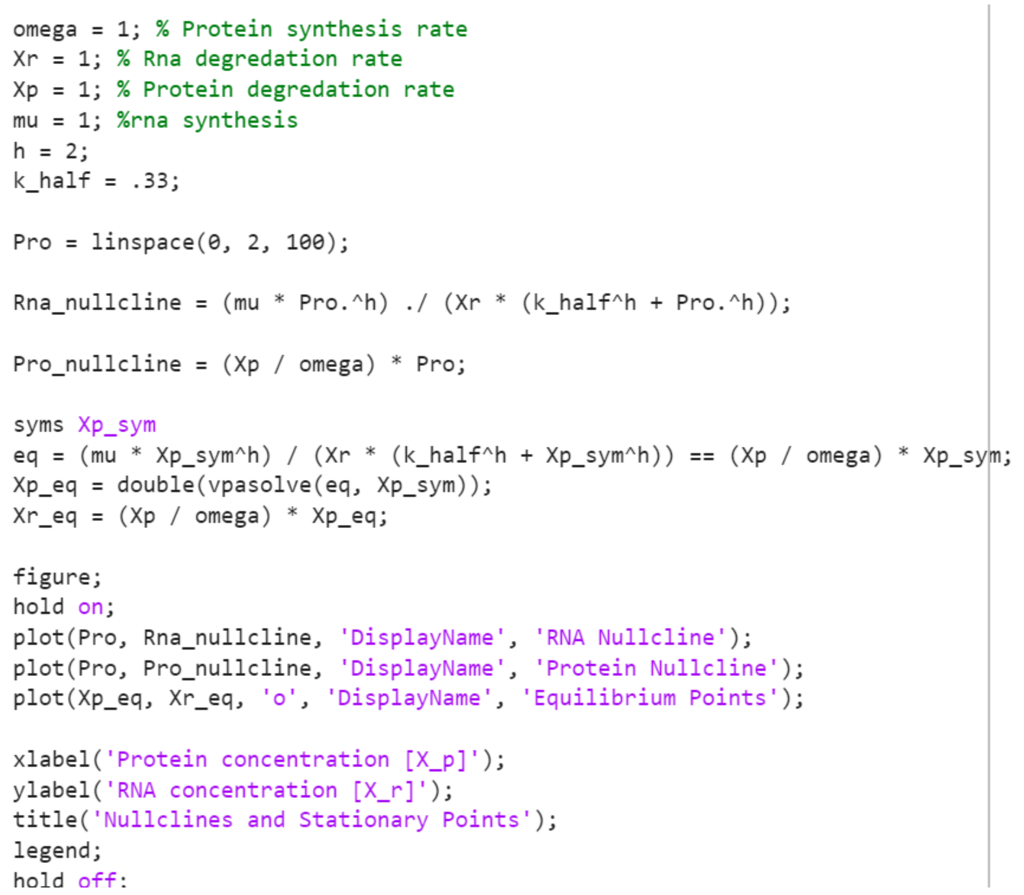
- Protein nullcline: Set $\frac{d[X_p]}{dt} = 0$, which gives:

$$\omega[X_r] - \chi_p[X_p] = 0$$

  Solving for $[X_r]$:

$$X_r = \frac{\chi_p}{\omega} X_p$$

We graph model the null clines and stationary point using MATLAB. The code used for this study is shown in Figure 8 and the graph generated by the code is shown in Figure 9.



```
omega = 1; % Protein synthesis rate
Xr = 1; % Rna degredation rate
Xp = 1; % Protein degredation rate
mu = 1; %rna synthesis
h = 2;
k_half = .33;

Pro = linspace(0, 2, 100);

Rna_nullcline = (mu * Pro.^h) ./ (Xr * (k_half^h + Pro.^h));

Pro_nullcline = (Xp / omega) * Pro;

syms Xp_sym
eq = (mu * Xp_sym^h) / (Xr * (k_half^h + Xp_sym^h)) == (Xp / omega) * Xp_sym;
Xp_eq = double(vpasolve(eq, Xp_sym));
Xr_eq = (Xp / omega) * Xp_eq;

figure;
hold on;
plot(Pro, Rna_nullcline, 'DisplayName', 'RNA Nullcline');
plot(Pro, Pro_nullcline, 'DisplayName', 'Protein Nullcline');
plot(Xp_eq, Xr_eq, 'o', 'DisplayName', 'Equilibrium Points');

xlabel('Protein concentration [X_p]');
ylabel('RNA concentration [X_r]');
title('Nullclines and Stationary Points');
legend;
hold off;
```

Figure 8: Code to plot the null clines and stationary points

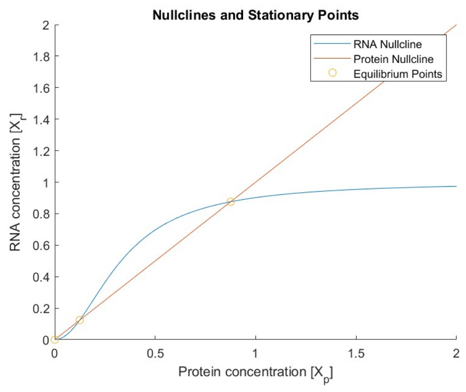


Figure 9: Graph generated by running code from Figure 8

Plotting the null clines reveals 3 stationary points. The point at (0,0) is a trivial stationary point. When we start with very low protein and RNA concentrations the system starts to collapse and move towards the trivial stationary point as seen in Figure 7. We also have a stationary point at (.88, .88) which matches the 2nd point we see in figure 7. This shows that when initial conditions are sufficient, the system regulates itself and stabilizes at the non-trivial stationary point at (.88, .88). There is also a third equilibrium point at (.12, .12) that no line in Figure 7 approaches. This is because this third point is an unstable point. This third point represents the protein and RNA concentration threshold our system needs to cross such that the system no longer collapses and starts to move towards a stable equilibrium.

Conclusion

Our analysis of the systems highlights the importance of initial conditions in determining the stability of autoregulatory systems. However, once the system crosses that threshold, the system comfortably maintains equilibrium and turns self sustaining. The model is able to predict the general function of an autoregulatory gene in isolation however, future studies could explore the affects of external factors on the gene and how it impacts protein production.